Neutral Citation Number: [2025] EWHC 532 (TCC)

Case No: HT-2021-000363

IN THE HIGH COURT OF JUSTICE
BUSINESS AND PROPERTY COURTS OF ENGLAND AND WALES
TECHNOLOGY AND CONSTRUCTION COURT (KBD)

Royal Courts of Justice
Rolls Building
London, EC4A 1NL

Date: 10 March 2025

Before:

Mrs Justice O'Farrell DBE

- -

Between:

IBM UNITED KINGDOM LIMITED	**Claimant**
- and -	
(1) LZLABS GmbH **(a company incorporated in Switzerland)** **(2) WINSOPIA LIMITED** **(3) LZLABS LIMITED** **(4) MARK JONATHAN CRESSWELL** **(5) THILO ROCKMANN** **(6) JOHN JAY MOORES**	**Defendants**

- -
- -

Nicholas Saunders KC, Matthew Lavy KC, Fred Hobson KC, James Weale, Laura Wright, Henry Edwards, Alex Taylor & Jacob Haddad (instructed by **Quinn Emanuel Urquhart & Sullivan UK LLP**) for the **Claimant**
Roger Stewart KC, Mark Vanhegan KC, Thomas Ogden, Jaani Riordan, George McDonald & Joshua Marshall (instructed by **Clifford Chance LLP**) for the **Defendants**

Reading dates: 22nd, 23rd, 24th, 25th & 26th April 2024
Hearing dates: 29th, 30th April 2024
1st, 2nd, 7th, 8th, 9th, 13th, 14th, 15th, 16th, 20th, 21st, 22nd, 23rd,
28th, 29th, 30th May 2024

3rd, 4th, 5th, 6th, 10th, 11th, 12th, 13th, 17th, 18th, 19th June 2024
1st, 2nd, 3rd, 4th July 2024
Circulation of draft Judgment: 21st February 2025

- -

JUDGMENT

This judgment was handed down remotely at 10am on Monday 10th March 2025 by circulation to the parties or their representatives by e-mail and by release to the National Archives.

Mrs Justice O'Farrell:

Section I - Introduction

1. This claim concerns allegations of reverse engineering arising out of the development of software, known as the Software Defined Mainframe ("the SDM"), which is said to enable its customers to take business applications developed for IBM mainframe computers and run them on x86-based computer architectures using the Linux operating system, without the need for source code changes or recompilation. One of the central issues is whether the defendants are entitled to rely on rights of observation, studying and testing and/or interoperability, conferred by Directive 2009/24/EC ("the Software Directive"), implemented by the Copyright, Designs and Patents Act 1988 ("the CDPA"), by way of defence to the claim.

2. The claimant ("IBM"), a subsidiary of International Business Machines Corporation ("IBM Corp"), is a supplier of computer hardware and software, and licensor of IBM mainframe software within the UK.

3. The first defendant ("LzLabs"), a Swiss company incorporated in 2011, is the developer and supplier of the SDM.

4. The second defendant ("Winsopia"), a company incorporated in England and Wales in 2013, is a wholly owned subsidiary of LzLabs. In 2013 Winsopia purchased an IBM mainframe computer and entered into a licence agreement with IBM in respect of IBM mainframe software.

5. The third defendant ("LzLabs UK"), a company incorporated in England and Wales in 2015, is a wholly owned subsidiary of LzLabs, providing specialist technical support services.

6. The fourth defendant, Mark Cresswell, is a non-executive chairman and former CEO of LzLabs, and a former director of Winsopia and LzLabs UK.

7. The fifth defendant, Thilo Rockmann, is the CEO of LzLabs, a director of Winsopia and a director of LzLabs UK.

8. The sixth defendant, John Jay Moores, is a software entrepreneur who financed the development of the SDM and is the main beneficial owner of LzLabs, Winsopia and LzLabs UK.

9. On 15 August 2013 IBM licensed IBM mainframe software to Winsopia pursuant to an IBM customer agreement ("the ICA"). Additional and updated licence agreements were entered into by IBM and Winsopia, subject to the same material terms and conditions.

10. IBM's primary case is that the defendants breached, or procured breach of, the ICA, using Winsopia's access to the IBM mainframe software to develop the SDM by unlawful reverse engineering of the licensed software.

11. In December 2020/January 2021, IBM requested an audit of Winsopia's compliance with the terms of the ICA. Winsopia refused to accede to the request

on the grounds that the request exceeded the ambit of IBM's contractual audit rights and sought to impose unreasonable demands within an unreasonable timescale for compliance.

12. By letter dated 24 February 2021, IBM purported to terminate the ICA and other licence agreements for contractual breach; alternatively at common law. By letter dated 1 March 2021, Winsopia disputed the validity of the purported termination and sought to affirm the agreements. By two letters each dated 29 July 2024, Winsopia purported to terminate the ICA and other licence agreements as from 31 August 2024. It follows that, although there is a dispute as to the mode and date of termination, it is common ground that the parties' primary obligations under the ICA have ceased. The parties confirmed in letters to the court that this does not affect any of the issues which fall to be decided at this stage in the hearing.

13. On 21 September 2021, IBM issued these proceedings, in which it seeks:

    i) a declaration that Winsopia's licence has been lawfully terminated;

    ii) an injunction restraining Winsopia from making any further use of the IBM licensed software, including from offering any services relying on the SDM that contains or uses any part of the IBM licensed software (and the other defendants from procuring the same); and

    iii) an account of profits and/or damages.

14. The defendants dispute the claims. Their case is that the SDM was developed by LzLabs following an extensive research and development process spanning almost ten years, using strict processes and policies which applied both to LzLabs and Winsopia, to ensure that no IBM material was used other than in compliance with the terms of the ICA and as permitted by the Software Directive. In developing the SDM, LzLabs employed a clean room process with a code of conduct and there was no unlawful use of the IBM licensed software.

15. The defendants counterclaim:

    i) injunctive / declaratory relief that Winsopia was not in breach of the ICA; and

    ii) damages for breach of the ICA.

16. This part of the trial is limited to issues of liability, including injunctive and declaratory relief. The parties have agreed that the precise form of any order and other consequential matters, including future disposal of the case, should be determined following this judgment.

Section II - Background to the dispute

IBM Mainframes

17. In 1964 the IBM group developed its first mainframe computer, System/360. Mainframes are high performance computers with large amounts of memory

and data processors that can process billions of calculations and transactions in real time. The following is a brief overview of IBM mainframe systems for the purpose of background to the issues in dispute.

18. The main processor in a computer is known as the central processing unit ("CPU"). A CPU is an electronic circuit that operates by executing a sequence of machine instructions stored in the computer's memory. Each instruction requires the CPU to execute a relatively simple computation. The complexity arises from the volume, speed, sequence, permutations and combinations of computations executed.

19. CPUs have registers, memory areas dedicated to each CPU where it can temporarily store information. Every IBM z CPU has 16 general purpose registers, that assembler programs can use or can be used for code generated by compilers, to store working data and provide input to CPU instructions.

20. The CPUs in IBM mainframe computers are based on z/Architecture. A feature of the mainframe is that different parts of the system can access and process data at the same time, so that if one part of the system fails, the data can still be accessed and processed without disruption, referred to as near zero downtime. Mainframes are highly secure in that customer data stored within the mainframe system can be encrypted and subject to a high degree of protection from malicious actors and inadvertent occurrences which might compromise the data. Furthermore, they are scalable in that mainframe users can add processors, memory and storage capacity without disrupting performance of the mainframe; and they have a high level of continuing compatibility, ensuring that technology support remains available for older applications.

21. Mainframes require an operating system to provide an environment in which application programs can run. The operating system is software that manages the computer system's physical resources, such as memory, processors, devices and file systems. The operating system most widely used on the mainframe is z/OS, software built on the IBM base control program, composed of many individual components which form layers of capability upon which mainframe customers build their application programs. The z/OS system can only run on IBM z mainframes, or software emulating mainframe hardware.

22. A machine code program is a sequence of instructions, in binary (ones and zeros) or hexadecimal format (numbers represented by 0 to 9 and A to F using base 16), ready to be processed by a CPU. Although it is possible to write all application programs as machine code programs, most programs are written in high-level language source code or assembly code.

23. Source code consists of a set of statements or instructions which tell the computer which mathematical operations to carry out so as to yield the desired outputs in response to specified inputs. It is human-readable code written in a programming language, consisting of numbers, alphanumeric characters and punctuation characters, such as COBOL, PL/I, Java, Python and C.

24. A CPU can only execute machine code programs; it cannot process high-level language source code. Therefore, source code must be compiled into object code

in binary or hexadecimal form before it can be executed. The compiler is a software program that reads source code from an input file, translates the statements into optimal machine code instructions and writes them to an output file. Additional to these functions, the compiler calls any runtime library services invoked by an appropriate command in the source code and generates additional executable code that is required to initialise the program for execution.

25. Assembler language is a high-level language that is used when a programmer needs more precision or control over the program's interaction with the mainframe hardware. Although it can be written and modified by a human programmer, it uses mnemonics and is closer in form to object code. Each machine instruction is described in text format by an "opcode", determining the operation that the instruction should perform, and a series of "operands", specifying the data values and/or memory locations with which the instruction should work.

26. Assembly code and machine code are closely related in that each assembler instruction maps directly on to a corresponding machine instruction. Therefore, converting from assembly code to machine code is a straightforward process of assembly; an assembly code program is assembled into object code by an assembler.

27. The runtime environment is a layer of capability which operates within an operating system and enables programs and applications to be executed. It consists of the runtime services required to implement the functions specified by the programmer in the source code of a program during the execution of that program. Runtime services are called upon by a program to facilitate the execution of particular functions. Collections of runtime services are referred to as runtime libraries. The Language Environment is a common runtime environment for applications written in a variety of programming languages for execution in z/OS, including COBOL, C and PL/I.

28. Application programs often are contained in a number of different high-level language files and require additional services provided by runtime libraries. Therefore, following compilation or assembly, a binder or linkage editor must bind or link the various object modules which make up the object code, together with existing object code files from the runtime libraries, to form an executable load module (or program object). Load modules can be loaded into the main storage of the computer, allowing the program to run.

29. 'Middleware' products sit between the services provided by an operating system and the applications, producing application programming interfaces ("APIs"). An API is a functional interface supplied by the operating system (or by a licensed program) that allows an application program written in a high-level language to use specific data or functions of the operating system or licensed program. APIs enable multiple transactions to be processed at the same time whilst preserving the integrity and consistency of the data. IBM CICS Transaction Server ("CICS") and IBM Information Management System ("IMS") are middleware software products within z/OS. CICS is a general-purpose transaction processing system for online applications, often used for

small but high-volume tasks, such as banking transactions. IMS is both a general-purpose transaction processing system and a hierarchical database management system for online applications.

30. CICS applications access their data in files, such as virtual storage access method ("VSAM") managed files, using the "EXEC CICS FILE" family of API commands. Application programmers write an EXEC CICS statement into the source code. Such statements must be pre-processed by the CICS translator before the relevant COBOL (or PL/I) compiler can compile them. The CICS translator replaces each EXEC CICS statement in the application source code with a sequence of language specific source code statements which it generates. The generated sequences invoke (by call statements) the required function in the CICS runtime. The detail of each generated sequence is determined using Language Definition Tables, an IBM proprietary set of algorithms and mappings contained within the CICS translator.

31. CICS provides functions to facilitate the passing of data between different regions of CICS efficiently. Distributed Program Link ("DPL") enables an application running in one region of CICS to communicate with an application running in a different CICS region. Distributed Transaction Processing ("DTP") enables a CICS transaction to communicate with a different transaction running in another non-CICS system.

32. The CICS translator can also be used for IMS services. In such cases, the programmer invokes IMS APIs and IMS run times by inserting "EXEC DLI" statements into the source code. The IMS database ("IMS DB"), is accessed by IMS programs through an IMS API, known as "IMS DL/I API". The IMS transaction monitor provides transactional capabilities which are similar to the transaction capabilities provided by CICS.

33. The ability to store and access information in a database is essential for commercial database processing. Database managers perform many tasks, most importantly providing APIs to allow applications to access, store and share data, to allow administrators to manage workloads and data, and to ensure resilience of the database even if an application program crashes in the middle of a transaction. IMS is a hierarchical database system, storing data in different categories of files, which supports transaction processing. IBM Db2 is an alternative relational database manager which runs on z/OS. Db2 uses tables (catalogs), containing columns and rows, to store, manage and retrieve data. It also uses catalogs to store information about the data stored in Db2.

34. Programs can be executed in real-time or as batch jobs. Job Entry System ("JES2") is a z/OS system used to provide an execution environment and manage batch jobs. z/OS batch jobs are created using a z/OS scripting language, Job Control Language ("JCL"). JCL statements can be used to specify how and when a job will run, the name of the job, programs that will be executed, and data files that the job may use.

35. Since the 1960s, the IBM group has continued to research, develop and market its product worldwide and it is now the only manufacturer and supplier of new mainframes.

The SDM

36. The intended purpose of the SDM is to allow IBM mainframe customers to run their existing customer applications, written for a mainframe, without mainframe hardware or software, thereby achieving cost savings. The functionality provided by IBM hardware running the z/OS operating system is instead defined and provided by alternative software. The SDM comprises a number of programs which can run on conventional x86 hardware (used by most laptops, PCs and servers) using Linux or other open-source operating systems and open-source database products. The aim of the SDM is to migrate existing applications which have been written to run on a mainframe and enable such programs to be run on the x86 runtime environment without recompilation.

37. LzLabs describes the SDM as a "compatibility layer", that allows an application that has been compiled and link edited to run on a mainframe to function in the same way in an x86 runtime environment.

38. There are three key difficulties in designing an alternative computer environment that can execute an application written and compiled for a mainframe. First, the application has been written and compiled for z/Architecture machine code rather than x86 machine code. Second, the application depends on services from the z/OS, COBOL and other IBM runtime environments. In order to call a library or language function from such runtime environments, it is necessary to identify: (a) the name of the function; (b) the number of inputs that the function takes; (c) the form that each input should take; (d) the number of outputs that the function produces; and (e) the form of each output. Third, customer data is encoded in an IBM mainframe specific data format and is not accessible in a readable format in another environment.

39. The SDM claims to enable an application written and compiled for an IBM z series mainframe to run in an alternative computing environment without having to recompile the same. First, it translates the z/Architecture machine code instructions for the application into a format that the x86 processor can understand. Second, it provides a compatibility layer that replaces calls made by the application to z/OS services and redirects those calls to replacement services which follow the same interface format as the interfaces expected by the application. Third, it migrates the customer's database from its original data format on the mainframe to an alternative, open source database format that can be used by the application in the new computing environment.

Hercules

40. In 1998, Jan Jaeger, now the Chief Technology Officer and Chief Software Architect at LzLabs, with his colleague, Roger Bowler, developed "Hercules". Hercules was a mainframe emulator, which supported, on an x86 CPU, functions equivalent to those provided by IBM mainframe hardware. The purpose of Hercules was to enable programs designed to run on the mainframe to be executed on a personal computer with an x86 processor by executing z/Architecture instructions, and implementing that instruction set on an x86 machine. Although a significant achievement, Hercules could only emulate the mainframe hardware; it did not replace the mainframe runtime environment.

Consequentially, it could not be used to run a mainframe application directly; it was necessary to install on Hercules an operating system, such as Linux for zSeries, to enable it to run such an application.

41. The development of Hercules by Mr Bowler and Mr Jaeger was achieved by implementing x86 versions of each of the machine instructions executed by mainframe CPUs. This task was facilitated, in part, using documented instructions, and in part, using reverse engineering for undocumented instructions, as explained by Mr Jaeger in cross-examination:

> "Q. Mr Jaeger, a fair description of the work you carried out was reverse-engineering on the mainframe, wasn't it?
>
> A. I -- well, you can call it reverse-engineering, fine. I don't know what you mean exactly by that when it was most -- we were not interested in how it worked, it was just – it's just like what we've done with the SDM. We want to know interfaces. A single instruction is also an interface. We just needed to know the interface.
>
> Q. Mr Jaeger, we're concerned here with undocumented instructions on the mainframe processor, aren't we?
>
> A. Yes.
>
> Q. Are you seriously saying that you consider those to be interfaces?
>
> A. They're interfaces into the hardware, absolutely.
>
> Q. I see. Which mainframe did you use to do that work?
>
> A. We had a mainframe at the ING Bank. It was running 14 z/OS, z/VM, Linux, it was running everything.
>
> Q. I see. And that was the mainframe that was used?
>
> A. Yeah."

42. Mr Jaeger's evidence was that some of the Hercules code was incorporated into the product that was developed into the SDM. He agreed that the SDM needed that compatibility layer with the processor instruction set to run the instructions on an x86 processor but denied that any reverse engineered instructions from Hercules were used in the SDM.

Neon litigation

43. Mr Moores is a software entrepreneur and the main investor in LzLabs, Winsopia and LzLabs UK. In 1980 he founded BMC Software, Inc. ("BMC") in Houston, Texas, a highly successful multinational company, developing and selling software products for IBM mainframe computers. In 1995 he founded

Peregrine Bridge Transfer Corp ("Peregrine Bridge"), the predecessor to Neon Enterprise Software LLC ("Neon").

44. In about 2009 Neon developed "zPrime", a utility that was designed to move discrete workloads off the mainframe central processor onto speciality processors. It was intended to disrupt the mainframe market by offering mainframe customers an alternative, less expensive means of processing legacy workloads. At the time, Mr Rockmann worked for a related company, Neon Enterprise Software Limited ("Neon Limited") and was involved in selling the zPrime product.

45. In December 2009, Neon commenced anti-trust proceedings against IBM Corp in the US District Court for the Western District of Texas ("the Neon litigation"). In its response to the Neon litigation, IBM Corp alleged that Neon reverse assembled z/OS, an IBM Program, to develop zPrime, in breach of its licence agreement with IBM Corp and that, by running zPrime, IBM customers unlawfully copied IBM's copyrighted program code without authorisation, thereby committing copyright infringement.

46. In 2011, following admissions by an employee of Neon that he had committed acts of reverse assembly and destroyed evidence, the Neon litigation was concluded by settlement. The settlement included agreed terms of a permanent injunction dated 31 May 2011, restraining Neon, Mr Moores and others ("the Enjoined Neon Persons") from directly or indirectly reverse assembling, reverse compiling or otherwise translating any IBM program or any portion thereof without the prior written consent of IBM Corp. The terms of the injunction included:

    i) a requirement for the Enjoined Neon Persons to deliver up the source, object and executable code and listings for each and every version or release of zPrime (and destroy other copies);

    ii) a prohibition on the Enjoined Neon Persons, directly or indirectly, from disclosing or transferring the source, object or executable code of zPrime to any other person;

    iii) a prohibition on the Enjoined Neon Persons, directly or indirectly, from developing, creating, modifying or using any software to enable workloads to be run on specialty engines (other than those expressly authorised in writing by IBM Corp);

    iv) a prohibition on the Enjoined Neon Persons, directly or indirectly, from disclosing or transferring any know-how relating to zPrime to any other person;

    v) a prohibition on the Enjoined Neon Persons, directly or indirectly, from reverse assembly, reverse compilation or other translation of any IBM Program (as defined in the ICA dated January 19, 2000, between Peregrine Bridge and IBM) or any portion thereof, except with the prior written consent of IBM Corp.

47. The terms of the injunction meant that operations at Neon and Neon Limited, including Mr Rockmann's employment at Neon Limited, came to an abrupt end.

Formation of LzLabs and Winsopia

48. In June 2011, within a few weeks after the conclusion of the Neon litigation, Mr Rockmann visited Mr Moores at his home in Pebble Beach, California, during which a new project, to develop what subsequently became the SDM, was conceived.

49. In August 2011 Mr Rockmann returned to the US and attended a meeting with software developers Eric Spencer, Andrew Galewsky and Mr Bowler, who discussed ideas for a proposal to enable mainframe customers to migrate workloads to a non-mainframe platform. Following that meeting, Mr Moores and Mr Rockmann agreed that Mr Rockmann would set up a company and assemble a team to develop the concept. It was agreed that Mr Moores would be the ultimate beneficial owner and provide the funding for the venture.

50. In furtherance of the proposed venture, in August 2011, Mr Moores incorporated Texas Wormhole LLC ("Texas Wormhole") for the purpose of developing what was then referred to as the "Rent Control" project.

51. On 3 November 2011 LzLabs was incorporated in Zürich, Switzerland. Mr Rockmann was appointed as Chairman and CEO of LzLabs in December 2011. In May 2015, he became the Chairman and Chief Operating Officer, and Mr Cresswell took over as CEO. In November 2020, he resumed his role as CEO.

52. A number of software developers or other employees who had worked at Neon and/or Peregrine Bridge were recruited to work on the Rent Control project at Texas Wormhole, LzLabs and, subsequently, Winsopia. They included Tom Harper (LzLabs), Mr Galewsky (LzLabs), Mr Spencer (Texas Wormhole and LzLabs), Bryan Young (Texas Wormhole and LzLabs), Paul Viebrock (Winsopia and LzLabs UK), Brad Taylor (Texas Wormhole and LzLabs) and Chris Palmer (Winsopia and Texas Wormhole).

53. In evidence Mr Jaeger stated that Mr Spencer was the main developer working on the prototype SDM, with the initial focus on developing a proof of concept. The early work involved testing using the NIST COBOL test suite, which is a set of COBOL testing programs that are available to download from the US National Institute of Standards and Technology website. This testing was carried out using a modified form of Hercules to emulate the mainframe hardware. Mr Jaeger's evidence was that not many of the NIST test programs executed successfully on the COBOL runtime environment in Linux; it was a process of trial and error.

54. In 2012, LzLabs engaged Texas Wormhole and OnTarget, an outsourcing company based in St Petersburg, Russia, which had previously worked with Neon, to provide software development services.

55. Once the LzLabs office opened in 2013, a number of the Texas Wormhole and OnTarget contractors moved to Switzerland and became employed by LzLabs.

56. Following consultation with the developers, Mr Rockmann formed the view that LzLabs would benefit from having a test facility available on an IBM mainframe, so as to allow the output and behaviour of applications on a mainframe to be compared to their output and behaviour on the LzLabs product. To achieve compatibility, LzLabs needed the output and behaviour to be the same in both environments.

57. Mr Rockmann decided to set up a mainframe testing environment using a separate company, ultimately named Winsopia. The intention was that LzLabs would acquire Winsopia; and Winsopia would acquire an IBM mainframe and license the z/OS operating system and subsystems, including Db2, IMS and CICS.

58. In early 2013, Keith Rastall was contacted by Ira Broussard, who was then working at Texas Wormhole, with a view to potential involvement in the project. Mr Rastall travelled to Kentucky to meet Mr Moores and his team to discuss setting up a company with a licensed mainframe to provide freelance testing services. He understood that Mr Moores and his team wanted to develop software which would allow applications that had been developed for a mainframe to be executed on non-mainframe platforms.

59. Mr Rastall was aware of the Neon litigation in the US. Although he did not know the details, he was aware that there had been an unfavourable outcome to the Neon litigation and that there was a strong possibility that IBM would start proceedings against Winsopia, sooner or later. Indeed, in an email to Mr Rockmann dated 19 February 2013, Mr Rastall stated:

> "I have one significant reservation and that is my liability and or my companies liability in the event that I be pursued in the courts for my involvement in the project. I would like some advice on this from Lzlabs lawyers? And if Liability insurance is required what insurance cover would be required and who would pay for it."

60. Notwithstanding that concern, Mr Rastall agreed to Mr Moores' proposal. On 15 March 2013, he incorporated Winsopia in the UK.

61. A few months later, Mr Rockmann approached Mr Rastall and told him that LzLabs wanted to acquire Winsopia because LzLabs required its services on an exclusive basis. Mr Rastall considered that he had no real option but to sell, as LzLabs was Winsopia's principal, indeed its only, customer.

62. On 5 July 2013 LzLabs acquired Winsopia and Winsopia became a wholly owned subsidiary of LzLabs. Mr Rockmann became sole director of Winsopia. Mr Rastall became the general manager of Winsopia and is now Vice President of Customer Success at Winsopia.

63. In August 2013 Winsopia purchased a second-hand IBM z10 mainframe from GMT360 Limited, through Mr Wilson at RSM Partners Ltd ("RSM"). By a contract dated 16 July 2013 between Winsopia and RSM, RSM agreed to

provide mainframe hosting and support services in respect of Winsopia's mainframe.

The ICA

64. Intellectual property rights in the IBM mainframe software products are owned by IBM Corp and licensed to IBM pursuant to a Principal Licence Agreement dated 1 April 1987 (as subsequently amended).

65. On 15 August 2013, Winsopia entered into the ICA (and associated agreements) with IBM. Pursuant to the ICA, the following software was licensed by IBM subject to the terms and conditions of the ICA:

    i) DB2 10 for z/OS, a relational database management system;

    ii) IMS V12 Database Manager, a hierarchical database management system;

    iii) IMS V12 Transaction Manager, a message-based transaction processor;

    iv) IBM COBOL V4, a development environment for COBOL high level programming language;

    v) IBM Enterprise PL/I for z/OS, a development environment for PL/I, a high level programming language;

    vi) CICS Transaction Server for z/OS, a mixed-language application server which can provide services such as security and exchanging data between new and existing applications;

    vii) z/OS V1 Base, providing basic operating system functionality;

    viii) z/OS V1 DFSMS dss, a direct access storage device data and space management tool;

    ix) z/OS V1 DFSORT, a program used to sort, merge, and copy information;

    x) z/OS V1 HLASM Toolkit, facilitating use of the IBM assembler;

    xi) z/OS V1 SDSF (System Display and Search Facility), a utility to monitor, control, and view the output of jobs in the system; and

    xii) z/OS V1 Security Server (Resource Access Control Facility), a mainframe security manager.

66. Subsequently, Winsopia licensed three further programs from IBM:

    i) MQ V8, a message-oriented middleware product used to send messages between applications;

    ii) IMS ETO (Extended Terminal Option) V12;

iii) RMF (Resource Management Facility) V2, a utility for measuring and reporting on performance and usage.

67. The recital to the ICA states:

> "This IBM Customer Agreement (called the "Agreement") governs transactions by which the Customer purchases Machines, licences ICA Programs, obtains Program Licences and acquires Services (including, without limitation, customised development and support, business consulting, and maintenance Services) from IBM United Kingdom Limited ("IBM")."

68. The scope of the licence is set out in clause 4.1:

> "4.1 When IBM accepts the Customer's order, IBM grants the Customer a non-exclusive licence to use the ICA Programs only within the Customer's Enterprise in the United Kingdom. ICA Programs are owned by International Business Machines Corporation, one of its subsidiaries, or a third party and are copyrighted and licenced (not sold).
>
> 4.1.1 Authorised Use
>
> Under each licence, IBM authorises the Customer to:
>
> a. use the ICA Program's machine-readable portion on only the Designated Machine. If the Designated Machine is inoperable, the Customer may use another machine temporarily. If the Designated Machine cannot assemble or compile the ICA Program, the Customer may assemble or compile the ICA Program on another machine. If the Customer changes a Designated Machine previously identified to IBM, the Customer agrees to notify IBM of the change and its effective date;
>
> b. use the ICA Program to the extent of authorisations the Customer has obtained;
>
> c. make and install copies of the ICA Program, to support the level of use authorised, provided the Customer reproduces the copyright notices and any other legends of ownership on each copy or partial copy; and
>
> d. use any portion of the ICA Program IBM provides i) in source form, or ii) marks restricted (for example "Restricted materials of IBM") only to: (1) resolve problems related to the use of the ICA Program, and

(2) modify the ICA Program so that it will work together with other projects.

4.1.2 The Customer's Additional Obligations

For each ICA Program, the Customer agrees to:

a. comply with any additional or different terms in its Licensed Program Specifications or an Attachment or Transaction Document;

b. ensure that anyone who uses it (accessed either locally or remotely) does so only for the Customer's authorised use and complies with IBM's terms regarding ICA Programs; and

c. maintain a record of all copies and provided to IBM at its request.

4.1.3 Actions The Customer May Not Take

The Customer agrees not to:

a. reverse assemble, reverse compile, otherwise translate, or reverse engineer the ICA Program unless expressly permitted by applicable law without the possibility of contractual waiver; or

b. sublicence, assign, rent, or lease the ICA Program or transfer it outside the Customer's Enterprise."

69. Relevant definitions were set out in clause 1.3 of the ICA, including:

i) Designated Machine:

"either i) the machine on which the Customer will use an ICA Program for processing and which IBM requires the Customer to identify to IBM by type/model and serial number, or ii) any machine on which the Customer uses the ICA Program if IBM does not require the Customer to provide this identification."

ii) Enterprise:

"any legal entity (such as a corporation) and the subsidiaries it owns by more than 50 percent. The term "Enterprise" applies only to the portion of the Enterprise located in the United Kingdom."

iii) ICA Program:

"an IBM Program licensed under Part 4 of this Agreement."

iv) Machine Code:

> "microcode, basic input/output system code (called "BIOS"), utility programs, device drivers, diagnostics, and any other code (all subject to any exclusions in the licence provided with it) delivered with an IBM Machine the purpose of enabling the Machine's function as stated in its Specifications …"

v) Materials:

> "literary works or other works of authorship (such as software programs and code, documentation, reports and similar works) that IBM may deliver to the Customer as part of a Service. The term "Materials" does not include Programs, Machine Code, or other items available under their own licence terms or agreements."

vi) Non-IBM Program:

> "a Program licensed under a separate third party licence agreement."

vii) Other IBM Program:

> "an IBM Program licensed under a separate IBM licence agreement (e.g., IBM International Program Licence Agreement)."

viii) Product:

> "a Machine or a Program"

ix) Program:

> "the following, including the original and all whole or partial copies:
>
> a. machine readable instructions and data;
>
> b. components;
>
> c. audio-visual content (such as images, text, recordings, or pictures); and
>
> d. related licenced materials.
>
> The term "Program" includes any ICA Program, Other IBM Program, or Non-IBM Program that IBM may provide to the customer. The term does not include Machine Code or Materials."

x) Specifications:

> "information specific to a Product … ICA Program Specifications are in a document entitled "Licensed Program Specifications"."

xi) Specified Operating Environment:

> "The machines and programs with which an ICA Program is designed to operate, as described in its Licensed Program Specifications."

70. Provisions regarding IBM's right to conduct an audit are contained in clause 4.4:

> "4.4.1 IBM's right to verify the Customer's usage data and other information affecting the calculation of charges also includes the right to verify the Customer's compliance with other terms of this Agreement (including applicable Attachments and Transaction Documents) relating to the Customer's use of ICA Programs at all sites and for all environments in which the Customer installs or uses ICA Programs for any purpose. IBM may use an independent auditor to assist with such verification, provided IBM has a written confidentiality agreement in place with such auditor.
>
> 4.4.2 The Customer agrees to create, retain, and provide to IBM and its auditors written records, system tools outputs, and other system information sufficient to provide auditable verification that the Customer's installation and use of ICA Programs complies with the Agreement terms, including IBM's applicable licensing and pricing terms. IBM will notify the Customer in writing if any such verification indicates that the Customer is not in compliance with Agreement terms. The rights and obligations in this section remain in effect during the period any ICA Programs are licenced to the Customer and for two years thereafter."

71. Clause 1.11.4 makes the following provision for dispute resolution:

> "Each party will allow the other reasonable opportunity to comply before it claims that the other has not met its obligations under this Agreement. The parties will attempt in good faith to resolve all disputes, disagreements, or claims between the parties relating to this Agreement. Unless otherwise required by applicable law without the possibility of contractual waiver or limitation, i) neither party will bring a legal action, regardless of form, arising out of or related to this Agreement or any transaction under it more than two years after the cause of action arose; and ii) after such time limit, any legal action arising out of

> this Agreement or any transaction under it and all respective rights related to any such action lapse."

72. Clause 1.11.5 makes the following provision regarding confidential information:

> "The exchange of any confidential information will be made under a separate, signed confidentiality agreement. However, to the extent confidential information is exchanged in connection with any Product or Service under this Agreement, the applicable confidentiality agreement is incorporated into, and subject to, this Agreement."

73. Clause 1.12 contains the following provisions regarding termination:

> "1.12.1 Either party may terminate this Agreement on written notice to the other following the expiration or termination of the terminating party's obligations under this Agreement, including any applicable Attachment or Transaction Document.
>
> 1.12.2 Either party may terminate this Agreement if the other does not comply with any of its terms, provided the one who is not complying is given written notice and reasonable time to comply. Licence termination and termination of a Services transaction are described in Parts 4 and 5, respectively.
>
> 1.12.3 Any terms of this Agreement that by their nature extend beyond the Agreement termination remain in effect until fulfilled, and apply to both parties' respective successors and assignees."

74. Licence Termination is dealt with in clause 4.5:

> "4.5.1 The Customer may terminate the licence for an ICA Program at any time on one month's written notice to IBM.
>
> …
>
> 4.5.3 IBM may terminate the Customer's licence if the Customer fails to comply with the licence terms. If IBM does so, the Customer's authorisation to use the ICA Program is also terminated."

75. The ICA contains an entire agreement provision at clause 1.4.5:

> "… This Agreement, including its applicable Attachments and Transaction Documents, is the complete agreement regarding transactions by which the Customer purchases Machines, licences ICA Programs, obtains Program licences, and acquires

> Services from IBM, and replaces any prior oral or written communications between the Customer and IBM. In entering into this Agreement, including each Attachment and Transaction Document, neither party is relying on any representation that is not specified in this Agreement … Additional or different terms in any written communication from the Customer (such as a purchase order) are void."

SDM development and the clean room procedures

76. In November 2013 the Winsopia mainframe was moved from RSM to Winsopia's Farnborough office.

77. An early idea, conceived by Mr Galewsky, then at Texas Wormhole, was to develop "the Appliance", a non-mainframe machine comprising both hardware and software, to which some batch processing would be offloaded from a customer's mainframe. A version of the Linux-based COBOL runtime environment could then be used to run the batches on the Appliance. The project, and the software that ran on the appliance, was referred to as "Rent Control".

78. As part of the development process, a version of the Appliance was installed at Winsopia's premises. The Appliance operated under the control of mainframe software that became known as "the Agent", which was written by Mr Palmer at Winsopia. The Agent was a z/OS program that was intended to run on the customer's mainframe and which would intercept batch jobs from the customer application and redirect eligible job steps to the Appliance via a connection. The intention was that the Appliance would be configured to receive batch jobs in the Linux environment, execute them in LzLabs' software runtime environment, and then return the results to the mainframe for use in the customer application.

79. On 4 December 2013, Winsopia and LzLabs entered into a consultancy services agreement ("the Services Agreement"), whereby Winsopia agreed to perform discovery, quality assurance and software development work for LzLabs. The services were described in Schedule 1 of the agreement as: (i) discovery and QA services to LzLabs as described in the Winsopia Code of Conduct; (ii) software development services to LzLabs in respect of the Agents; and (iii) managerial and operational support and cooperation as necessary, including attendance at meetings and conference calls.

80. Mr Rastall's evidence was that Winsopia used the mainframe and IBM licensed software for the sole purpose of assisting LzLabs in its software development. The main services provided by Winsopia comprised work on the Agent program, from 2014 until 2015, when the Agent and Rent Control Appliance project was abandoned; discovery work by Winsopia engineers through the DR process to assist LzLabs in developing the SDM; and testing work. This included testing existing compiled mainframe applications to see if they operated successfully in their native environment, and creating and compiling test applications on the mainframe so that their output could be compared to the behaviour of those applications on early versions of the SDM.

81. The defendants set up clean room procedures to ensure compliance with the terms of the ICA during development of what became the SDM. The clean room procedures included: (i) codes of conduct to be followed by Winsopia and LzLabs; and (ii) the Discovery Request ("DR") system.

82. The intention behind the codes of conduct was to impose:

    i) restrictions on the use of ICA Programs by anyone outside Winsopia's Enterprise;

    ii) formal processes for the LzLabs developers to request Winsopia to provide information through the DR system, including compilation and execution of test programs, provision of redacted information and communications between LzLabs and Winsopia;

    iii) restrictions on which publicly available sources of information could be accessed and used by the LzLabs developers;

    iv) restrictions on methods of communication between Winsopia staff and LzLabs, including those who were permitted to visit the Winsopia premises, physical security measures during the visits and records of such visits; and

    v) training of personnel and external legal oversight.

83. The DR System was designed to minimise the risk of infringing contractual and intellectual property rights of IBM or third parties, by preventing LzLabs developers from seeing and copying any IBM or third-party material they were not entitled to see or copy. Mr Rockmann described the operation of the DR System as follows:

    i) A developer at LzLabs would lodge a request for information ("a DR request"). Typically, LzLabs developers requested information in the form of test results in respect of test programs designed by LzLabs and/or customer applications run with specific parameters.

    ii) A team lead at LzLabs would approve the DR request.

    iii) The DR request would be forwarded, together with any attachments, for review by the review team (which, for most but not all of the relevant period, included external lawyers).

    iv) The review team would review the DR request for compliance with the Codes of Conduct.

    v) The DR request would then be passed on to a manager at Winsopia, who would review the request and allocate it to the Winsopia employee who was most appropriate or available to action the DR request.

    vi) The Winsopia employee would complete the DR request by carrying out the relevant testing using test or customer applications on the IBM mainframe.

vii) The Winsopia employee would complete the testing and send the response with attachments to the Winsopia manager. Winsopia would scrub binary attachments and redact any other material that was not permitted to be shared with LzLabs developers.

viii) The Winsopia manager would review the Winsopia Response and forward it to the review team.

ix) The review team would, either forward the Winsopia Response to the team lead at LzLabs, or return it to the manager at Winsopia with comments, where the response was incomplete or there was any issue with its content.

x) The team lead at LzLabs would forward the Winsopia response to the developer that made the DR request.

84. There were a number of different versions of the Codes of Conduct throughout the development of the SDM.

85. The first version of the LzLabs Developer Code of Conduct was dated 4 November 2013. It applied to LzLabs Developers, who were listed in Schedule 1, including Mr Galewsky, Mr Bowler, Mr Jaeger, Mr Bond and Mr Taylor. The introduction explained the context for the LzLabs Developer Code of Conduct as the development of what became the SDM:

> "2.1 LzLabs GmbH ("LzLabs") will develop the RentControl Appliance (the "Appliance"). The Appliance will be a binary-compatible drop-in replacement for certain functions of an IBM z Series mainframe running z/OS. To develop the Appliance, it is necessary to perform discovery work on a z/OS system in order to determine how the Appliance should behave."

86. The original Winsopia Employee Code of Conduct was dated 19 November 2013. It applied to all Winsopia Employees, including employees, managers and consultants providing services to Winsopia. It also applied to "Winsopia Researchers", that is, all people employed by Winsopia as developers, researchers or managers of developers and/or researchers, and all consultants carrying out such work for Winsopia. It provided that LzLabs and Texas Wormhole would each develop aspects of the Appliance; Winsopia would carry out discovery work on the IBM z/OS system mainframe, together with QA and development work pursuant to the Services Agreement entered into by Winsopia and LzLabs.

87. The material terms of the LzLabs Developer Code of Conduct and the Winsopia Employee Code of Conduct (together, "the Codes of Conduct") were the same.

88. The purpose of the Codes of Conduct was stated to be to set out rules which Winsopia and LzLabs were required to follow in order to respect third party intellectual property rights, even if they resulted in inefficient working. Both Codes of Conduct contained a warning that failure to follow the code might

expose LzLabs, or Winsopia, to legal risk and potential injunctions preventing sale of the developed software.

89. The Codes of Conduct provided that development of the Appliance would be carried out using a strict controlled access environment process. Winsopia had acquired a mainframe and licences for z/OS components, which were listed in Schedules to each code. The controlled access environment depended on separation of the development work at LzLabs and Texas Wormhole from the discovery and testing work at Winsopia. To that end, only Winsopia would have access to a version of the Appliance and the mainframe. Winsopia would carry out tests to be run on the Appliance, tests to be run on the mainframe, and tests to be initiated on the mainframe but intercepted by the Agent (developed by Mr Palmer at Winsopia) and diverted to the Appliance for processing.

90. There were restrictions on communications between Winsopia and LzLabs/Texas Wormhole. These included the following.

    i) It was mandatory for all communications between Winsopia and LzLabs Developers or Texas Wormhole Developers to be made via the Jira Systems, subject to express narrow exceptions (primarily to allow Mr Rastall and Mr Galewsky to communicate with Winsopia and LzLabs employees in furtherance of their managing roles).

    ii) Winsopia and LzLabs/Texas Wormhole Developers were prohibited from meeting face to face, or knowingly being in the same building at the same time. Indeed, Winsopia employees were not permitted to visit LzLabs offices in Switzerland or Texas; and LzLabs employees were not permitted to visit Winsopia's office in Farnborough in the UK.

    iii) Winsopia and LzLabs/Texas Wormhole Developers were prohibited from speaking to each other by telephone, Skype and videoconferencing systems, or by email instant messaging, file transfer, or sharing service.

    iv) They were prohibited from sending to or receiving from the other any physical object (including but not limited to books, documents and portable storage media).

    v) They were prohibited from carrying out their work on any computer, tablet, smartphone or other comparable device other than computers and devices issued by their employers.

    vi) LzLabs Developers were prohibited from accessing or attempting to access any computer system owned or controlled by Winsopia, including the Appliance; and Winsopia was prohibited from accessing or attempting to access any computer system owned or controlled by LzLabs or Texas Wormhole.

91. There was a procedure for dealing with customer programs and/or data that would be supplied to LzLabs or Texas Wormhole. The customer would supply the data to LzLabs on a removable storage disk (such as a USB stick) and LzLabs would deliver it to Winsopia. Winsopia would remove from the data

and/or replace with LzLabs equivalents all material subject to a third party's copyright, such as z/OS Language Environment CSECTs in COBOL load modules and/or IBM CSECTs. The cleaned data would be copied by Winsopia to a directory on its FTP server accessible by LzLabs Developers.

92. Compiler listings requested through the DR system would be redacted by the legal team to remove comments and other non-essential material inserted by the compiler before being sent to LzLabs or Texas Wormhole. One hard copy of the unredacted compiler listing could be sent to the individual who had submitted the request.

93. Winsopia and LzLabs were required to keep a record of the sources referred to during their work. In particular, the Codes of Conduct provided that Claire Wilmot of Winsopia would organise and maintain records of all visitors to Winsopia and a log of unredacted hard copy compiler listings sent by Winsopia to LzLabs or Texas Wormhole Developers.

94. By March 2014 it is clear that Mr Moores was unhappy with the rate of progress achieved by LzLabs and considered that the Codes of Conduct were hindering development, as set out in his email dated 25 March 2014. Mr Moores considered that maintaining the Chinese Wall processes was a waste of time and money. He wanted the LzLabs Developers to have direct access to a mainframe and to unredacted listings, including assembly listings, traces and other information obtained through analysis of programs on the mainframe.

95. In April 2014 the Codes of Conduct were revised to remove references to Texas Wormhole and to relax some of the restrictions imposed on Winsopia and LzLabs. Under the revised Codes of Conduct, developers at LzLabs were able to access a test instance of the Appliance, which was housed at Winsopia. The purpose of such access was to test and update the developing LzMatrix software on the Appliance. This was done through a remote network connection called "the Airlock". The purpose of the Airlock was firstly, to ensure that LzLabs developers could only access the Airlock for the purpose of testing the Appliance, without being able to access the wider Winsopia network; secondly, to provide a reliable mechanism for recording everything that those LzLabs developers did.

96. It was determined that communication between LzLabs Developers and Winsopia would be required to develop the Agent and ensure that it communicated correctly with the Appliance. For that purpose, a number of exceptions to the strict communications rules were introduced. To be permitted access to the Airlock, LzLabs Developers had to be located outside the USA and have an absolute requirement to test network communications with the mainframe in order to perform the assigned tasks.

97. Further, it was decided that it might be necessary for LzLabs Developers to visit Winsopia and interact with Winsopia system administrators in order to troubleshoot problems with the Airlock or the Appliance. This was required to be subject to legal oversight by Daniel Hedley or other legal advisers in attendance and the LzLabs Developers were not permitted to log in to or directly access the mainframe.

98. The existing procedure for redaction of electronic compiler listings was revised to allow Winsopia to provide LzLabs with compiler listings or traces of network traffic (such as IP packet traces or GTF traces), subject to redaction of English-language strings of three words or more.

99. Additionally, on 28 May 2014 further amendments to the Codes of Conduct were made, whereby communication was permitted by instant messaging outside the DR system and some of the LzLabs Developers were re-designated as LzLabs Support Engineers, named as Ulrich Weibel and John Cassidy. Mr Weibel and Mr Cassidy would agree with customers the means by which they would deliver their programs or data to Winsopia (by email or upload to Winsopia's FTP server). As set out in earlier versions of the Codes of Conduct, Winsopia would remove from the data and/or replace with LzLabs equivalents all material subject to a third party's copyright, such as z/OS Language Environment CSECTs in COBOL load modules and/or IBM CSECTs. The cleaned data would be copied by Winsopia to a directory on its FTP server accessible by LzLabs Developers.

100. On 22 August 2014, a new LzLabs Support Team Code of Conduct was introduced. This new code applied to LzLabs Support Engineers, who were defined as employees of LzLabs who: (i) worked in customer support, field service or QA; (ii) did not meet the definition of LzLabs Developer in the LzLabs Code of Conduct; and (iii) did not write code which would form part of the Appliance.

101. The LzLabs Support Engineers were not subject to the Codes of Conduct but the overriding principle identified in their dealings with customers was that they must not cause the customer to breach its licence agreements with IBM. Examples of potential breach included: (i) directly using the customer's mainframe when the customer did not have the right to permit such use; (ii) taking copies of IBM Licensed Material in source code form, such as IBM copybooks; and (iii) attempting to reverse engineer aspects of the customer's mainframe.

102. The overriding principle identified in dealings between the LzLabs Support Engineers and LzLabs Developers was that they must not enable, assist or cause any LzLabs Developer to breach the LzLabs Developer Code of Conduct. Examples of potential breach included: (i) involving LzLabs Developers in troubleshooting a customer problem where debugging of LzLabs code was not necessary to resolve the problem; (ii) providing excessive or unnecessary debug data from customer mainframes to LzLabs Developers; (iii) providing LzLabs Developers with access to the raw encrypted diagnostic feed from Winsopia's Appliance; and (iv) holding conference calls with both LzLabs Developers and Winsopia in attendance. Similar guidance was provided in respect of dealings with Winsopia and Texas Wormhole.

103. In August and September 2014 the Codes of Conduct were revised, permitting Winsopia employees to communicate directly with LzLabs Support Engineers, by email or face-to-face, where necessary. Changes included the addition of a secure, monitored instant messaging system ("the IM System") to enable LzLabs Developers with access to the Airlock to communicate with Winsopia

in respect of integration and functional tests. In particular, provision was made for LzLabs Developers to be seconded to Winsopia on a temporary basis, pursuant to a Supply of Staff Agreement dated 14 August 2014.

104. Dissatisfaction with the restrictions imposed by the Codes of Conduct continued to be raised. By email dated 28 May 2015, Chris Cole forwarded to Mr Moores an email chain with Mr Rockmann and LzLabs' lawyers, that appeared to show that legal issues were causing a delay with a DR response. In response, Mr Moores stated that there needed to be further changes to the Code of Conduct because development progress was being substantially impacted. He was clear that continuing the existing clean room system would not work.

105. Mr Moores was vociferous in his calls for LzLabs developers to be given direct access to the mainframe. Although in evidence he stated that he did not put pressure on the legal team to make any specific changes to the Codes of Conduct, very shortly after this exchange, further revisions to Codes of Conduct were made.

106. On 10 June 2015 the LzMatrix Code of Conduct for LzLabs Employees and Winsopia Employees was published, replacing the earlier Codes of Conduct ("the Integrated Code of Conduct"). This introduced a significant change, namely, that under the Integrated Code of Conduct, access to the mainframe would be granted to LzLabs:

> "4.1 Winsopia, based in Farnborough, England, has acquired a z Series mainframe, DASD devices and licences for various z/OS components (each, a "Component" and together, the "Mainframe"). Save as otherwise provided in this code of conduct, only Winsopia Employees will have access to the Mainframe.
>
> 4.2 LzLabs Developers may now additionally have access to the Mainframe provided that (i) the visit and its purposes (to be set out in writing when seeking approval) are pre-approved by (a) Winsopia and (b) a member of the LzLabs executive team; (ii) that access occurs at Winsopia's premises; and (iii) is solely for the purposes of assisting Winsopia Employees in testing (including debugging) already developed code.
>
> 4.3 LzLabs Support Staff may access the Mainframe provided that they do so at Winsopia's premises for the purposes of supporting Winsopia's operations and activities.
>
> 4.4 All visits to Winsopia by LzLabs Employees will be logged."

107. On 23 September 2015 LzLabs UK was incorporated as a wholly-owned subsidiary of LzLabs, to provide a UK base for LzLabs and to carry out QA services. Mr Cresswell and Mr Rockmann were appointed as directors.

108. On 13 October 2015 the Integrated Code of Conduct was revised to include LzLabs UK employees. Under this revised code, LzLabs UK employees were

not permitted to carry out discovery work on the mainframe for use in developing LzLabs code. However, under this revised Code, additional provision was made for permitted direct communications between LzLabs Developers and Winsopia Employees at clause 6.1:

> "In addition to performing discovery and QA work, Winsopia is developing a tool for recompiling load modules into x86 native code, as well as certain other mainframe- based tools, utilities and agents to facilitate customer migration (together, the "Winsopia Development Projects") using the Mainframe. It is acknowledged that communication between relevant LzLabs Developers and Winsopia Employees is required in order to develop the Winsopia Development Projects and to ensure that they work correctly with the Product. However, please note that development of the Winsopia Development Projects is subject to the same communications restrictions applicable to all interactions between LzLabs Employees and Winsopia Employees …"

109. Further amendments permitted LzLabs, LzLabs UK and Winsopia employees to communicate by instant messaging, support systems, email, Skype, videoconference, phone or face-to-face, although they were required to use their company systems and record all such communications.

110. Thus, the evolution of the Codes of Conduct, following their introduction in late 2013 can be summarised as follows:

i) By April 2014, Winsopia employees were afforded greater latitude to engage in communication with LzLabs developers; LzLabs developers were given indirect access to the Winsopia mainframe; and Winsopia was permitted to send to LzLabs developers network traces and other listings, which were subject to relaxed redaction rules.

ii) By May 2014, communication was permitted by instant messaging outside the DR system and some of the LzLabs developers were re-designated as LzLabs support engineers, who were not bound by the restrictions that applied to developers under the Codes of Conduct.

iii) By September 2014, those designated as LzLabs support engineers were afforded greater freedom to interact with both Winsopia employees and LzLabs developers and provision was made for LzLabs developers to be seconded to Winsopia on a temporary basis.

iv) By June 2015, LzLabs developers were given direct access to the Winsopia mainframe.

Launch of the SDM

111. The SDM was officially launched in March 2016 at the CeBIT IT fair in Hannover, Germany. The key SDM components can be summarised as follows.

112. "The SDM COBOL runtime" provides a compatibility layer which enables customers to run end user applications compiled with and reliant upon the IBM COBOL runtime environment for z/OS. The SDM PL/I runtime runs end user applications compiled with IBM Enterprise PL/I for z/OS. They depend on Language Environment for low level-language related operations, such as obtaining system time or I/O operations.

113. "The SDM LE runtime" provides a compatibility layer which enables customers to run applications which were compiled with and reliant upon the low-level language runtime environment. The LE runtime supports COBOL, PL/I and Assembler programs utilising Language Environment functions.

114. "LzOnline" is the transaction server component of the SDM, comprising a compatibility layer which enables customers to execute applications and data that rely on the IBM CICS runtime environment in alternative environments.

115. "LzHierarchical" provides support for hierarchical databases, comprising a compatibility layer which enables customers to execute applications and data that rely on the IBM IMS runtime environment in alternative environments. LzHierarchical uses an open-source RDBMS, PostgreSQL, to provide compatibility with the data types and functions uniquely found in IMS, enabling mainframe applications and data to be rehosted in the new environment.

116. "LzRelational" provides support for relational databases, comprising a compatibility layer which enables customers to execute applications and data that rely on DB2 in alternative environments. LzRelational uses PostgreSQL user-defined types to provide compatibility with the data types and functions uniquely found in DB2, enabling mainframe applications and data to be rehosted in the new environment.

117. "LzBatch" delivers binary-compatible execution for batch applications written as JCL jobs in PL/I, COBOL and Assembler. LzBatch is a compatibility layer which enables customers to execute batch applications designed to run on IBM mainframes in alternative environments. Batch applications can run, without recompilation or data reformatting, in a Linux environment based on x86 or ARM hardware. JCL support is provided, enabling batch jobs to be submitted locally, by Network Job Entry ("NJE"), or via FTP-connected mainframes.

118. "SDM Core" provides the basic functionality to run the above components, including configuration files, catalog data sets, configuration file parser, debugging functionality, data set I/O manager, JSON parser and the SDM loader.

119. The Centerpiece Export tool ("CPX") was developed by Winsopia to run on a customer's IBM mainframe. CPX removes stub CSECTS contained in the customer application that call external services provided by the z/OS system or its runtime environments and replaces them with temporary placeholders. The customer application is imported into the SDM by the Centerpiece Import tool ("CPI"). The compatibility layer of the SDM replaces the removed stub CSECTS with new CSECTS ("wormholes") that call replacement services in the alternative x86 Linux environment, using appropriate parameters, and

provides replacement functionality that produces the same outputs that would be produced by the mainframe runtime environment.

Project Eiger

120. In about April 2016, Mark Anzani, of IBM Corp, learned about the SDM and became suspicious as to the means by which it had been developed. He launched an investigation into LzLabs, known as "Project Eiger".

Further development of the SDM

121. On 30 October 2017 a further revision of the Integrated Code of Conduct was issued. The roles of the main companies were described as follows:

    i) LzLabs, based in Switzerland, was the main trading entity and the main hub of development activity. Most developers of the SDM were employed or engaged there, and those based at external contractors were ultimately managed from LzLabs;

    ii) LzLabs UK, based in the UK, was focused on QA and testing of SDM, as well as having technical sales, customer care and support functions;

    iii) Winsopia, based in the UK, was the mainframe test and development house, responsible for discovery work, testing of customer applications in the legacy mainframe environment and of the SDM's interoperability with the legacy mainframe environment, due diligence on customer applications and data, and development of the z/OS-based tools which accompany the SDM, such as CPX and LMD.

122. The project roles for the SDM were described as (i) Developers – people working for LzLabs or its contractors who designed, developed and wrote the code for the SDM, including QA working within the development team; (ii) pre-sales, post-sales and support; (iii) QA – people working for LzLab or LzLab UK responsible for internal testing and quality assurance, other than QA personnel working within the development team; (iv) sales, management and other non-technical roles; and (v) Winsopia – responsible for discovery, testing on or interacting with the mainframe, processing of materials from customers' mainframes, setting customer applications in a mainframe test environment, and development of z/OS-based tools.

123. Under this revised code, greater freedom was granted to LzLabs, Winsopia and LzLabs UK to communicate with each other, including visits to Winsopia's premises.

124. Mr Rastall agreed in cross-examination that there was no supervision of LzLabs employees who visited Winsopia and for several years all passes and badges issued by Ms Wilmot at Winsopia automatically included access to the whole office, including access to the mainframe room. Such visitors included Mr Bleach, Mr Viebrock and Mr Vitale of LzLabs. Although this did not give them direct access to the mainframe, it enabled them to talk directly with those at Winsopia who did have direct access and allowed them to "look over their

shoulders" as they used z/OS. There is no evidence that any notes of such discussions were taken, contrary to the requirements in the Codes of Conduct.

125. By March 2020, the development of the SDM was sufficiently advanced so that the clean room procedures were no longer required, as indicated in a message dated 24 March 2020 from Mr Moores to Mr Cresswell:

> "I had a nice discussion with Thilo about managing our legal expenses - including merging companies, eliminating the review process, etc.
>
> I will be surprised if Thilo hasn't had a long chat with Watson by now.
>
> It seems to me that what we needed to do to build SDM is now securely in the can. And Lz has a very nice record to prove perfect reverse engineering & development.
>
> I donno what role Clifford has going forward - but maybe a diminished role is feasible. Obviously you want to keep your terrific long standing relationship."

126. Shortly afterwards, on 20 April 2020 a further revised version of the Integrated Code of Conduct was issued. The descriptions of the main companies changed. Winsopia was the mainframe test and z/OS-based tools development house, handling discovery work, testing of customer applications in the legacy mainframe environment and of the SDM's interoperability with the legacy mainframe environment, due diligence on customer applications and data, and development of the z/OS-based tools which accompany the SDM, such as CPX and LMD. 'LZ' was stated to encompass all the other operations of LzLabs and its subsidiaries.

127. Under this final version of the Integrated Code of Conduct, there was further relaxation of communications between LzLabs and Winsopia, and the requirement for legal oversight was removed.

128. In mid-2020, the LzLabs UK office was closed and its employees were transferred to Winsopia.

Audit request and termination

129. By letter dated 3 December 2020 Mr Anzani, of IBM Corp, wrote to Winsopia, informing it that IBM would be conducting an audit of Winsopia's compliance with the terms of the ICA and seeking documentary information within thirty days after receipt of the letter.

130. Winsopia replied by letter dated 23 December 2020, disputing IBM Corp's entitlement to seek any information from Winsopia, given that the contractual relationship was between Winsopia and IBM.

131. By letter dated 12 January 2021 Mr Wallin of IBM wrote to Winsopia, alleging a breach of the ICA by Winsopia, repeating the demand for an audit and stating that IBM might terminate the ICA if Winsopia failed to comply.

132. Mr Rockmann of Winsopia replied by letter dated 8 February 2021, denying any breach of the ICA, disputing the audit demand as unreasonable and stating that IBM was not entitled to terminate the ICA.

133. By letter dated 24 February 2021, IBM notified Winsopia that it was terminating the ICA, on the grounds that Winsopia was in breach of clause 4.4 of the ICA in failing to comply with the valid audit demand and, further, in breach of clause 4.1 of the ICA.

134. By letter dated 1 March 2021, Winsopia sought to affirm the ICA. However, by two letters, each dated 29 July 2024, Winsopia purported to terminate the ICA and other licence agreements as from 31 August 2024.

135. As set out above, it is now common ground that the ICA has been terminated.

Section III - The proceedings

136. On 21 September 2021 IBM commenced these proceedings against the first to fifth defendants.

137. On 29 November 2023 Mr Moores was joined as the sixth defendant to the proceedings.

The Issues

138. The key issues can be summarised as follows:

i) on a proper construction of the ICA, what fell within the definition of an ICA Program, the rights conferred by the licence and/or the Licensed Program Specifications, and the acts that were restricted by clause 4;

ii) whether any of the acts alleged to be in breach of the ICA fell within the rights conferred by the Software Directive, implemented by sections 50B, 50BA and 50C of the CDPA; if so, whether they were acts which IBM could restrict or prohibit by contract;

iii) whether Winsopia was in breach of the ICA as alleged by IBM;

iv) in respect of any breach of the ICA by Winsopia, whether the first, third, fourth, fifth and/or sixth defendants procured such breach;

v) whether any of the defendants are liable for the tort of unlawful means conspiracy;

vi) whether IBM validly terminated the ICA, or whether its purported termination of the ICA and/or the issuing of these proceedings constituted repudiatory breach of the ICA;

vii) whether any of IBM's claims are time-barred pursuant to clause 1.11.4 of the ICA (contractual limitation);

viii) whether any of IBM's claims are time-barred pursuant to Section 2 and/or Section 5 of the Limitation Act 1980 (statutory limitation);

ix) whether any party is entitled to declaratory and/or injunctive relief.

The factual witnesses

139. IBM relied on evidence from the following factual witnesses, who produced witness statements and gave oral evidence:

i) Steve Wallin, executive product development leader at IBM;

ii) Ian Lyon, First Line Sales Manager of Systems business at IBM;

iii) Paul Knight, Sales Manager with responsibility for Winsopia;

iv) Emma Wright, IBM's Head of Legal, UK and Ireland;

v) Mark Anzani, Special Projects Executive at IBM Corp;

vi) Ian Mitchell, IBM Z Strategic Modernisation Leader at IBM.

140. The court heard oral evidence from the following factual witnesses who produced witness statements on behalf of the defendants:

i) Thilo Rockmann, fifth defendant, CEO of LzLabs, Director of Winsopia and Director of LzLabs UK;

ii) Jan Jaeger, Chief Technology Officer and Chief Software Architect at LzLabs;

iii) Christian Wehrli, former Vice President of Product Delivery at LzLabs;

iv) David Bond, Senior Software Developer at LzLabs;

v) Brad Taylor, Senior Software Developer at LzLabs;

vi) Christopher Palmer, Software Engineer at Winsopia;

vii) Keith Rastall, General Manager of Winsopia;

viii) John Moores, the sixth defendant, principal investor in the LzLabs Group and Winsopia;

ix) Mark Cresswell, fourth defendant, CEO and Director of LzLabs and former Director of Winsopia and LzLabs UK;

x) Gary Whittingham, Senior Software Engineer at Winsopia;

xi) Alan Playford, Systems Consultant at Winsopia;

xii) Thomas Grieve, Senior Mainframe Systems Engineer at Winsopia;

xiii) Kevin Lynch, Senior Mainframe Systems Engineer at Winsopia;

xiv) John Bray, Mainframe Systems Engineer at Winsopia.

141. The defendants also relied on witness statements produced by the following witnesses who were not required to give oral evidence (without any admissions as to the contents of their statements):

i) Daniel Hedley, solicitor who provides advice and legal services to LzLabs and Winsopia;

ii) Eric Spencer, Senior Software Developer at LzLabs;

iii) Bryan Young, Head of IT Infrastructure at LzLabs;

iv) Martin Bleach, Quality Assurance Engineer at LzLabs.

142. The court received reports and heard oral evidence from the following experts for IBM:

i) Michael Swanson, a former software designer for IBM mainframe systems and an IBM fellow with over thirty years' experience as a systems programmer, who produced his first report dated 9 January 2024, second report dated 20 March 2024 and third report dated 24 June 2024;

ii) Professor Jon Weissman, a Professor of Computer Science at the University of Minnesota, who produced his first report dated 6 January 2024, second report dated 19 March 2024 and third report dated 24 June 2024.

143. The court received reports and heard oral evidence from the following experts for the defendants:

i) Professor Alastair Donaldson, a Professor in Computing at Imperial College London, who produced his first report dated 27 February 2024, an addendum report dated 3 April 2024, second report dated 25 April 2024 and third report dated 30 May 2024;

ii) David Stephens, the Lead Systems Programmer and owner of Longpela Expertise, an Australian mainframe consultancy company, with over thirty years' experience working with IBM mainframes, who produced his first report dated 20 February 2024, a technical primer dated 20 February 2024 and second report dated 31 May 2024.

144. It is clear that each of the above individuals acquired a deep understanding of the technical issues in dispute and used his considerable intellectual understanding and expertise to assist the court. In particular, the court had the benefit of very clear and helpful joint statements from the experts, setting out their agreements and individual opinions on matters on which they disagreed:

i) first Expert Joint Statement dated 15 December 2023;

ii) second Expert Joint Statement dated 5 April 2024;

iii) Joint Primer dated 10 April 2024;

iv) third Expert Joint Statement dated 19 April 2024.

145. I am very grateful to the experts for their careful consideration of the expert issues and their co-operation in producing materials to assist the court to determine the numerous technical issues in this case.

146. Likewise, I express my thanks to counsel on both sides for their clear and percipient cross-examination and submissions, and for their co-operation in ensuring that the hearing was conducted in a respectful and efficient manner.

Section IV - Construction of the ICA

147. IBM's case on the construction of the ICA is that:

i) the ICA provides a licence of limited scope, permitting Winsopia to use the licensed ICA Programs solely for its internal purposes; this does not extend to the provision of services using its mainframe and ICA Programs to LzLabs and the other defendants;

ii) the definition of ICA Programs is broad, comprising "machine readable instructions and data" as well as "components";

iii) Winsopia's use of an ICA Program obtained from a third party is subject to the restrictions in the ICA because it remains an IBM Program and, once licensed under the ICA, is an ICA Program;

iv) under clause 4.1.3(a), Winsopia was prohibited from reverse engineering ICA Programs;

v) under clause 4.1.3(b), Winsopia was prohibited from transferring ICA Programs out of its Enterprise;

vi) under clause 4.1.1(d), Winsopia could only use the source code of programs supplied in this form for certain narrow purposes; in particular, Winsopia was not permitted to copy and transfer copies of this source code to other parties;

vii) the Licensed Program Specifications relied on by the defendants do not operate to widen the scope of Winsopia's licence in light of clause 4.1.2(a); any right to distribute copies of CSECTs is limited to the distribution of copies of such CSECTs to other licensed users of ICA Programs and not LzLabs or the other defendants.

148. The defendants' case is that the alleged acts of reverse engineering were not prohibited by the effective terms of the ICA:

i) the contractual restrictions apply only to the twelve ICA Programs identified in the Agreement for Third Party Program Access (as updated) together with the additional three ICA Programs subsequently licensed; in particular, this does not include code fragments, sub-routines or individual programs as they do not constitute the product licensed;

ii) the contractual restrictions relied on do not apply to customer applications;

iii) the operative restrictions must be construed against the relevant factual matrix, including the practices of independent software vendors ("ISVs"), sharing and distribution of customer applications and modules, and use of debugging tools and techniques;

iv) on a true construction of the ICA, Winsopia was entitled to use the licensed software to perform the activities that are said by IBM to amount to breaches of contract;

v) further or alternatively, Winsopia was entitled to carry out the alleged acts of reverse engineering pursuant to rights guaranteed to licensees of computer programs by Articles 5(1), 5(3) and/or 6 of the Software Directive and the CDPA;

vi) the contractual restrictions are ineffective to the extent that they are contrary to the rights conferred by the Software Directive and the CDPA.

Approach to construction of the ICA

149. The principles applicable to contractual interpretation are well established and not in dispute. When interpreting a written contract, the court is concerned to ascertain the intention of the parties by reference to what a reasonable person, having all the background knowledge which would have been available to the parties, would have understood them to be using the language in the contract to mean. It does so by focussing on the meaning of the relevant words in their documentary, factual and commercial context. That meaning has to be assessed in the light of (i) the natural and ordinary meaning of the clause, (ii) any other relevant provisions of the contract, (iii) the overall purpose of the clause and the contract, (iv) the facts and circumstances known or assumed by the parties at the time that the document was executed, and (v) commercial common sense, but (vi) disregarding subjective evidence of any party's intentions: *Arnold v Britton* [2015] UKSC 36 per Lord Neuberger at [15]-[23]; *Wood v Capita Insurance Services Ltd* [2017] AC 1173 at [11]-[15]; *Rainy Sky SA v Kookmin Bank* [2011] UKSC 50 per Lord Clarke at [21]-[30]; *Chartbrook Ltd v Persimmon Homes Ltd* [2009] UKHL 38 per Lord Hoffmann at [14]-[15], [20]-[25].

Scope of licence

150. Clause 4.1 provides that:

> "… IBM grants the Customer a non-exclusive licence to use the ICA Programs only within the Customer's Enterprise in the United Kingdom…"

151. This entitles Winsopia, as the Customer, to use the ICA Programs identified in the licence documentation. There is no dispute as to the named ICA Programs falling within the licence and an agreed list was provided to the court with the parties' opening submissions.

152. The permitted use of the ICA Programs is limited to the Customer's Enterprise in the United Kingdom.

153. Enterprise is defined in clause 1.3 as:

> "any legal entity (such as a corporation) and the subsidiaries it owns by more than 50 percent. The term "Enterprise" applies only to the portion of the Enterprise located in the United Kingdom."

154. This amounts to both a geographical limitation, namely, within the United Kingdom, and a business limitation, namely, to Winsopia's business. IBM correctly notes that this restricts use within and for the purposes of Winsopia's organisation, thereby excluding use by external companies. IBM seeks to go further and submits that use of the ICA Programs is limited to Winsopia's internal purposes. I reject that submission because it stretches the words beyond their natural and obvious meaning. The nature and extent of Winsopia's business enterprise is not confined to any specific purpose or scope. There is no express prohibition on Winsopia using the ICA Programs to provide services to LzLabs and others, such as testing and quality assurance, provided that such use otherwise complies with the terms of the ICA.

155. Clause 4.1.1 sets out the authorised use of the ICA Programs. At 4.1.1(a) Winsopia is permitted to:

> "use the ICA Program's machine-readable portion on only the Designated Machine."

156. This is a physical restriction, not on the location of use but on the machine that can be used to run the ICA Program, namely, the Winsopia mainframe, regardless of where that mainframe is sited.

157. Clause 4.1.1(b) permits Winsopia to use the ICA Program to the extent of the authorisations that Winsopia has obtained. There is no presumption that Winsopia is entitled to unlimited use, subject only to specific exclusions or limitations. The ICA provides for specific and limited permission to use the licensed software, together with express exclusions and limitations.

158. Clause 4.1.1(c) permits Winsopia to make and install copies of the ICA Program but subject to the proviso that: (i) such copying is in support of the level of use authorised; and (ii) Winsopia reproduces the copyright notices and any other legends of ownership on each whole or partial copy.

159. Clause 4.1.1(d) permits Winsopia to use any portion of the ICA Program that is provided by IBM in source form or marked restricted only to: (i) resolve problems related to the use of the ICA Program; and (ii) modify the ICA Program so that it will work together with other products.

160. It follows from the above provisions that under the ICA, authorised use of the ICA Programs is limited to use by Winsopia, on the designated Winsopia mainframe, for the execution and running of the defined ICA Programs.

The ICA Programs

161. The scope of the licence granted and authorised use in clause 4.1 applies to the ICA Programs.

162. ICA Program is defined in clause 1.3 as:

> "an IBM Program licensed under Part 4 of this Agreement."

163. Other IBM Program is defined as:

> "an IBM Program licensed under a separate IBM licence agreement (e.g., IBM International Program Licence Agreement)."

164. Non-IBM Program is defined as:

> "a Program licensed under a separate third party licence agreement."

165. It is common ground that the ICA Programs are as set out in section 6 of the Agreement for Third Party Program Access, signed by IBM, Winsopia and RSM on 15 August 2013 (together with the updated and supplemental ICA Programs) and listed above. The issue is whether, as submitted by IBM, the constituent parts of such programs fall within the definition of an ICA Program, or as submitted by the defendants, for the purpose of the ICA, the definition of an ICA Program is limited to the whole of any such program.

166. Program is separately defined in clause 1.3 as:

> "the following, including the original and all whole or partial copies:
>
> a. machine readable instructions and data;
>
> b. components;
>
> c. audio-visual content (such as images, text, recordings, or pictures); and
>
> d. related licenced materials.

> The term "Program" includes any ICA Program, Other IBM Program, or Non-IBM Program that IBM may provide to the customer. The term does not include Machine Code or Materials."

167. The natural and ordinary meaning of the above words used is that an ICA Program falls within the definition of a Program; and a Program is defined as comprising any and/or all of the computer artefacts described in a. through to d. If the intention were to refer to an ICA Program only in its full, composite form, one would expect that to be stated expressly. The reference to the inclusion of whole or partial copies of those artefacts in the definition is a clear indication that a Program could be less than the sum of its parts. Analysis of the words used suggests that an ICA Program includes whole or partial copies of the program and any of the constituent parts described in a. through to d.

168. Each ICA Program is a collection of component parts, including function and data software code, code fragments, routines and sub-routines. The descriptions of the artefacts in (a) to (d) are general and broad. Machine readable instructions and data encompass software in source code, assembler language or object code. This includes control sections ("CSECTs"), macros (assembly code statements that are expanded to machine-readable code during assembly), copybooks (source code statements that are expanded to machine-readable code during compilation) and metadata (data about data). Components are distinct from the whole program, comprising any identifiable part of an ICA Program, including software modules, routines and sub-routines. Images and text include software manuals for the ICA Program. Related licenced materials is self-explanatory, including licensed materials necessary to enable the ICA Program to be run on the mainframe.

169. Support for that construction can be found by examining the design of computer software programs, such as the ICA Programs in this case. Professor Weissman's evidence is that almost all non-trivial programs are built, not as a single, large component but as multiple, separate components that are interconnected. The number of individual components and the method by which they are interconnected is a matter for the designer of the program. The advantages of such a design approach include the ability to alter one of the components, such as the runtime component, without any impact on, or the need to alter, the other components, such as the application program component.

170. Similarly, it is a matter of choice for the designer of the program as to when and how code is inserted and/or generated by a program component in response to instructions in a customer application. A program component can generate and insert in-line code that implements functionality directly into an application program; alternatively, it can generate code that calls a service from another component; or a combination of the above. Functional code can be generated at the time that the instruction is processed by the program component or code can be inserted to generate functionality dynamically at a later stage, such as runtime. Regardless of when or how ICA Program code is generated or inserted, it forms an essential part of the computer program.

171. It follows that the natural and ordinary meaning of ICA Program encompasses each of its constituent parts as well as the whole program.

172. The defendants submit, correctly, that the identified ICA Programs have each been individually licensed as discrete units. However, it does not follow that the scope of the licence for each discrete unit is engaged only in respect of that product as a whole, rather than each component part of the program.

173. It is said by the defendants that where IBM intends to refer to "portions" or "parts" of the ICA Programs, that is stated expressly. There are express references to portions or parts of the ICA Programs where the context requires distinction between specific parts of a program and others. An example is the reference in clause 4.1.1(a) to "machine-readable portion"; it is logical to distinguish this portion from text or other parts of the program that are not read by machine because the context is the Designated Machine on which such portion may be used. A further example is the reference in clause 4.1.1(d) to "any portion" provided in source form or marked restricted; it is logical to distinguish these parts from other parts of the program because they are subject to specific additional restrictions on use. Those references do not override the clear definition of Program which includes "components" and other parts.

174. As submitted by IBM, the restrictions on use are inextricably linked to the authorised use under the ICA. The scope of the licence granted in clause 4.1 is for use of the ICA Programs. If the defendants' construction were correct, Winsopia would not be permitted to use any component part of an ICA Program unless it involved use of the whole program, including all component parts. The defendants accept that an ICA Program may comprise many thousands or even millions of individual computer programs, routines and/or sub-routines, each themselves comprised of numerous pieces of code fragments, routines and sub-routines. Executing an application on the mainframe may use many different component parts of different programs but none uses any named ICA Program in its entirety. The defendant's construction would entail a restriction on use that would deprive Winsopia of any real benefit from the ICA. In practice, it would be unable to compile, assemble, link-edit and execute applications. Such restricted use of the software is unrealistic and would not make commercial sense. It is a very strong indication that the defendants' construction is wrong.

175. In conclusion on this issue, the permission granted and restrictions imposed in respect of the use of the ICA Programs under the ICA apply to the whole of each identified ICA Program and to any component part of such program.

Customer applications

176. The defendants' case is that the ICA terms and conditions do not apply to third party customer applications. If, and to the extent that, IBM modules, data and code fragments form part of compiled and link-edited load modules which originated from third party customers, rather than IBM, they do not constitute ICA Programs and therefore their use is not restricted under the ICA.

177. IBM's case is that the ICA does not distinguish between ICA Programs provided by IBM and those obtained through other means. Therefore, if

Winsopia obtains a copy of an ICA Program from a third party, it is subject to the ICA. IBM does not contend that a customer application becomes an ICA Program when compiled and link-edited. Rather, its case is that a compiled and link-edited customer application contains elements of ICA Program code, inserted or generated as a result of the compilation and link-editing process.

178. When considering this question, it is important to recognise that the court is not concerned with ownership of physical objects. The ICA is concerned with intellectual property rights and the terms on which permission is granted to use software. Whether a third-party customer application, or any part of it, becomes subject to the terms and conditions of the ICA depends on the use made of the application by Winsopia.

179. The starting point is to consider the process of creating an executable load module using a customer application.

i) The customer application is written by the customer in high level language source code or assembly code.

ii) If the source code includes an EXEC CICS command, it is pre-processed by the CICS translator, which replaces each EXEC CICS statement by generating a sequence of language specific source code statements; those sequences invoke a CALL to a CICS routine called DFHEI1, with a particular parameter list.

iii) An optimised version of the source code is compiled using an IBM compiler that: (a) translates the business logic of the source code into object code; (b) inserts data or other code invoked by copybook statements in the source code; and (c) generates initialisation code that sets up data structures in working memory and instructions for executing this sequence of code at runtime.

iv) Where the customer application is written in assembly code, the assembler: (a) converts each assembler mnemonic instruction into its corresponding machine instruction; and (b) expands any macro by executing a call to the relevant macro library and inserting the code instructions or data from the macro definition.

v) The resulting executable machine code from the compiler or assembler is known as a "User CSECT".

vi) The User CSECT is link-edited (or bound) with other User CSECTs and IBM CSECTs, comprising additional executable machine code instructions invoking Language Environment runtime services and/or data to form an executable load module.

vii) The load module is loaded into the storage area of the mainframe where it can be executed.

180. It is apparent from the above summary that IBM CSECTs, other code fragments and data are inserted into the customer application during the process of

compilation and/or link-editing. They are machine-readable instructions and data generated by the ICA Programs; as such, they each fall within the definition of an ICA Program.

181. The process of compilation and link-editing does not transform a customer application into an ICA Program. However, the process adds or inserts into the customer application components that form part of the ICA Programs. Use of such ICA Program components is subject to the terms and conditions of the ICA.

182. Test applications written by, or on behalf of, Winsopia that are compiled and link-edited on the Winsopia mainframe contain IBM CSECTs that are subject to the restrictions on use in clause 4 of the ICA.

183. Likewise, third-party customer applications that are compiled and link-edited on third party mainframes and supplied to Winsopia contain IBM CSECTs that are subject to the permissions and restrictions on use in the relevant licence. The customer applications may be owned by the customer but such intellectual property rights in the application do not extend to the IBM CSECTs. As is made clear in clause 4.1 of the ICA, ICA Programs are copyrighted and licensed, not sold. In order to study, test and observe the load modules supplied by third-party customers, Winsopia must load them into memory and/or run them on the mainframe. The acts of loading, displaying, running and storage of applications containing IBM CSECTs involve reproduction of the IBM software and are subject to authorisation. Winsopia's permission to do this derives only from the terms of the ICA. As explained above, permission to use the ICA Programs is subject to the restrictions in clause 4.

184. It does not follow that all IBM proprietary code in third-party customer applications will be subject to the ICA just because it passes through Winsopia's possession. If Winsopia loads a third-party customer load module on its mainframe and packages or scrubs it using CPX, this invariably involves copying or reproduction of IBM code within the application. Further, any use of the runtime environment is subject to the terms and conditions of the ICA, as is any code inserted or generated at runtime. However, mere possession of an application does not necessarily convert all IBM code in the application into an ICA Program (although it is likely that such code will be subject to the terms of the licence under which it was originally generated, potentially imposing restrictions on use by the customer under its licence agreement). In each case, it is necessary to examine Winsopia's actions in respect of the customer application, to determine whether they fell within or outside the scope of permitted use under the ICA.

Licensed Program Specifications

185. The defendants rely on the terms of Licensed Program Specifications as giving permission for the redistribution of customer applications incorporating IBM CSECTs and other code fragments to third parties.

186. Specifications are defined in clause 1.3 as:

> "information specific to a Product … ICA Program Specifications are in a document entitled "Licensed Program Specifications"."

187. Clause 4.1.2 sets out the Customer's Additional Obligations in respect of each ICA Program, including agreement to:

> "a. comply with any additional or different terms in its Licensed Program Specifications or an Attachment or Transaction Document."

188. The Licensed Program Specifications relied on by the defendants include Language Environment Vendor Interfaces – z/OS v 1.13 (SA22-7568-12). That Licensed Program Specification states that IBM z/OS Language Environment provides common services and language specific routines in a single runtime environment for languages including C, C++, COBOL and PL/I. It is the prerequisite runtime environment for applications generated with IBM compiler products.

189. The Licensed Program Specification describes in detail the steps involved in using the services provided by Language Environment, including the conventions required to call the services, and contains sample application programs to illustrate various programming techniques. It also contains the following permission notice:

> "This book includes information about certain callable service stub and linkage-assist (stub) routines contained in specific data sets that are intended to be bound or link-edited with code and run on z/OS systems. In connection with your authorized use of z/OS, you may bind or link-edit these stubs into your modules and distribute your modules with the included stubs for the purposes of developing, using, marketing and distributing programs conforming to the documented programming interfaces for z/OS, provided that each stub is included in its entirety, including any IBM copyright statements ..."

190. The words are clear that, contrary to the defendants' submissions, a licensee is permitted to distribute customer load modules incorporating IBM CSECTs, other code fragments and routines but such permission is limited to authorised use of z/OS, which requires a licence. Furthermore, such distribution must be for the purposes of developing, using, marketing and distributing programs conforming to the documented programming interfaces for z/OS, necessitating the IBM Language Environment runtime for which an appropriate licence is required. Finally, the stipulation that IBM copyright statements must be included in the IBM stubs incorporated into the load module demonstrates that use of such IBM materials remains subject to licence.

191. It follows that this explicit permission notice does not change the above analysis as to the applicability of the ICA permission and restriction provisions to customer applications that have been compiled and link-edited.

192. The defendants also place reliance on the Notices in the Licensed Program Specifications ("LPS") which state:

> "… Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead…"

193. This does not advance the arguments one way or another; although it indicates that functionally equivalent products, programs or services may be used, the proviso is that they do not infringe any IBM intellectual property right. Thus, it does not answer the question whether there would be any infringement or breach of the licence conditions by use of any alternative service. To answer that question, it is necessary to refer to the terms of the ICA, which sets out the basis on which use of the ICA Programs is permitted.

Independent software vendors (ISVs)

194. Reliance is placed by the defendants on the availability of software products for z/OS mainframes by independent software vendors ("ISVs") as relevant context to the ICA. Such products can replace or extend the functionality of IBM programs that operate on the mainframe, or provide new functionality. Mr Stephens explained that these ISVs typically license their software and provide ongoing support, usually requiring regular payments for licensing fees and support. In cross-examination, Mr Swanson agreed that in 2013 it was common practice for ISVs to share customer applications by shipping load modules in link-edited form.

195. This factual matrix is acknowledged but it does not assist in the proper construction of the ICA for the purpose of the issues in dispute. As set out in the extract from the Licensed Program Specification above, IBM confirmed that distribution of customer load modules was permitted, provided that: (i) it was in connection with authorised use of z/OS (that is, in accordance with the terms of the relevant ICA) and (ii) each stub was included in its entirety, including IBM copyright statements. The sharing and distribution of such products for use on the z/OS system may be widespread and is not necessarily prohibited but any use of the z/OS system remains subject to the terms and conditions of the ICA.

Debugging tools

196. The z/OS operating system licenced to Winsopia under the ICA includes diagnostic and error-correction ("debugging") tools which facilitate in depth analysis of software as it executes on an IBM mainframe. In addition, there are commercially available tools which perform similar debugging functions.

197. Such tools include the LIST option within the compiler, which, if specified, will cause the compiler to emit a document known as a compiler listing, which contains information about the compilation of a given source file, such as an assembly code representation of the machine instructions that the compiler

would generate for the program, and the options that were passed to the compiler.

198. Likewise, an assembler listing can produce assembler language source statements and a pseudo-assembly listing from the object code. The reference to pseudo-assembly reflects the fact that the original assembly code is shown but: (i) any comments in the original assembly program will be lost (because they do not form part of the executable content of the program and are discarded during assembly); and (ii) macro expansions created during assembly will not be reversed. IBM documentation states that the assembler language source file and listing can be used for purposes such as program understanding, debugging, and recovery of lost source code.

199. A "dump" can be used to assist in resolving a problem. If an error occurs during the execution of a program, the DUMP command can be used to display the storage contents in memory at the time of the error, such as which program was executing, where the error occurred and a code for the error. Typically, the information is shown as a hexadecimal representation of the data on the left and a character interpretation on the right of the display.

200. A "trace" records the processing or flow of a program or transaction as it executes, including network communications, memory content and the interactions between programs and transactions. The trace is saved in memory or exported to an external dataset such as General Tracing Facility ("GTF"). Interactive Problem Control System ("IPCS"), included as part of z/OS, can be used to format, display and analyse traces and dumps.

201. Other stand-alone products were installed on the Winsopia mainframe that have the capability to disassemble object code, including Colesoft z/XDC. Colesoft z/XDC is a commercially available debugging tool that enables a breakpoint to be set at a specific point in a load module, stopping execution and transferring control to the XDC program. It allows the user to step through the code and examine the execution of each instruction, view data areas and registers, disassemble object code and display dumps of memory in assembler language.

202. A common feature of the debugging tools available for use with z/OS is that they open a window into the internal implementation of the functions specified in application programs. By analysing the information produced by listings, dumps, traces and other tools, it is possible to deconstruct, not just the input and output of an application program, but also the code inserted at various stages in the process and the sequence of specific steps and instructions forming the internal logic of the operating system.

203. Mr Swanson agreed in cross-examination that IBM encourages the use of such tools by customers for testing and debugging their programs. Mr Stephens' evidence was that disassembly of IBM code by debugging tools could be used when resolving problems, such as diagnosing and fixing errors, but that it was rare for customers to do so directly. This is borne out by a cursory read of the published debugging guides. The IBM documentation for using the z/OS DUMP command indicates that it is used at IBM's request, when the IBM customer service providers are involved in diagnosing and solving a problem;

and the IBM documentation for tracing states that the CTRACE facilities are primarily used by IBM service personnel for diagnosing problems.

204. Mr Swanson and Mr Stephens agreed that it was possible for some of the debugging tools to be used to step through the internal operation of the code, beyond legitimate error diagnosis and correction. It is clear from the documentary evidence recorded in the IBM Corp support portal, that there were limited instances of express or tacit permission given by IBM for a customer to carry out disassembly when seeking help in fixing faults but there was no evidence that this was a widespread practice.

205. Notwithstanding the fact that the debugging tools could be used for purposes other than error diagnosis and correction, this does not imply permission for such purposes by IBM. It may be legitimate for a person to buy tools, such as screwdrivers, hammers and angle grinders, intended for DIY purposes, from a hardware centre; but that does not implicitly entitle the purchaser to use them to break into the hardware centre or steal a motorcycle in the street.

206. The limitations on legitimate use of debugging tools are illustrated by the z/OS HLASM Toolkit guide, which includes the following warning about copyright:

> "When you use this utility you must be aware of and respect the intellectual property rights of others. You are not authorized to use this utility to disassemble, copy, or create assembly listings or disassembled Assembler Language source code in violation of any contractual or other legal obligation. You are authorized to use this utility only for object code for which you have verified you have the right to perform disassembly."

207. It follows that the availability and use of debugging tools, including those distributed by IBM, does not of itself assist in answering the question whether the specific use of tools by Winsopia in this case was prohibited expressly or implicitly by the ICA. It is necessary to consider the details of each allegation in turn to determine whether Winsopia's activities fell within permitted use on a proper construction of the ICA.

Restrictions on use of ICA Programs

208. Clause 4.1.3 of the ICA imposes express restrictions on use of the ICA Programs by Winsopia:

> "The Customer agrees not to:
>
> a. reverse assemble, reverse compile, otherwise translate, or reverse engineer the ICA Program unless expressly permitted by applicable law without the possibility of contractual waiver; or
>
> b. sublicence, assign, rent, or lease the ICA Program or transfer it outside the Customer's Enterprise."

209. Clause 4.1.1(d) of the ICA limits authorised use of the ICA Programs:

> "Under each licence, IBM authorises the Customer to:
>
> …
>
> d. use any portion of the ICA Program IBM provides i) in source form, or ii) marks restricted (for example "Restricted materials of IBM") only to: (1) resolve problems related to the use of the ICA Program, and (2) modify the ICA Program so that it will work together with other projects."

210. Under clause 4.1.3(a), Winsopia was prohibited from reverse assembly, reverse compilation, other translation or reverse engineering of any ICA Program. These are not defined terms in the ICA. The experts have not identified any standard definition of them in the computer science community. They do not agree the precise definitions of these terms for the purpose of their application to the allegations of breach in this case and their differences are examined in the context of the relevant allegations. At a high level of generality, there appears to be agreement between the experts that reverse assembly can be described as the act of using a tool to rebuild assembly code from binary object code; decompiling can be described as the act of using a tool to recreate high-level language source code from a program's binary object code; translation changes the language or form of the code whilst retaining its sense; and reverse engineering can be described as the testing or analysis of the internal structures or workings of an application to ascertain how it has been built and/or how it is executed. It should be stressed that each allegation of breach must be analysed in turn to determine whether or not the relevant activity amounted to any of the prohibited acts.

211. Under clause 4.1.3(b), Winsopia was prohibited from transferring ICA Programs outside its Enterprise. In this case, the Enterprise is the legal entity that is Winsopia. This provision would be engaged whenever Winsopia transferred an ICA Program, or component thereof, to a third party.

212. Under clause 4.1.1(d), Winsopia could only use the source code of programs supplied in this form for certain narrow purposes; in particular, Winsopia was not permitted to facilitate use of this source code by third parties.

213. The above restrictions on use of the ICA Programs are express and clear. However, it is common ground that they must be read subject to the Software Directive and CDPA, to which I now turn.

Legislative framework

214. The defendants rely on the Software Directive, embodied in English Law by the CDPA, in support of their defence that, regardless of the terms of the ICA, as a lawful user of the ICA Programs, Winsopia had the benefit of its statutory rights to make back-up copies, decompile, observe, study and test the functioning of the computer programs, copy and adapt them in accordance with their intended purpose, including for error correction and interoperability.

215. The defendants' case is that the ICA must be construed against the background of the Software Directive so as to permit the acts taken by them to develop the SDM; alternatively, the provisions of the ICA that contradict the relevant provisions of the Software Directive must be null and void.

216. IBM disputes that the Software Directive is engaged; the allegations of reverse engineering advanced in this case do not fall within the scope of the permissive provisions relied on by the defendants. The terms and conditions of the ICA relied on by IBM are clear and valid.

217. The facts and matters the subject of the key allegations in this dispute occurred prior to the end of 2023. For the purpose of these claims, the European Union (Withdrawal) Act 2018 (as amended by the European Union (Withdrawal) Act 2020) is applicable. Although those Acts were amended by the Retained EU Law (Revocation and Reform) Act 2023, which disapplies general principles of EU law, that provision applies only after the end of 2023. Therefore, EU-derived domestic legislation, as it had effect in domestic law immediately before 30 January 2020, continued to have effect in respect of these claims as set out in section 5(2) of the 2018 Act.

218. As explained in *Wright & Ors v BTC Core & Ors* [2023] EWCA Civ 868 per Arnold LJ at [34]:

> "Legislation which transposed an EU directive into domestic law prior to 31 December 2020 remains part of UK law unless and until it has been repealed or amended. None of the legislation which implemented the Software Directive and the Information Society Directive has been repealed or amended, and the CDPA remains in force (as amended). Furthermore, the principle of supremacy of EU law continues to apply "so far as relevant to the interpretation, disapplication, or quashing of any enactment or rule of law passed or made before" 31 December 2020: see section 5(2) of the European Union (Withdrawal) Act 2018 and *R (Open Rights Group) v Secretary of State for the Home Department* [2021] EWCA Civ 800 at [12]–[13] per Warby LJ. Although this Court now has the power under the 2018 Act (as amended) to depart from decisions of the CJEU rendered before 31 December 2020 in an appropriate case, the default position is that such decisions remain binding."

219. It is well established that domestic legislation enacted or amended to implement a European directive must be construed so far as is possible in conformity with, and to achieve the result intended by, the directive: Case C-106/89 *Marleasing SA v La Comercial Internacional de Alimentación SA* [1990] ECR I-4135 at [8]; *Litster v Forth Dry Dock and Engineering Co Ltd* [1990] 1 AC 546, HL at 558C-H (Lord Templeman) and 576E-577D (Lord Oliver of Aylmerton); Joined Cases C-397/01 to C-403/01 *Pfeiffer v Deutsches Rotes Kreuz, Kreisverband Waldshut eV* [2004] ECR I-8835 at [113]-[117]; and *R (IDT Card Services Ireland Ltd) v Commissioners for Her Majesty's Revenue and Customs* [2006] EWCA Civ 29, [2006] STC 1252 at [73]-[92] (Arden LJ).

220. This requires the court to address any potential incompatibility between such domestic legislation and EU law by a purposive interpretation: *Marleasing SA v La Comercial Internacional de Alimentacion SA* (Case C-106/89) at [8], subject to the proviso that the meaning should go with the grain of the legislation and be compatible with the underlying thrust of the legislation being construed: *Vodafone 2 v Revenue and Customs Commissioners* [2009] EWCA Civ 446 per Sir Andrew Morritt at [37]-[38].

221. Further, the court must interpret both European and domestic legislation as far as possible in the light of the wording and purpose of relevant international agreements to which the UK/EU is a party, such as TRIPS and the WIPO Copyright Treaty.

222. I accept the defendants' primary argument as to the approach to construction, namely, that the ICA must be construed against the factual matrix of the Software Directive, so as to avoid any conflict. Clause 4.1.3 of the ICA, prohibiting acts of reverse engineering, contains the proviso: *"unless expressly permitted by applicable law without the possibility of contractual waiver"* indicating a common intention to read the words in the light of, and subject to, provisions such as the Software Directive. If it were impossible to construe the relevant provisions of the ICA in accordance with the provisions of the Software Directive, it is clear that the defendants' alternative approach to construction, namely, that the conflicting provisions should be null and void, would apply.

223. Both sides referred to *travaux preparatoires* as materials to assist in the interpretation of the Software Directive. Such materials are an aide only where there is ambiguity in the words used in the Directive. Although they are of general interest and provide useful background information, the parties have not identified any ambiguity that requires reference to those materials to determine specific issues in this case.

Berne Convention

224. The International Convention for the Protection of Literary and Artistic works (initially signed at Berne on 9 September 1886, revised by the Paris Act of 1971 and amended in 1979) ("the Berne Convention"), to which the United Kingdom and all Member States of the European Union are parties, includes the following provisions:

> "Article 2
>
> (1) The expression 'literary and artistic works' shall include every production in the literary, scientific and artistic domain, whatever may be the mode or form of its expression, such as books, pamphlets and other writings; lectures, addresses, sermons and other works of the same nature; dramatic or dramatico-musical works; choreographic works and entertainments in dumb show; musical compositions with or without words; cinematographic works to which are assimilated works expressed by a process analogous to cinematography; works of drawing, painting, architecture, sculpture, engraving

and lithography; photographic works to which are assimilated works expressed by a process analogous to photography; works of applied art; illustrations, maps, plans, sketches and three-dimensional works relative to geography, topography, architecture or science.

(2) It shall, however, be a matter for legislation in the countries of the Union to prescribe that works in general or any specified categories of works shall not be protected unless they have been fixed in some material form.

…

(5) Collections of literary or artistic works such as encyclopedias and anthologies which, by reason of the selection and arrangement of their contents, constitute intellectual creations, shall be protected as such, without prejudice to the copyright in each of the works forming part of such collections.

(6) The works mentioned in this Article shall enjoy protection in all countries of the Union. This protection shall operate for the benefit of the author and his successors in title.

…

Article 9

(1) Authors of literary and artistic works protected by this Convention shall have the exclusive right of authorising the reproduction of these works, in any manner or form.

(2) It shall be a matter for legislation in the countries of the Union to permit the reproduction of such works in certain special cases, provided that such reproduction does not conflict with a normal exploitation of the work and does not unreasonably prejudice the legitimate interests of the author.

…

Article 20

The Governments of the countries of the Union reserve the right to enter into special agreements among themselves, in so far as such agreements grant to authors more extensive rights than those granted by the Convention, or contain other provisions not contrary to this Convention. The provisions of existing agreements which satisfy these conditions shall remain applicable."

TRIPS

225. The Agreement on Trade-related Aspects of Intellectual Property Rights ("TRIPS") which forms Annex 1C to the Agreement establishing the World Trade Organisation signed in Morocco on 15 April 1994, to which the United Kingdom and the European Union and all its Member States are parties, includes the following provisions:

> "Article 9
>
> *Relation to the Berne Convention*
>
> 1. Members shall comply with Articles 1 through 21 of the Berne Convention (1971) and the Appendix thereto. However, Members shall not have rights or obligations under this Agreement in respect of the rights conferred under Article 6 of that Convention or of the rights derived therefrom.
>
> 2. Copyright protection shall extend to expressions and not to ideas, procedures, methods of operation or mathematical concepts as such.
>
> Article 10
>
> *Computer Programs and Compilations of Data*
>
> 1. Computer programs, whether in source or object code, shall be protected as literary works under the Berne Convention (1971).
>
> 2. Compilations of data or other material, whether in machine readable or other form, which by reason of the selection or arrangement of their contents constitute intellectual creations, shall be protected as such. Such protection, which shall not extend to the data or material itself, shall be without prejudice to any copyright subsisting in the data or material itself.
>
> …
>
> Article 13
>
> *Limitation and Exceptions*
>
> Members shall confine limitations or exceptions to exclusive rights to certain special cases which do not conflict with a normal exploitation of the work and do not unreasonably prejudice the legitimate interests of the author."

WIPO

226. The World Intellectual Property Organisation Copyright Treaty agreed in Geneva on 20 December 1996 ("the WIPO Copyright Treaty"), to which the United Kingdom and the European Union and all its Member States are parties, includes the following provisions:

"Article 1

Relation to the Berne Convention

(1) This Treaty is a special agreement within the meaning of Article 20 of the Berne Convention for the Protection of Literary and Artistic Works, as regards Contracting Parties that are countries of the Union established by that Convention. This Treaty shall not have any connection with treaties other than the Berne Convention, nor shall it prejudice any rights and obligations under any other treaties.

(2) Nothing in this Treaty shall derogate from existing obligations that Contracting Parties have to each other under the Berne Convention for the Protection of Literary and Artistic Works.

…

(4) Contracting Parties shall comply with Articles 1 to 21 and the Appendix of the Berne Convention.

Article 2

Scope of Copyright Protection

Copyright protection extends to expressions and not to ideas, procedures, methods of operation or mathematical concepts as such.

Article 3

Application of Articles 2 to 6 of the Berne Convention

Contracting Parties shall apply mutatis mutandis the provisions of Articles 2 to 6 of the Berne Convention in respect of the protection provided for in this Treaty.

Article 4

Computer Programs

Computer programs are protected as literary works within the meaning of Article 2 of the Berne Convention. Such protection applies to computer programs, whatever may be the mode or form of their expression.

Article 5

Compilations of Data (Databases)

> Compilations of data or other material, in any form, which by reason of the selection or arrangement of their contents constitute intellectual creations, are protected as such. This protection does not extend to the data or the material itself and is without prejudice to any copyright subsisting in the data or material contained in the compilation.
>
> …
>
> Article 10
>
> Limitation and Exceptions
>
> (1) Contracting Parties may, in their national legislation, provide for limitations of or exceptions to the rights granted to authors of literary and artistic works under this Treaty in certain special cases which do not conflict with a normal exploitation of the work and do not unreasonably prejudice the legitimate interests of the author.
>
> (2) Contracting Parties shall, when applying the Berne Convention, confine any limitations of or exceptions to rights provided for therein to certain special cases which do not conflict with a normal exploitation of the work and do not unreasonably prejudice the legitimate interests of the author."

227. The general principles that can be derived from the above Convention and Treaties are that:

i) computer programs are protected as literary works within the meaning of the Berne Convention;

ii) copyright protection for computer programs extends to expressions but not to ideas, procedures, methods of operation or mathematical concepts as such; and

iii) it is a matter for the parties to the Convention and Treaties to legislate for limitations and exceptions to copyright protection, provided that the permitted reproduction does not conflict with normal exploitation of the work and does not unreasonably prejudice the legitimate interests of the author.

Software Directive

228. The above principles are reflected in the Software Directive of 2009 (amending the earlier Directive 91/250/EEC), which includes the following Recitals:

i) Recital (6):

> "The Community's legal framework on the protection of computer programs can accordingly in the first instance be

limited to establishing that Member States should accord protection to computer programs under copyright law as literary works and, further, to establishing who and what should be protected, the exclusive rights on which protected persons should be able to rely in order to authorise or prohibit certain acts and for how long the protection should apply."

ii) Recital (7):

"For the purpose of this Directive, the term 'computer program' shall include programs in any form, including those which are incorporated into hardware. This term also includes preparatory design work leading to the development of a computer program provided that the nature of the preparatory work is such that a computer program can result from it at a later stage."

iii) Recital (10):

"The function of a computer program is to communicate and work together with other components of a computer system and with users and, for this purpose, a logical and, where appropriate, physical interconnection and interaction is required to permit all elements of software and hardware to work with other software and hardware and with users in all the ways in which they are intended to function. The parts of the program which provide for such interconnection and interaction between elements of software and hardware are generally known as 'interfaces'. This functional interconnection and interaction is generally known as 'interoperability'; such interoperability can be defined as the ability to exchange information and mutually to use the information which has been exchanged."

iv) Recital (11):

"For the avoidance of doubt, it has to be made clear that only the expression of a computer program is protected and that ideas and principles which underlie any element of a program, including those which underlie its interfaces, are not protected by copyright under this Directive. In accordance with this principle of copyright, to the extent that logic, algorithms and programming languages comprise ideas and principles, those ideas and principles are not protected under this Directive. In accordance with the legislation and case-law of the Member States and the international copyright conventions, the expression of those ideas and principles is to be protected by copyright."

v) Recital (13):

"The exclusive rights of the author to prevent the unauthorised reproduction of his work should be subject to a limited exception

> in the case of a computer program to allow the reproduction technically necessary for the use of that program by the lawful acquirer. This means that the acts of loading and running necessary for the use of a copy of a program which has been lawfully acquired, and the act of correction of its errors, may not be prohibited by contract. In the absence of specific contractual provisions, including when a copy of the program has been sold, any other act necessary for the use of the copy of a program may be performed in accordance with its intended purpose by a lawful acquirer of that copy."

vi) Recital (14):

> "A person having a right to use a computer program should not be prevented from performing acts necessary to observe, study or test the functioning of the program, provided that those acts do not infringe the copyright in the program."

vii) Recital (15):

> "The unauthorised reproduction, translation, adaptation or transformation of the form of the code in which a copy of a computer program has been made available constitutes an infringement of the exclusive rights of the author. Nevertheless, circumstances may exist when such a reproduction of the code and translation of its form are indispensable to obtain the necessary information to achieve the interoperability of an independently created program with other programs. It has therefore to be considered that, in these limited circumstances only, performance of the acts of reproduction and translation by or on behalf of a person having a right to use a copy of the program is legitimate and compatible with fair practice and must therefore be deemed not to require the authorisation of the rightholder. An objective of this exception is to make it possible to connect all components of a computer system, including those of different manufacturers, so that they can work together. Such an exception to the author's exclusive rights may not be used in a way which prejudices the legitimate interests of the rightholder or which conflicts with a normal exploitation of the program."

viii) Recital (16):

> "Protection of computer programs under copyright laws should be without prejudice to the application, in appropriate cases, of other forms of protection. However, any contractual provisions contrary to the provisions of this Directive laid down in respect of decompilation or to the exceptions provided for by this Directive with regard to the making of a back-up copy or to observation, study or testing of the functioning of a program should be null and void."

229. Article 1 of the Software Directive sets out the general provision for copyright protection for computer programs:

> "1. In accordance with the provisions of this Directive, Member States shall protect computer programs, by copyright, as literary works within the meaning of the Berne Convention for the Protection of Literary and Artistic Works. For the purposes of this Directive, the term 'computer programs' shall include their preparatory design material.
>
> 2. Protection in accordance with this Directive shall apply to the expression in any form of a computer program. Ideas and principles which underlie any element of a computer program, including those which underlie its interfaces, are not protected by copyright under this Directive."

230. Article 4 of the Software Directive sets out the protected rights of the copyright owner of a computer program:

> "1. Subject to the provisions of Articles 5 and 6, the exclusive rights of the rightholder within the meaning of Article 2 shall include the right to do or to authorise:
>
> (a) the permanent or temporary reproduction of a computer program by any means and in any form, in part or in whole; in so far as loading, displaying, running, transmission or storage of the computer program necessitate such reproduction, such acts shall be subject to authorisation by the rightholder;
>
> (b) the translation, adaptation, arrangement and any other alteration of a computer program and the reproduction of the results thereof, without prejudice to the rights of the person who alters the program;
>
> (c) any form of distribution to the public, including the rental, of the original computer program or of copies thereof."

231. Article 5 provides specific exceptions to the protection afforded by the Directive:

> "1. In the absence of specific contractual provisions, the acts referred to in points (a) and (b) of Article 4(1) shall not require authorisation by the rightholder where they are necessary for the use of the computer program by the lawful acquirer in accordance with its intended purpose, including for error correction.

> 2. The making of a back-up copy by a person having a right to use the computer program may not be prevented by contract in so far as it is necessary for that use.
>
> 3. The person having a right to use a copy of a computer program shall be entitled, without the authorisation of the rightholder, to observe, study or test the functioning of the program in order to determine the ideas and principles which underlie any element of the program if he does so while performing any of the acts of loading, displaying, running, transmitting or storing the program which he is entitled to do."

232. A further exception is set out in Article 6:

> "1. The authorisation of the rightholder shall not be required where reproduction of the code and translation of its form within the meaning of points (a) and (b) of Article 4(1) are indispensable to obtain the information necessary to achieve the interoperability of an independently created computer program with other programs, provided that the following conditions are met:
>
> (a) those acts are performed by the licensee or by another person having a right to use a copy of a program, or on their behalf by a person authorised to do so;
>
> (b) the information necessary to achieve interoperability has not previously been readily available to the persons referred to in point (a); and
>
> (c) those acts are confined to the parts of the original program which are necessary in order to achieve interoperability.
>
> 2. The provisions of paragraph 1 shall not permit the information obtained through its application:
>
> (a) to be used for goals other than to achieve the interoperability of the independently created computer program;
>
> (b) to be given to others, except when necessary for the interoperability of the independently created computer program; or
>
> (c) to be used for the development, production or marketing of a computer program substantially similar in its expression, or for any other act which infringes copyright.

> 3. In accordance with the provisions of the Berne Convention for the protection of Literary and Artistic Works, the provisions of this Article may not be interpreted in such a way as to allow its application to be used in a manner which unreasonably prejudices the rightholder's legitimate interests or conflicts with a normal exploitation of the computer program."

233. Article 8 precludes the use of contractual terms and conditions to override or circumvent the exceptions in Articles 5(2) and 5(3):

> "Any contractual provisions contrary to Article 6 or to the exceptions provided for in Article 5(2) and (3) shall be null and void."

Copyright, Designs and Patents Act 1988 (CDPA)

234. The material provisions of the Software Directive were implemented in the CDPA (as amended). Sections 1, 2 and 3 provide that copyright is a property right which subsists in original literary works, including a computer program.

235. Acts restricted by copyright in a work include the right to copy (including storage in any medium by electronic means), issue copies and adapt the work:

> Section 16
>
> "(1) The owner of the copyright in a work has, in accordance with the following provisions of this Chapter, the exclusive right to do the following acts in the United Kingdom—
>
> (a) to copy the work (see section 17);
>
> (b) to issue copies of the work to the public (see section 18);
>
> …
>
> and those acts are referred to in this Part as the "acts restricted by the copyright".
>
> (2) Copyright in a work is infringed by a person who without the licence of the copyright owner does, or authorises another to do, any of the acts restricted by the copyright.
>
> (3) References in this Part to the doing of an act restricted by the copyright in a work are to the doing of it—
>
> (a) in relation to the work as a whole or any substantial part of it, and
>
> (b) either directly or indirectly;
>
> and it is immaterial whether any intervening acts themselves infringe copyright."

Section 17

"(1) The copying of the work is an act restricted by the copyright in every description of copyright work; and references in this Part to copying and copies shall be construed as follows.

(2) Copying in relation to a literary, dramatic, musical or artistic work means reproducing the work in any material form. This includes storing the work in any medium by electronic means.

…

(6) Copying in relation to any description of work includes the making of copies which are transient or are incidental to some other use of the work."

236. Acts incidental to, and necessary for, the purposes of lawful use by a lawful user do not constitute an infringement of copyright:

Section 50A

"(1) It is not an infringement of copyright for a lawful user of a copy of a computer program to make any back up copy of it which it is necessary for him to have for the purposes of his lawful use.

(2) For the purposes of this section and sections 50B, 50BA and 50C a person is a lawful user of a computer program if (whether under a licence to do any acts restricted by the copyright in the program or otherwise), he has a right to use the program.

(3) Where an act is permitted under this section, it is irrelevant whether or not there exists any term or condition in an agreement which purports to prohibit or restrict the act (such terms being, by virtue of section 296A, void)."

237. Decompilation of a computer program is permitted for interoperability purposes subject to express limitations:

Section 50B

"(1) It is not an infringement of copyright for a lawful user of a copy of a computer program expressed in a low level language—

(a) to convert it into a version expressed in a higher level language, or

(b) incidentally in the course of so converting the program, to copy it, (that is, to "decompile" it), provided that the conditions in subsection (2) are met.

(2) The conditions are that—

(a) it is necessary to decompile the program to obtain the information necessary to create an independent program which can be operated with the program decompiled or with another program ("the permitted objective"); and

(b) the information so obtained is not used for any purpose other than the permitted objective.

(3) In particular, the conditions in subsection (2) are not met if the lawful user—

(a) has readily available to him the information necessary to achieve the permitted objective;

(b) does not confine the decompiling to such acts as are necessary to achieve the permitted objective;

(c) supplies the information obtained by the decompiling to any person to whom it is not necessary to supply it in order to achieve the permitted objective; or

(d) uses the information to create a program which is substantially similar in its expression to the program decompiled or to do any act restricted by copyright.

(4) Where an act is permitted under this section, it is irrelevant whether or not there exists any term or condition in an agreement which purports to prohibit or restrict the act (such terms being, by virtue of section 296A, void)."

238. The rights to observe, study or test the functioning of a computer program are permitted as provided in Article 5(3) of the Software Directive:

Section 50BA

"(1) It is not an infringement of copyright for a lawful user of a copy of a computer program to observe, study or test the functioning of the program in order to determine the ideas and principles which underlie any element of the program if he does so while performing any of the acts of loading, displaying, running, transmitting or storing the program which he is entitled to do.

(2) Where an act is permitted under this section, it is irrelevant whether or not there exists any term or condition in an agreement which purports to prohibit or restrict the act (such terms being, by virtue of section 296A, void). "

Section 50C

> "(1) It is not an infringement of copyright for a lawful user of a copy of a computer program to copy or adapt it, provided that the copying or adapting—
>
> (a) is necessary for his lawful use; and
>
> (b) is not prohibited under any term or condition of an agreement regulating the circumstances in which his use is lawful.
>
> (2) It may, in particular, be necessary for the lawful use of a computer program to copy it or adapt it for the purpose of correcting errors in it.
>
> (3) This section does not apply to any copying or adapting permitted under section 50A, 50B or 50BA."

239. Where a person has the use of a computer program under an agreement, any term or condition in the agreement shall be void in so far as it purports to prohibit or restrict the rights set out in sections 50A, section 50B, or section 50BA:

> Section 296A
>
> "(1) Where a person has the use of a computer program under an agreement, any term or condition in the agreement shall be void in so far as it purports to prohibit or restrict—
>
> (a) the making of any back up copy of the program which it is necessary for him to have for the purposes of the agreed use;
>
> (b) where the conditions in section 50B(2) are met, the decompiling of the program; or (c) the observing, studying or testing of the functioning of the program in accordance with section 50BA.
>
> (2) In this section, decompile, in relation to a computer program, has the same meaning as in section 50B."

Applicable legal principles

240. The parties adopted the convention of making submissions by reference to the provisions in the Software Directive. For the purpose of the issues in dispute in this case, the key principles that can be derived from the Software Directive are as follows:

i) Computer programs are protected as literary works within the meaning of the Berne Convention.

ii) The restricted rights of a copyright owner include loading, displaying, running, transmission, storage, translation, adaptation or distribution of part, or all, of a computer program by any means and in any form.

iii) Copyright protection for computer programs extends to expression of the program but not to ideas and principles underlying functionality of the program.

iv) Exceptions to the restricted rights include observation, study and testing of the functioning of a program to determine its underlying ideas and principles; and decompilation where necessary for the purpose of interoperability.

v) Contractual terms and conditions that seek to circumvent the above exceptions are null and void.

vi) A further exception to the restricted rights is error correction but that is subject to any term or condition of an agreement prohibiting such use.

vii) The limitations and exceptions to the copyright protection should be interpreted restrictively having regard to the right of the copyright owner to exploit the work and the legitimate interests of the author. See also: *Infopaq International A/S v Danske Dagblades Forening* (Case C-5/08) [2009] ECR I-6569 at [56] and [57].

241. The above principles were considered by Pumfrey J in *Navitaire Inc v easyJet Airline Company & Another* [2004] EWHC 1725 (Ch). The claim in that case was an action for infringement of copyright in computer software implementing an airline booking system, known as "OpenRes". It was common ground that the defendants did not have any access to the source code of OpenRes; the main allegation was that they had carried out "non-textual copying" of the user interface so that an agent or private individual seeking to make an airline booking would experience the screen displays and response to key commands as if they were using OpenRes. The court rejected the claim that copyright subsisted in the command names or their syntax on the basis that this amounted to a computer language:

> "[86] I consider that … it is not possible to infringe the copyright that subsists either in the source code for a parser or in the source code for a parser generator by observing the behaviour of the final program and constructing another program to do the same thing…
>
> …
>
> [94] Copyright protection for computer software is a given, but I do not feel that the courts should be astute to extend that protection into a region where only the functional effects of a program are in issue. There is a respectable case for saying that copyright is not, in general, concerned with functional effects, and there is some advantage in a bright line rule protecting only

> the claimant's embodiment of the function in software and not some superset of that software."

242. The court held that the character-based VT100 screen displays amounted to ideas underlying the software interfaces in the sense used in Article 1(2) of the Software Directive, providing the static framework for the display of the dynamic data which it was the task of the software to produce. As such, they were not subject to copyright protection. In contrast, the user interface provided by the GUI screens, which interacted with the COBOL server module, was found to be subject to copyright protection because it was the product of intellectual creativity:

> "The GUI screens stand in a different position. The Directive is concerned only with the protection of computer programs as literary works, and I do not read it as having any impact on relevant artistic copyrights. It is certainly possible to view the GUI screens as tables, because they are 'drawn' by selecting from a palette of available objects things such as command buttons, toggle buttons, checkboxes, scrolling lists and so forth and moving them around on a form until a satisfactory layout is concerned. The 'interface builder' program provides 'stubs' for the routines that will be executed when the user selects or clicks on one of these objects, and it is the task of the programmer to provide the necessary code to ensure that the right thing happens when the user presses (for example) the OK button. Although composed of elements made available by the manufacturer of the interface builder program, I can see that the screen resulting from such an operation might properly be considered to be an artistic work. What the programmer ultimately produces is code that depends upon a large number of complex graphic routines that draw the background, the boxes and the shading in the places selected, and act appropriately when the mouse moves over them or they are selected. The programmer does not write this code: it is the scaffolding for his or her own window design."

243. The court rejected Navataire's argument that the business logic of the software had been appropriated by non-textual copying:

> "[112] I shall return to first principles. For present purposes, a computer running a particular program is a deterministic machine. A particular input to the machine will produce a predictable result derived from all previous inputs to the machine. If therefore one studies a machine in operation, it should be possible to identify the machine's response to all possible sequences of inputs, and so construct a new machine that operates to give the same outputs for the same sequences of inputs by writing an appropriate program. Navitaire contend that if this is done, it follows axiomatically that any copyright in the source code for the first machine must be infringed in writing the second program. Indeed, it was urged on me at an earlier hearing that it was unnecessary to consider any of the source code for the

> OpenRes system in determining whether there had been copying of a substantial part of the copyright(s) subsisting in the source code for it.
>
> …
>
> [129] The questions in the present case are both a lack of substantiality and the nature of the skill and labour to be protected. Navitaire's computer program invites input in a manner excluded from copyright protection, outputs its results in a form excluded from copyright protection and creates a record of a reservation in the name of a particular passenger on a particular flight. What is left when the interface aspects of the case are disregarded is the business function of carrying out the transaction and creating the record, because none of the code was read or copied by the defendants. It is right that those responsible for devising OpenRes envisaged this as the end result for their program: but that is not relevant skill and labour. In my judgment, this claim for non-textual copying should fail.
>
> [130] I do not come to this conclusion with any regret. If it is the policy of the Software Directive to exclude both computer languages and the underlying ideas of the interfaces from protection, then it should not be possible to circumvent these exclusions by seeking to identify some overall function or functions that it is the sole purpose of the interface to invoke and relying on those instead. As a matter of policy also, it seems to me that to permit the 'business logic' of a program to attract protection through the literary copyright afforded to the program itself is an unjustifiable extension of copyright protection into a field where I am far from satisfied that it is appropriate."

244. Pumfrey J's reasoning in *Navataire* was approved as correct by the Court of Appeal in *Nova Productions Ltd v Mazooma Games Ltd* [2007] EWCA Civ 219.

245. These issues were considered in detail by Arnold J (as he then was) in *SAS Institute Inc v World Programming Ltd* [2010] EWHC 1829 ("*SAS No 1"*). SAS was a developer of analytical software programs ("the SAS System"), which enabled users to carry out a wide range of data processing and analysis tasks, and in particular statistical analysis. The core component of the SAS System was Base SAS, which enabled users to write and run application programs written in the SAS language to manipulate data. The functionality of Base SAS could be extended by the use of additional SAS components, providing enhanced features, such as statistical analysis, graph plotting and third-party data sources. SAS published detailed technical manuals, documenting the functionality of the SAS System and the SAS components but not the internal behaviour of the same. It was common ground that each of the SAS components was an original computer program in which copyright subsisted; further, that each of the SAS manuals was an original literary work in which copyright subsisted.

246. WPL was a competitor of the claimant and developed alternative software ("WPS"), which could execute application programs written in the SAS Language by emulating the functionality of the SAS Components as closely as possible with a view to ensuring that the same inputs would produce the same outputs.

247. The claim included allegations by SAS that WPL infringed copyright and acted in breach of contract when creating WPS by: (i) copying the SAS manuals; (ii) copying indirectly the SAS components; and (iii) using the learning edition of the SAS System outside the scope of the applicable licence.

248. SAS did not allege that WPL had access to the source code of the SAS components, or decompiled any SAS object code, or that WPL copied any of the text or structural design of the source code of the SAS components.

249. The court found that WPL used the SAS manuals to emulate functionality of the SAS System in WPS but did not reproduce the SAS source code by going beyond reproduction of its functionality. Further, WPL copied elements of the SAS source code into the source code of WPS by examining log files produced during use of the SAS learning edition. WPL also ran SAS test scripts through the SAS learning edition to observe the output produced and then through WPS to check whether WPS produced the same output or failed.

250. Having carried out a full and detailed analysis of the legislative background, Software Directive and relevant authorities, including *Navitaire*, *Nova Productions* and *Infopaq*, Arnold J stated:

> "[206]… it is necessary to distinguish between "expressions" on the one hand and "ideas, procedures, methods of operation and mathematical concepts as such" on the other. What is protected by copyright in a literary work is the form of expression of the literary work itself. Other things which are conveyed by or described in the literary work, of which "ideas, procedures, methods of operation and mathematical concepts" is evidently a non-exhaustive list, are not protected. Thus these provisions draw a line between copyright protection and the public domain…
>
> …
>
> [217] I am not persuaded that Pumfrey J was wrong to conclude that programming languages are not protected. While I acknowledge that recital [14] can be read in the manner contended for by counsel for SAS Institute, it should not be construed as if it were an operative provision in an English statute. It is there to guide courts as to the purpose of Article 1(2). It must be read in its context between recitals [13] and [15] (a point which is reinforced in the codified version by the fact that all three have been combined in one recital, recital (11)). Read in that context, the words "to the extent that" can be understood as meaning "in as much as". As for the legislative history, I do

not agree that this demonstrates, as counsel for SAS Institute argued, that an exclusion of programming languages was deliberately not included in the Software Directive. To the contrary, I consider that it indicates that Article 1(2) is to be broadly interpreted. Furthermore, I think that the distinction which Pumfrey J drew between a computer program and the language it is written in is, despite his hesitancy on the point, perfectly consistent with the distinction between expressions and ideas, procedures, methods of operation and mathematical formulae.

…

[226] Again, I am not persuaded that Pumfrey J was wrong to conclude that interfaces as described in recital [15] of the Software Directive are not protected by the copyright in a computer program. In my judgment the legislative history supports this interpretation. The inclusion of Article 6 in the Software Directive does not support the opposite interpretation. It can be seen from the passages quoted above that the purpose of Article 6 is to entitle third parties to obtain information about interfaces by decompiling the object code of a program where the necessary information is not available from either (i) published sources such as manuals, (ii) common standards or (iii) observation, study or testing of the program. Where information about an interface can be obtained in one or more of those ways, it is evident that the Commission considered that competitors would be free to copy the interface anyway.

…

[232] Even leaving aside the decision of the Court of Appeal in *Nova*, I am not persuaded that Pumfrey J was wrong to hold that it is not without more an infringement of the copyright in a computer program to create another computer program which has the same functionality. I accept that copyright protection is not limited to the text of the source code of the program, but extends to protecting the design of the program, that is, what has been referred to in some cases as its "structure, sequence and organisation". If there were any doubt about this, then the conferring of protection on "preparatory design material" confirms it. But there is a distinction between protecting the design of the program and protecting its functionality. It is perfectly possible to create a computer program which replicates the functionality of an existing program, yet whose design is quite different.

[233] In my judgment Pumfrey J was right to say that at [129] the key question is "the nature of the skill and labour"… Copyright in the computer program (including any preparatory design material) protects the skill, judgement and labour in

devising the form of expression of the program (including any preparatory design material), that is to say, its design and source code.

…

[236] Accordingly, I consider that the functionality of a computer program falls on the wrong side of the line drawn by Article 1(2) of the Software Directive, Article 9(2) of TRIPS and Article 2 of the WIPO Copyright Treaty.

[237] In any event, Pumfrey J's judgment on this point was upheld by the Court of Appeal in *Nova*, and that decision is binding on me unless and until overruled by either the Supreme Court or the Court of Justice.

…

[294] SAS Institute contends that Article 5(3) is a "for the avoidance of doubt" provision, which simply confirms that acts of observing, studying and testing a computer program are not infringements provided that the user is licensed to use the program in the manner in question…

…

[302] WPL contends that the words "the acts … which he is entitled to do" in Article 5(3) refer to the *kind* of acts which the user is entitled to do, not to their *purpose*. Thus WPL contends that the kinds of acts in question are those referred to immediately before the words "which he is entitled to do", namely "loading, displaying, running, transmitting or storing the program". Thus if the licence does not entitle the user to transmit the program, Article 5(3) does not permit transmission; but if the licence entitles the user to load and run the program, then the user can observe, study and test while loading and running…

…

[311] My view is that WPL's interpretation is to be preferred for the reasons given by counsel for WPL. I would add two points. First, the starting point is that Article 5(3) is expressed to be an exception to the restricted acts referred to in Article 4. In my opinion it follows that it should be interpreted as a positive defence to a claim of copyright infringement and not merely a "for the avoidance of doubt" provision with no substantive effect. This is a pointer against SAS Institute's interpretation.

[312] Secondly, Article 5(3) must be read together with the last sentence of Article 9(1). That makes it clear that the copyright proprietor cannot override Article 5(3) by contract. To allow

> copyright proprietors to override Article 5(3) by the use of standard form licence terms of the kind relied on by SAS Institute in the present case would make it very easy for the proprietors to circumvent Article 5(3). That would be contrary to the important public interest which underlies Article 5(3).
>
> [313] On the assumption that Article 5(3) is to be construed as WPL contends, it follows the licence terms for the Learning Edition are null and void to the extent that they make it an infringement for the user to observe, study and test the Learning Edition in order to determine the ideas and principles which underlie any element of the program…"

251. The conclusions reached by Arnold J included at [332]:

i) Although the court was not persuaded that Pumfrey J was wrong to conclude in *Navitaire* that, on the true interpretation of Article 1(2) of the Software Directive, copyright in computer programs does not protect from copying (a) programming languages, (b) interfaces (where achievable without decompilation of the object code) and (c) the functions of the programs, these were questions on which guidance from the ECJ was required.

ii) On the assumption that Pumfrey J's interpretation of Article 1(2) of the Software Directive was correct, WPL did not infringe SAS copyright in the SAS components or the SAS manuals by producing or testing WPS.

iii) On the court's interpretation of Article 5(3) of the Software Directive, a question on which guidance from the ECJ was required, WPL's use of the SAS learning edition was within Article 5(3); to the extent that the licence terms sought to prevent this, they were null and void. As a result, none of WPL's acts complained of was a breach of contract or an infringement of copyright, save that there was substantial reproduction of the SAS manuals in the WPL manual, thereby infringing copyright.

252. The CJEU gave its judgment on 2 May 2012: Case C-406/10 [2012] ECR I-0000. In response to the questions posed regarding the interpretation of Article 1(2) of the Software Directive, the court stated:

> "[39] … it must be stated that, with regard to the elements of a computer program which are the subject of Questions 1 to 5, neither the functionality of a computer program nor the programming language and the format of data files used in a computer program in order to exploit certain of its functions constitute a form of expression of that program for the purposes of Article 1(2) of Directive 91/250.
>
> [40] As the Advocate General states in point 57 of his Opinion, to accept that the functionality of a computer program can be protected by copyright would amount to making it possible to

> monopolise ideas, to the detriment of technological progress and industrial development.
>
> [41] Moreover, point 3.7 of the explanatory memorandum to the Proposal for Directive 91/250 [COM (88) 816] states that the main advantage of protecting computer programs by copyright is that such protection covers only the individual expression of the work and thus leaves other authors the desired latitude to create similar or even identical programs provided that they refrain from copying.
>
> [42] With respect to the programming language and the format of data files used in a computer program to interpret and execute application programs written by users and to read and write data in a specific format of data files, these are elements of that program by means of which users exploit certain functions of that program.
>
> [43] In that context, it should be made clear that, if a third party were to procure the part of the source code or the object code relating to the programming language or to the format of data files used in a computer program, and if that party were to create, with the aid of that code, similar elements in its own computer program, that conduct would be liable to constitute partial reproduction within the meaning of Article 4(a) of Directive 91/250.
>
> [44] As is, however, apparent from the order for reference, WPL did not have access to the source code of SAS Institute's program and did not carry out any decompilation of the object code of that program. By means of observing, studying and testing the behaviour of SAS Institute's program, WPL reproduced the functionality of that program by using the same programming language and the same format of data files."

253. In response to the questions posed regarding the interpretation of Article 5(3) of the Software Directive, the court stated:

> "[50] The Court observes that, from the wording of that provision, it is clear, first, that a licensee is entitled to observe, study or test the functioning of a computer program in order to determine the ideas and principles which underlie any element of the program.
>
> [51] In this respect, Article 5(3) of Directive 91/250 seeks to ensure that the ideas and principles which underlie any element of a computer program are not protected by the owner of the copyright by means of a licensing agreement.
>
> [52] That provision is therefore consistent with the basic principle laid down in Article 1(2) of Directive 91/250, pursuant

to which protection in accordance with that directive applies to the expression in any form of a computer program and ideas and principles which underlie any element of a computer program are not protected by copyright under that directive.

[53] Article 9(1) of Directive 91/250 adds, moreover, that any contractual provisions contrary to the exceptions provided for in Article 5(2) and (3) of that directive are null and void.

[54] Second, under Article 5(3) of Directive 91/250, a licensee is entitled to determine the ideas and principles which underlie any element of the computer program if he does so while performing any of the acts of loading, displaying, running, transmitting or storing that program which he is entitled to do.

[55] It follows that the determination of those ideas and principles may be carried out within the framework of the acts permitted by the licence.

[56] In addition, the 18th recital in the preamble to Directive 91/250 states that a person having a right to use a computer program should not be prevented from performing acts necessary to observe, study or test the functioning of the program, provided that these acts do not infringe the copyright in that program.

[57] As the Advocate General states in point 95 of his Opinion, the acts in question are those referred to in Article 4(a) and (b) of Directive 91/250, which sets out the exclusive rights of the rightholder to do or to authorise, and those referred to in Article 5(1) thereof, relating to the acts necessary for the use of the computer program by the lawful acquirer in accordance with its intended purpose, including for error correction.

[58] In that latter regard, the 17th recital in the preamble to Directive 91/250 states that the acts of loading and running necessary for that use may not be prohibited by contract.

[59] Consequently, the owner of the copyright in a computer program may not prevent, by relying on the licensing agreement, the person who has obtained that licence from determining the ideas and principles which underlie all the elements of that program in the case where that person carries out acts which that licence permits him to perform and the acts of loading and running necessary for the use of the computer program, and on condition that that person does not infringe the exclusive rights of the owner in that program.

[60] As regards that latter condition, Article 6(2)(c) of Directive 91/250 relating to decompilation states that decompilation does not permit the information obtained through its application to be used for the development, production or marketing of a computer

> program substantially similar in its expression, or for any other act which infringes copyright.
>
> [61] It must therefore be held that the copyright in a computer program cannot be infringed where, as in the present case, the lawful acquirer of the licence did not have access to the source code of the computer program to which that licence relates, but merely studied, observed and tested that program in order to reproduce its functionality in a second program.
>
> [62] In those circumstances, the answer to Questions 6 and 7 is that Article 5(3) of Directive 91/250 must be interpreted as meaning that a person who has obtained a copy of a computer program under a licence is entitled, without the authorisation of the owner of the copyright, to observe, study or test the functioning of that program so as to determine the ideas and principles which underlie any element of the program, in the case where that person carries out acts covered by that licence and acts of loading and running necessary for the use of the computer program, and on condition that that person does not infringe the exclusive rights of the owner of the copyright in that program."

254. In the context of a discussion concerning reproduction of the SAS manuals, the Court stated:

> "[66] In the present case, the keywords, syntax, commands and combinations of commands, options, defaults and iterations consist of words, figures or mathematical concepts which, considered in isolation, are not, as such, an intellectual creation of the author of the computer program.
>
> [67] It is only through the choice, sequence and combination of those words, figures or mathematical concepts that the author may express his creativity in an original manner and achieve a result, namely the user manual for the computer program, which is an intellectual creation…
>
> [68] It is for the national court to ascertain whether the reproduction of those elements constitutes the reproduction of the expression of the intellectual creation of the author of the user manual for the computer program at issue in the main proceedings."

255. On the basis of the guidance set out in that judgment, save for the court's finding in respect of infringement of copyright in the SAS manuals, Arnold J dismissed the claims by SAS in *SAS Institute Inc. v World Programming Ltd* [2013] EWHC 69 (Ch).

256. The Court of Appeal upheld those findings and dismissed the appeal at [2013] EWCA Civ 1482 per Lewison LJ:

"[33] What seems to me to be clear … is (a) that if expression is dictated by technical function then the criterion of originality is not satisfied; and (b) that, where that is the case, the product is not an intellectual creation of the author at all. It is of importance to note that this emphasis on questions of function applies to the Information Society Directive and not just to the Software Directive...

…

[39] In the course of the reference in our case both the Advocate-General and the court discussed the distinction between ideas and the expression of ideas. The Advocate-General began his discussion at [42] to [44] concluding at that point that the originality "of a work" lies not in an idea, but in the expression of an idea. At this point the Advocate-General was dealing with works generally, not limited to computer programs. However, at [47] to [50] he recognised that elements of creativity, skill and inventiveness manifest themselves in the way in which a program is put together; and that copyright protection for a program is conceivable from the point at which the selection and compilation of its elements are indicative of the creativity and skill of the author. He concluded that the protection of a computer program was not confined to the source code and object code but extended to any other element expressing the creativity of its author.

[40] The Advocate-General then turned to consider what counts as an idea, rather than the expression of an idea; in particular the functionality of a computer program. He defined that expression at [52] as follows: "The functionality of a computer program can be defined as the set of possibilities offered by a computer system, the actions specific to that program. In other words, the functionality of a computer program is the service which the user expects from it."

[41] He then gave an example taken from the facts in *Navitaire Inc v easyJet Airline Co Ltd* [2004] EWHC 1725 (Ch) [2006] RPC 3. In short he said that the functionalities of a computer program are dictated by a specific and limited purpose: "In this, therefore, they are similar to an idea. It is therefore legitimate for computer programs to exist which offer the same functionalities."

[42] But he added at [55] that: "There are, however, many means of achieving the concrete expression of those functionalities and it is those means which will be eligible for copyright protection under [the Software Directive]. As we have seen, creativity, skill and inventiveness manifest themselves in the way in which the program is drawn up, in its writing. The programmer uses formulae, algorithms which, as such, are excluded from

> copyright protection because they are the equivalent of the words by which the poet or the novelist creates his work of literature. However, the way in which all of these elements are arranged, like the style in which the computer program is written, will be likely to reflect the author's own intellectual creation and therefore be eligible for protection."
>
> …
>
> [74] … What is protected is the form of expression of an intellectual creation. The intellectual creation itself is not protected; and the functionality of a computer program does not count as a form of expression. The functionality of a computer program (in the sense of what it does and how it responds to particular inputs) falls on the ideas side of the line."

257. From the above authorities, the following principles relevant to the dispute in these proceedings can be drawn.

258. First, copyright protection for computer programs extends to expression of the program but not to ideas and principles underlying functionality of the program. Expression of the author's creativity may include the choice, sequence and combination of words, figures or mathematical concepts selected by the author of the program. This can be contrasted with functionality of a computer program, which is the service which the user expects from it.

259. Second, exceptions to the restricted rights of a copyright owner include observation, study and testing of the functioning of a program by a lawful user to determine its underlying ideas and principles. This allows a lawful user to observe the output of a program in response to a given input in order to determine its behaviour and reproduce the same functionality in another program. However, this does not extend to circumstances in which the user gains access to the source or object code of the computer program and reproduces the expression of the intellectual creation of the copyright owner.

260. Third, where necessary for the purpose of interoperability, a lawful user of a computer program is entitled to decompile and/or reproduce that part of the program known as the interface, described in the Software Directive as the logical and physical interconnection and interaction required to permit all elements of software and hardware to work with other software and hardware and with users in all the ways in which they are intended to function. This entitlement is not without limit, however, and is subject to the legitimate interests of the copyright owner; if the source code or object code is used to create another computer program that is substantially similar in its expression, such conduct is likely to constitute partial reproduction in breach of copyright.

261. This brings into sharp focus a dispute between the parties as to the extent to which the Software Directive grants any right to a lawful user of licensed software to reverse engineer an interface and, in that context, what is meant by an interface.

262. A useful starting point is the IBM published documentation, which provides the following definitions:

> "Application Programming Interface:
>
> A functional interface supplied by the operating system or by a separately orderable licensed program that allows an application program written in a high-level language to use specific data or functions of the operating system or the licensed program."
>
> "Customer programming interface:
>
> Any product method that lets a customer-written program obtain the services of the product (for example, CSECT names, data areas or control blocks, data sets or files, exits, macros, parameter lists, and programming languages)."

263. The defendants' position is that the effect of the decisions in *SAS v WPL* and the terms of the Software Directive is that the ICA is incapable of prohibiting or restricting Winsopia's observation, study and testing of any element of the interfaces utilised and relied upon by customer applications because such elements are unprotectable ideas and principles. They adopt a broad definition of such interfaces to include IBM CSECTs bound into customer applications from the Language Environment or the compiler, initialisation data structures generated by the compiler, the CICS EXEC interface, macros, copybooks, network communication protocols and user configuration data structures.

264. Leaving to one side the fact-specific arguments on interoperability, the defendants submit that these interfaces share a number of common features, namely: (i) the interface exists separately to, and does not itself provide, the functionality of the ICA Program that is requested by the customer application; (ii) although the interface may contain multiple elements, some of which are located in the runtime environment or middleware, the relevant part for the purposes of this dispute is limited to the interface within the compiled and/or bound object code of the customer application; (iii) the interface is an interconnection between the customer application and the runtime environment or middleware; and (iv) the interface is necessary for the customer application to run and/or to request or receive callable services from z/OS Base or from other IBM Programs upon which the customer application depends.

265. IBM's position is that there is no free-standing right in the Software Directive to reverse engineer interfaces. This is clear from the terms of both Articles 5(3) and 6, neither of which mention interfaces. Article 5(3) permits 'black-box' testing to derive the ideas and principles underlying interfaces. It does not permit decompilation, disassembly or any other 'white-box' analysis to derive information about interfaces. In practice, this means that internal component to component interfaces cannot be analysed under Article 5(3) as, at the very least, some study of disassembled code or pseudo-assembly language in compiler listings (where applicable) is required to derive insight into these interfaces. Article 6 permits decompilation or disassembly but only where this is "necessary to achieve interoperability between an independently created

program with other programs." This in practice means that the Software Directive envisages that there will be interfaces in a software product that cannot be accessed through Article 6.

266. IBM identifies interfaces for the purpose of the Software Directive as user interfaces (commands that a user can input into a software product to obtain particular results), APIs (source code statements that can be used in a program to request services from a separate program) and programming languages. IBM submits that these interfaces share a number of common features, namely: (i) they are used by program users and/or application developers to access functionality provided by a program or runtime execution environment without knowledge as to how the functionality is provided; (ii) they consist of a syntax or series of rules specifying what statements and parameters are needed to request the relevant functionality or services; and (iii) a competing product utilising the same command interfaces, programming language or APIs can be created without any knowledge or understanding of the internal implementation or the architecture of the original product utilising them.

267. Professor Weissman and Professor Donaldson agree that an operating system makes its services available to software applications via a number of interfaces. When a software application requires a service from the operating system, the software application invokes the required service via the appropriate interface. In some cases, the software application invokes the operating system service directly. In other cases, the software application calls a routine provided by the programming language runtime environment, and the runtime environment invokes an operating system service. The interface can be considered as comprising (i) code that calls, or invokes, the required service, making the connection between the customer application and the relevant API; (ii) code that declares or specifies the function required, including the name of the function, its form, parameters and constraints; and (iii) code that implements the computation(s) required, which may require calls to other library functions in order to complete the task. It is a matter of choice by the designer of the interface as to how and when each component part is activated and processed. Professor Weissman and Professor Donaldson agree that almost all non-trivial programs are built not as a single large component but as multiple separate components that are bolted together and that the connection between the components is accomplished through interfaces. This is known as modularity and is a basic principle of software design, because it allows modules to be maintained and updated separately from the customer applications provided that their interfaces remain the same.

268. In his second expert report, Professor Weissman draws a distinction between external interfaces (the perimeter of a unit of software which can be used to interact with another piece of software), say A and B, and internal interfaces (inside the perimeter of a unit of software that is a result of architectural design choices within that unit of software about how that software is internally implemented), say B and C. Professor Donaldson quibbles with this definition as confusing; he notes, correctly, that it oversimplifies the arrangements between components and does not address the possibility that the interface of component C might be available directly to A but simply not needed.

Nonetheless, the principle identified by Professor Weissman is sound, namely, that the interconnection and interaction that is required to permit the user application to work with other independently created programs (i.e. interoperability) is that between A and B. It is a matter of choice by the designer, in respect of which the user does not need any knowledge, as to whether, when and how the functionality requested by A is provided directly by B, indirectly by C and/or by any other component.

269. Thus, the relevant question in each case is not whether the interaction can be described as an interface on any level, nor how the software designer has chosen to arrange the calling code, the declaring code and the implementation code; rather, it is whether the particular code under scrutiny can be properly categorised as falling within "ideas and principles" or "expression" for the purpose of Article 5(3); or "information necessary to achieve interoperability" for the purpose of Article 6.

270. Finally, it is important to appreciate that the Software Directive is framed in very general language; no doubt, this reflects the negotiation process of a number of EU States and recognition that it would need to be applied in a variety of circumstances. Its application to any given set of facts requires careful analysis of the technical and factual context in which the issue is to be determined.

Conclusions on ICA

271. Drawing together the strands of the above analysis of the applicable legal principles against the terms and conditions of the ICA, my conclusions on contractual construction are as follows:

i) The ICA provided a licence of limited scope, permitting Winsopia to use the licensed software for the purposes of Winsopia's business, using the Designated Machine, for the execution and running of the defined ICA Programs.

ii) The definition of ICA Programs is broad, comprising machine readable instructions and data as well as components. The permission granted and restrictions imposed in respect of the use of the ICA Programs applied to the whole of each identified ICA Program and to any component part of such program.

iii) Winsopia's use of test applications or third-party customer applications would be subject to the restrictions in the ICA if they comprised or contained an ICA Program (or component thereof).

iv) Clause 4 of the ICA imposed restrictions on Winsopia's use of the licensed software; in particular, Winsopia was prohibited from reverse engineering ICA Programs, transferring ICA Programs out of its Enterprise, or copying and/or transferring IBM source code to third parties.

v) Winsopia was not prohibited from sharing or distributing its customer applications and modules with third parties, subject to the terms of material Licensed Program Specifications and/or the terms of the ICA.

vi) The use of debugging tools, whether supplied by IBM or other commercially available tools, remained subject to the terms of the ICA.

vii) Winsopia was entitled to exercise the rights of a licensee of computer programs conferred by Articles 5(1), 5(3) and/or 6 of the Software Directive and the CDPA in accordance with the principles summarised above.

viii) The terms and conditions of the ICA must be read and construed as subject to the Article 5(3) and Article 6 rights under the Software Directive.

ix) Article 5(1) contains a further exception to the restricted rights, namely, error correction, but that is subject to any terms or conditions in the ICA prohibiting such use.

x) The limitations and exceptions to the copyright protection should be interpreted restrictively having regard to the right of the copyright owner to exploit the work and the legitimate interests of the author.

272. With those principles in mind, I now turn to consider the alleged technical breaches. It should be emphasised that, in each case, it is fact-specific as to whether the activity in question constituted legitimate observation, study and testing, or decompilation for the purpose of interoperability; or amounted to a breach of the ICA.

Section V - Alleged breaches of the ICA

273. The pleaded case is that, in breach of the ICA, Winsopia used the IBM mainframe software, or permitted it to be used for the purpose of development of the SDM and/or otherwise reverse engineered parts of the IBM mainframe software. The particulars of the technical breaches served on 18 August 2023 set out examples of the alleged breaches under the following categories:

i) reverse engineering of the IBM mainframe software by disassembly, decompilation and translation;

ii) reverse engineering through the systematic creation and analysis of compiler listings;

iii) reverse engineering through the systematic use of traces, dumps, slip traps, packet sniffing and other debugging tools/techniques;

iv) copying IBM source code, macro expansions and copybooks;

v) transferring “unscrubbed” and/or partially “scrubbed” materials containing IBM Mainframe Software;

vi) further use outside Enterprise and use beyond the Designated Machine.

Disassembly, decompilation and translation

Item 1: IGZCUST binary module (Paragraph 11.1 of the Technical Particulars)

274. IBM alleges that Winsopia disassembled code in the IGZCUST module, sent it to LzLabs and Mr Moores, and failed to prevent unauthorised use of such code, in breach of clauses 4.1, 4.1.2(b), 4.1.3(a) and 4.1.3(b) of the ICA.

275. The defendants' case is that this allegation concerns an isolated historic instance of alleged disassembly for the purpose of diagnosing and correcting an error in a customer application. As such, it was permitted by the ICA, the disassembled code was not used by LzLabs and caused no loss.

276. In July and August 2014, Mr Lynch of Winsopia observed high CPU usage while testing a customer supplied program on the Winsopia mainframe. He traced the high CPU usage to a series of instructions contained within a COBOL support module called IGZCUST, a component of load module IGZCPAC.

277. IGZCPAC is an IBM supplied load module that contains a number of general COBOL library routines supplied as part of z/OS Base. It is supplied in object code only. IGZCUST handles the 'UNSTRING' statement employed in COBOL programs, decomposing a text string into discrete fields according to a specified set of delimiters.

278. Following a request by Mr Rastall of Winsopia to document his findings, on 22 August 2014 Mr Lynch sent an email to Mr Rastall, stating:

> "When we were initially testing program from [a customer] we thought it might be infinitely looping due to its high CPU consumption and low I/O rate.
>
> So I took a series of dumps while it was running to try to discover if this was true or not. This is the results of the dump analysis.
>
> … this cluster of instructions are … part of a COBOL support module called IGZCUST which is a component of load module IGZCPAC. As a loaded program IGZCPAC was not supplied to us by [the customer] but was loaded from the COBOL Language Environment libraries on our z/OS 1.13 system. IBM describe IGZCUST as 'UNSTRING VERB LIBRARY SUB-ROUTINE'.
>
> In order to understand further what the program was doing in this cluster of instructions it was necessary to disassemble the code.
>
> Analysis of this code shows that it is a search loop where the following registers and storage are being used …
>
> I have also worked back along the save area chain to find the point where [the customer application] calls IGZCUST. I'm still

> trying to determine what parameters are passed in the call but this is proving rather difficult at the moment."

279. Mr Lynch included in his email an analysis of four dumps taken during the execution of the program, line-by-line annotation of the disassembled code and partial extracts from each of the four dumps taken.

280. Mr Rastall forwarded the email to Mr Galewsky and Mr Rockmann of LzLabs, and to Mr Moores, who provided comments by return email:

> "This thing EXecutes a TRT? Ask JJ about the performance implications of a TRT. **Massively slow** … My guess is that stuff like this is gonna have to be rewritten in C or C++."

281. The SDM Git repository commit history shows that development of the SDM implementation of the functionality of IGZCUST was started in February 2012, prior to this incident of disassembly. It also shows that, following Mr Lynch's email, 15 further commits were made to the relevant SDM code between 3 September 2014 and 3 October 2014 by Mr Bowler of LzLabs.

282. The experts' second joint statement includes the following agreements:

i) Winsopia produced dumps of an application program received from a customer to diagnose a performance issue occurring during its execution on the Winsopia z/OS system.

ii) Information in the dumps included the object code of IGZCUST, a runtime CSECT in the SCEERUN library provided by IBM to Winsopia with z/OS.

iii) Mr Lynch disassembled a portion (approximately 35 instructions) of IGZCUST for problem determination purposes.

iv) This portion of disassembled code was not the basis of nor sufficient for creation of the entire SDM IGZCUST module.

v) This portion of disassembled code was sent by e-mail to LzLabs.

vi) In reply to the email, Mr Moores stated that changes would be required to the pre-existing SDM equivalent module.

vii) Shortly after the email exchanges, the SDM module was modified.

viii) The SDM file does not reproduce source or object code from the IBM IGZCUST module.

283. It is common ground that Winsopia disassembled part of the IGZCUST module and sent the disassembled code to LzLabs and Mr Moores. The issues are:

i) whether the portion of IGZCUST disassembled was an ICA Program within the meaning of the ICA;

ii) whether the disassembly carried out by Mr Lynch was permitted error correction, on a true construction of the ICA and/or under Article 5(1) of the Software Directive, or in breach of clause 4.1.3(a) of the ICA;

iii) whether Mr Lynch's actions fell within permitted observation, study and testing of the IGZCUST module pursuant to Article 5(3) of the Software Directive;

iv) whether Winsopia's transfer out of the IGZCUST code to LzLabs and Mr Moores constituted breach of clause 4.1.3(b) of the ICA;

v) whether Winsopia permitted LzLabs to use the disassembled code in breach of clauses 4.1 and/or 4.1.2(b) of the ICA.

284. The defendants' case is that the portion of IGZCUST that was disassembled is not an ICA Program because it is merely a part of the z/OS Base V1 product. I reject that argument for the reasons set out above, namely, that component parts of a program fall within the definition of an ICA Program for the purpose of the ICA. The experts agreed that the disassembled code included the object code of the IGZCUST runtime CSECT that formed part of the SCEERUN library provided with z/OS.

285. It is said by the defendants that on its true construction the ICA does not contain any provisions specifically prohibiting error correction. Therefore, Winsopia was entitled to disassemble the code for the purpose of diagnosing or correcting errors in a customer program, pursuant to Winsopia's rights under the Software Directive.

286. The ICA does not contain an express prohibition against error correction but Clause 4.1.3(a) of the ICA contains an express prohibition against reverse assembly, reverse compilation, translation or reverse engineering of an ICA Program. The experts agree that Mr Lynch carried out disassembly and Mr Lynch stated in his email that he disassembled the code. Clause 4.1.1(d) contains specific and limited permission for Winsopia to use any portion of an ICA Program in source form for resolving problems related to use of the ICA Program but this did not extend to object code. The portion of the IGZCUST module disassembled by Mr Lynch was in object code. It follows that the ICA contained provisions whereby Winsopia was prohibited from carrying out disassembly of object code for any purpose, including error correction.

287. Article 5(1) of the Software Directive provides that:

> "In the absence of specific contractual provisions, the acts referred to in points (a) and (b) of Article 4(1) shall not require authorisation by the rightholder where they are necessary for the use of the computer program by the lawful acquirer in accordance with its intended purpose, including for error correction."

288. This does not assist the defendants. The permission to use the program for error correction is expressly subject to *"specific contractual provisions"*. The ICA

contains a specific contractual provision that prohibits disassembly (subject to the exception permitting the use of source code for error correction). Article 8 of the Software Directive, precluding the use of contractual terms to override the rights conferred by the Directive, does not apply to Article 5(1).

289. In any event, in cross-examination, Mr Moores stated that the purpose of this testing on the Winsopia mainframe was to improve performance on the SDM. Therefore, it was not carried out to correct any error in the ICA Program or the customer application.

290. The defendants submit that Mr Lynch's actions fell within permitted observation, study and testing of the IGZCUST module pursuant to Article 5(3) of the Software Directive. This requires consideration of what Mr Lynch did and for what purpose.

291. In his first witness statement, Mr Lynch stated that he used XDC to carry out the disassembly of the code:

> "I used XDC to examine the cluster of instructions in the IGZCUST module to identify whether there was a bug in the system which might explain why it was causing the customer's application to run slowly, or whether the slow running and high CPU usage were caused by something else. XDC displays the code in disassembled form by default; this is essentially its primary user interface. "

292. As explained earlier in this Judgment, XDC is a commercially available debugging tool that enables a breakpoint to be set at a specific point in a load module, stopping execution and transferring control to XDC. It allows the user to step through the code examining the execution of each instruction, view data areas and registers, disassemble object code and display dumps of memory.

293. Mr Lynch's explanation that he used XDC to carry out this disassembly was echoed by Mr Rastall, who stated in his second witness statement:

> "Mr Lynch was using standard debugging facilities to identify the source of the problem. To be clear, Winsopia (including Mr Lynch) used the XDC tool in the way it was designed to be used."

294. In his third witness statement, Mr Lynch stated that, on further reflection, he could not be sure that he used XDC, although he did not know for certain that he did not use XDC in connection with this exercise. This uncertainty appears to have arisen as a result of an email exchange between Mr Palmer and Colesoft (suppliers of the XDC tool) in October 2014, indicating that XDC was not installed at Winsopia prior to that date. In cross-examination, Mr Lynch stated that he thought that he manually disassembled the code but could have used an alternative IPCS disassembly tool supplied by IBM. When pressed, he conceded that he had no recollection as to what he did.

295. In the light of this additional evidence, Mr Stephens' opinion was that Mr Lynch could have reviewed the dumps manually using a tool like IPCS. He noted that there were two errors in the disassembled code and commentary by Mr Lynch, which could point to manual disassembly as opposed to use of a disassembly tool such as XDC. However, in cross-examination he accepted that Mr Lynch could simply have made those errors when transcribing into the email the output generated by a disassembly tool.

296. Although the state of the evidence on this matter is unsatisfactory, it is not material to the issues in dispute. Regardless of whether the method of disassembly was manual or through use of a tool, Mr Lynch accepted that he disassembled part of the IGZCUST load module. Such disassembly involved translating parts of the module object code into assembly code. Mr Lynch did not confine his actions to observation, study and testing of the functioning of the IGZCUST module, which could be achieved simply by running the customer application and examining the input and output. On the contrary, he carried out a detailed analysis of the execution of each instruction within the disassembled portion of the module. Such detailed analysis investigated, not just the output of the program but how the program achieved its output, that is, expression of the program, rather than its functioning.

297. There is a dispute between the experts as to the amount of code disassembled. In the email sent by Mr Lynch, there were 35 disassembled instructions, out of about 1,369 lines of code. Mr Stephens and Professor Donaldson estimated that this represented about 2% of the IGZCUST module. However, Mr Swanson noted that there was an 'offset gap' between the penultimate and final lines of code in the email, suggesting that Mr Lynch must have disassembled substantially more lines of code than the areas of interest reproduced in the email. Mr Stephens agreed that more of the module must have been disassembled. He considered that it was likely that Mr Lynch disassembled the code above and around the TRT statement but it did not necessarily follow that he disassembled all the code between the penultimate and final lines of code in the email.

298. In cross-examination, Mr Lynch stated that he would have targeted his disassembly on the instructions of interest but this was qualified by his explanation that:

> "Once I had the disassembled code, then I know – I know where to look, and I know where not to look, because I'm not interested in it. "

299. This does not assist in answering the question how much code was disassembled before he knew where to look and could target the relevant instructions. As Mr Lynch accepted, he did not recall what he did. Regardless of the amount of code actually disassembled, none of the experts suggested that it was *de minimis* or superfluous code. The substantive quantity of disassembled code is evident from the extracts set out in Mr Lynch's email. Indeed, Mr Stephens' opinion was that all code was of more or less equal importance, observing that if it was unimportant, it would not be in the program at all.

300. I find that it is likely that a substantial amount of functional code was disassembled, the purpose of which was to discover the specific form, parameters, sequence and effect of each instruction. Such disassembly was not permitted by the terms of the ICA and did not fall with the observation, study and testing exception of the Software Directive.

301. It is not disputed that the disassembled code was sent by Mr Rastall of Winsopia to LzLabs and to Mr Moores. This amounted to a transfer of an ICA Program outside Winsopia's Enterprise in breach of clause 4.1.3(b) of the ICA. This was accepted in cross-examination by Mr Rockmann.

302. The defendants submit that no use was made of the disassembled portion of IGZCUST by LzLabs. The experts agreed that the portion of disassembled code sent to LzLabs was not the basis of nor sufficient for creation of the entire SDM IGZCUST module, which was developed some considerable time before August 2014 and included a functional UNSTRING routine. However, IBM's case is not that the disassembled code was used to create the SDM module but rather that it was used to make improvements to the same.

303. Support for IBM's case can be found in the proximity of time between Mr Lynch's email of 22 August 2014 and the Git repository commits by Mr Bowler during the period 3 September to 3 October 2014. First, the Git repository commit made on 8 September 2014 substituted a standard C library function with a bespoke searching function that would avoid unnecessary conversions between ASCII and EBCDIC (different representations of textual data in computer memory). Mr Swanson's opinion was that this change provided an equivalent function to the code analysed in Mr Lynch's email. Professor Donaldson agreed that the change made was a substantive change, in that it would make the SDM more accurate functionally and it would improve performance of the SDM code by avoiding the unnecessary conversions. However, his opinion was that, although this could lead to improved performance, copying the logic of such code would be unnecessary, as the search function in question could be easily written from scratch.

304. Second, on 12 September 2014, although he did not modify the existing code, Mr Bowler added an explanatory comment regarding interpretation of the parameters that were passed to IGZCUST. There is no explanation as to the source of this information, which is not published by IBM.

305. Third, on 3 October 2014, a commit was made, introducing new data types that described the structure of arguments to IGZCUST. Professor Donaldson considered that this was the most significant commit made during this period, adding the most substantive functionality. He was unable to identify any source code comments indicating what led to this improved understanding of IGZCUST parameters. Although this was the focus of part of Mr Lynch's work, as evidenced by the last paragraph of his email dated 22 August 2014, Mr Swanson and Professor Donaldson agreed that these improvements could not be attributed to any information set out in the email.

306. Unfortunately, the DR process was not used for this aspect of Winsopia's work on the mainframe and there is no record of any discussions between LzLabs and

Winsopia. Mr Rastall agreed in cross-examination that the direct communication between Winsopia and LzLabs was in breach of the Code of Conduct then in force. If the DR process had been used, there would have been a clear audit trail of information requested and received by LzLabs and the disassembled code would or should have been redacted.

307. Mr Jaeger's evidence was that he started to investigate performance issues in the SDM module from early August 2014 and made changes to improve the runtime speed of the customer application in question which were incorporated into the Git repository on 18 August 2014, before Mr Lynch's email of 22 August 2014. He was adamant that he did not use the analysis provided by Mr Lynch or the suggestions by Mr Moores in development of the SDM code. Mr Jaeger also stated that subsequently, Mr Bowler began looking for other ways to further improve the performance of the program. He asserts that there was no connection between the specific changes made by Mr Bowler and any of the material in Mr Lynch's email but he does not have direct knowledge as to the reasons for Mr Bowler's revisions, there was no explanation for the revisions in any contemporaneous records and Mr Bowler did not give evidence.

308. Mr Bowler's commits to the Git repository during the period 3 September to 3 October 2014 referenced a Jira ticket RC-2805, which was created by him on 2 September 2014 with the title "IGZCUST performance improvement" and referred to the time taken for the UNSTRING operations in respect of the customer application. In the comment section entry dated 19 September 2014, Mr Bowler recorded that the modification to the SDM code, eliminating the conversion between ASCII and EBCDIC, resulted in approximately 20% reduction in elapsed time.

309. Having regard to the timing and content of Mr Bowler's comments and commits, and in the absence of any direct evidence from Mr Bowler, I find that it is very likely that Mr Lynch's disassembled code and analysis was used to give LzLabs insight into how this part of the IGZCUST module worked and that it informed at least some of the changes to the SDM module to improve functionality and performance.

310. In summary for the reasons set out above:

    i) The portion of IGZCUST disassembled was an ICA Program within the meaning of the ICA.

    ii) On a true construction of the ICA, the disassembly carried out by Mr Lynch was not permitted error correction or as provided under Article 5(1) of the Software Directive but amounted to a breach of clause 4.1.3(a) of the ICA.

    iii) The disassembly did not fall within permitted observation, study and testing of the IGZCUST module pursuant to Article 5(3) of the Software Directive.

    iv) Winsopia's transfer out of part of the IGZCUST code to LzLabs constituted breach of or 4.1.3(b) of the ICA.

v) Winsopia permitted misuse of at least part of the IGZCUST code by LzLabs so as to constitute breach of clauses 4.1 and/or 4.1.2(b) of the ICA.

Item 2: Load Module Decompiler ("the LMD") (Paragraph 11.2 of the Technical Particulars)

311. The Load Module Decompiler ("the LMD") was a batch utility that was intended to supply an alternative migration path to the SDM for customer load modules that were compiled and link-edited for an IBM mainframe. The LMD was designed to transform the load modules into C language source code macros that could be re-compiled by the Load Module Compiler ("the LMC") to run natively on x86 systems, including the SDM.

312. IBM's case is that, in breach of clause 4.1.3(a) of the ICA, Winsopia used the LMD to decompile and translate, into C language macros, modules and code fragments which constituted ICA Programs. Although a scrubbing process was introduced, it was ineffective in preventing IBM CSECTs or other IBM modules from being processed by the LMD.

313. The defendants' case is that the modules and code fragments did not constitute ICA Programs. The LMD was not a decompiler (despite its name) and it did not carry out reverse compilation, reverse assembly or reverse translation. It was designed to filter out IBM proprietary material and there is no evidence that it was ineffective in doing so. Alternatively, it was necessary in order to achieve interoperability of the customer applications with the SDM and was permitted pursuant to Article 6 of the Software Directive.

314. From 2015 Tom Grieve, a Winsopia developer, worked on developing the LMD, whilst Mr Jaeger worked on the LMC at LzLabs.

315. Mr Grieve explained in the document attached to his first witness statement that the LMD was designed to convert a z/Architecture load module into C language source code, generally by taking each machine instruction in turn and creating a corresponding C macro.

316. LzLabs developed the set of C language source code macros, based upon Winsopia's analysis of the object code in COBOL load modules, using compiler listings and XDC, as set out in Mr Lynch's COBOL object code analysis reports sent to Mr Rockmann and Mr Cresswell at LzLabs on 1 April 2016.

317. The LMD used the binder APIs to load the load module into storage for inspection on the Winsopia mainframe and identify the entry point in the program for each CSECT. The LMD processed each of the binary machine code instructions in the load modules in sequence, by transforming them into assembly language mnemonics. It then converted the resulting sequence of assembly language mnemonics into a matching sequence of C language macro calls, one for each type and format of mnemonic. The LMC took the output of the LMD and used the C language macros to create a new executable load module that mimicked the sequence of machine instructions used in the original load module.

318. There were three versions of the LMD:

    i) LMD v1 developed between July 2015 and August 2016, to support load modules with COBOL version 4;

    ii) LMD v2 developed between January 2016 and April 2018, to support PL/I version 4 and COBOL version 5;

    iii) LMD v3 developed between April 2018 and March 2020, to support PL/I version 5 and COBOL version 6.

319. In 2016 the LMD was integrated into CPX as a diagnostic tool and was made available to customers as part of the CPX package. Ultimately, the LMD/LMC project was abandoned, following development and completion of the SDM.

320. The experts' second joint statement includes the following agreed matters:

    i) Winsopia developed a z/OS based tool called the LMD that processed load modules, producing C source code.

    ii) The C source code could be processed by another SDM tool called the LMC which allowed the LMD output to be compiled and executed on a non-mainframe computer.

    iii) The LMD included processing to exclude certain IBM supplied CSECTs.

    iv) The LMD did not exclude processing of code inserted by IBM compilers into a user CSECT.

    v) The LMD was removed in 2020.

321. The disputed issues are as follows:

    i) whether the LMD processed ICA Programs and/or filtered out IBM CSECTs and other code fragments;

    ii) whether the proper characterisation of the LMD process amounted to decompilation, disassembly and/or translation of ICA Programs in breach of clause 4.1.3(a) of the ICA;

    iii) whether the LMD process was necessary in order to achieve interoperability of customer applications with the LMC and SDM and, as such, was permitted by Article 6 of the Software Directive.

322. The defendants' case is that filtering rules built into the LMD and additional checks prevented the LMD from operating on IBM CSECTs. Mr Grieve's evidence was that code in the LMD used two independent checks to implement the filter. First, it checked the name of the CSECT against a list of prefixes, using wildcards ("the Exclusion List"). A second-stage filter checked the CSECT metadata of non-excluded CSECTs to confirm whether any CSECT was produced by a compiler other than the COBOL v4 compiler ("the Compiler

Test"). If either the Compiler Test or the Exclusion List resulted in a match, the CSECT would be skipped over and not processed by the LMD.

323. Mr Stephens elaborated on the system adopted in his second report. In version 1 of the LMD, CSECTs were excluded by comparing the first three characters of the CSECT name with known IBM module name prefixes. Professor Weissman pointed out in his third report that, as at 13 April 2016 in version 1 of the LMD, the exclusion prefix list was limited to eight IBM CSECT prefixes. Mr Stephens explained that, in version 2 of the LMD, the system was enhanced to compare CSECT names with known IBM prefixes of different lengths, using wildcards.

324. Mr Grieve confirmed in cross-examination that wild cards were not introduced until version 2. Therefore, if the CSECT did not match any of the initial Exclusion List, version 1 of the program would process the CSECT into the corresponding C language instruction. Mr Grieve stated that the Exclusion List was expanded on an incremental basis, as and when it was discovered that the LMD had processed, or attempted to process, IBM CSECTs, such as the CSQ CSECT, which was discovered by Mr Grieve in December 2016 as set out in his email dated 8 December 2016.

325. Mr Grieve explained that he carried out a further manual check, reviewing the output of the LMD on a line by line basis but in cross-examination he clarified that of the thousands of lines of code produced, he checked probably the first half dozen files.

326. Mr Grieve accepted in cross-examination that the LMD would process and decompile initialisation code inserted into a user CSECT by the IBM COBOL compiler. For the reasons explained above, I find that such code amounted to an ICA Program within the meaning of the ICA.

327. Mr Jaeger explained that the purpose of the LMD/LMC was to remove and replace IBM stub CSECTs. Even if a stub CSECT inadvertently was not excluded, so that it was processed into a C language macro by the LMD and compiled by the LMC, it would not be possible for the LMC to execute it, as it would point to an address in the IBM runtime library that would not exist outside the IBM mainframe. That is no doubt correct but it does not detract from the weight of evidence that the LMD was not programmed to exclude all IBM CSECTs and other code fragments.

328. In his second report Mr Stephens analysed the Compiler Test and concluded that it would be impossible that IBM CSECTs would be processed by version 1 of the LMD (or versions 2 and 3 with the appropriate flag set), on the assumption that no IBM CSECTs were written in COBOL or PL/I. However, Professor Weissman examined that issue in his third report and identified a number of IBM supplied modules that were written in COBOL. On that basis, he concluded that the Compiler Test would not have been effective at removing all IBM supplied CSECTs which were written in COBOL.

329. In the light of the admitted limitations of the initial filtering system implemented by the LMD, although it improved over time, it is highly likely that the LMD processed IBM CSECTs and other code fragments.

330. It is not disputed by the defendants that there was transformation of the load module machine code into C language macros. Clearly, this amounted to translation (from one computer code to another) on a natural and obvious meaning of the word.

331. The defendants contend that the proper characterisation of what the LMD did was not reverse compilation or reverse assembly.

332. There is a dispute between the experts as to the definition of "disassembly". Professor Weissman defines disassembly as "the conversion of binary object code into the equivalent human-readable assembly instructions." IBM assembler language is created in the LMD process implicitly as part of the conversion to C language code, in that an assembly operator must be first determined from the object code representation before it can be mapped to a C macro. Professor Donaldson defines disassembly as "disassembling all or part of the program into its assembler language source statements and then trying to understand the resulting assembly code". On the basis that the assembler language was an intermediate step and not intended to be read, it did not amount to disassembly.

333. Professor Weissman described the LMD process in his first report as follows:

> "LMD takes the original load module and performs a disassembly on the machine-code instructions in sequence one by one (as described by Tom Grieve in his First Witness Statement). That is, it performs an automatic transformation from the load module's machine-code (sequences of executable hexadecimal codes), into assembly language (human-readable) mnemonics. This process produces a 1:1 mapping between machine-code representation and assembly language representation, which can be performed in either direction …
>
> LMD disassembles the load module and decompiles the resulting sequence of assembly language mnemonics into a matching sequence of C-language macro calls, one for each type and format of mnemonic. The resulting C-language program therefore appears as a linear set of C-language macro calls, directly tracking the assembly-language representation of the load-module…
>
> … the load module machine-code instruction … is, component by component, disassembled and decompiled by LMD."

334. Although he disagreed with the description of the operation as reverse assembly, Mr Stephens agreed with the above summary in his report.

335. In cross-examination, Mr Grieve accepted that the LMD process involved disassembly before creation of the relevant C macros:

> "A. It's -- it's translating 390 machine instructions to C macros.
>
> Q. Yes, but … it does that by disassembling and obtaining the disassembled machine instructions in assembly language, doesn't it?
>
> A. It doesn't disassemble. It doesn't try and produce Assembler source from the object code. It produces C language macros, which is not the same thing at all.
>
> Q. No, but the C language macros correspond to the assembly language, don't they?
>
> A. Yeah, mostly one to one.
>
> Q. Yes. So, as it were, the assembly language is an intermediate step in the production of the C language macros, isn't it?
>
> A. Well, it's the input. It's not -- it's not an intermediate step.
>
> Q. Well, so -- yes, I mean, the macro generator, if that's the correct term, couldn't have operated unless it was fed an input, could it?
>
> A. Yes, that's correct.
>
> Q. And that formed -- that took the form of IBM assembly language instructions?
>
> A. That's correct, yeah.
>
> Q. And those were obtained by disassembly?
>
> A. Yes, okay."

336. In August 2018 Mr Moores asked Mr Jaeger about the way in which the LMC could compile an unconditional branch instruction. Mr Jaeger explained:

> "We take care of the branch in the LMD part of LMC.
>
> It disassembles the code, and say if the branch is B 20(R11), then it will look where Rl1 was previously loaded, where it was loaded from and what the contents was. So, it effectively disassembles it back to B Label, where we generate a GOTO In C …"

337. It is clear from the above evidence that the load module object code, including IBM CSECTs and other IBM code, was disassembled by the LMD as part of the process to translate the object code into C language macros.

338. Professor Weissman states his understanding that "decompilation" is the conversion of object code into any high level language, not necessarily the language from which the object code was originally compiled. The output of decompilation is a program that is in a human readable higher level language than the object code and itself can be compiled. He does not consider that decompilation is restricted to recovery of code written in the original programming language but includes the recovery of code in a different programming language that performs the same function as the machine code from which it was derived as part of the decompilation process. Professor Donaldson notes that the LMD is different from a traditional decompiler because it does not produce high-level language code that can be understood by human developers, but rather rewrites machine instructions into a corresponding C-macro form.

339. The fact that the high-level language code produced as the output of the LMD was used for an unconventional purpose does not affect the analysis of the process involved, which was self-evidently decompilation of compiled object code into high-level language. This was expressly recognised in the CPX Guide drafted by Mr Palmer, in which he described the LMD as follows:

> "The Load Module Decompiler (LMD) is a loosely coupled independent LzLabs component that is shipped with the Centerpiece Export (CPX) software by default, where it provides the ability to de-compile a load module into a format that can be executed natively on the SDM Linux operating system offering significantly improved performance."

340. It follows that the LMD process amounted to disassembly, decompilation and translation of the load module, including IBM CSECTs and other IBM modules that were not filtered out.

341. The defendants submit that the LMD processing was necessary in order to achieve the interoperability of customer applications with the LMC tool and the replacement x86 runtime environment. I reject that submission for the following reasons.

342. Firstly, the Article 6 exception provides that the authorisation of the rightholder is not required where reproduction of the code and translation of its form are indispensable to obtain the information necessary to achieve interoperability. Decompilation and translation of the load modules by the LMD was not carried out to obtain information to facilitate interoperability with the LMC; rather, it was carried out to transfer every machine code instruction in the entire load module (save for the excluded IBM CSECTs), in the form of C language macros, to the LMC.

343. Secondly, the LMC was not an independently created program. As set out in Mr Lynch's reports, the C language macros were part of the LMD/LMC project and were developed through analysis of the COBOL object code using compiler listings and XDC.

344. Thirdly, Article 6(2)(c) provides that information obtained by reproduction of the code and translation of its form is not permitted to be used for the development, production or marketing of a computer program substantially similar in its expression, or for any other act which infringes copyright. Development of the LMD/LMC was intended to provide an alternative program to execute the load module in an x86 environment by mapping all machine code instructions in the load module to C language macros in sequence on a one to one basis. This included code that was substantially similar in expression to the IBM CSECTs and other IBM code in the load module.

345. In summary, on this issue:

    i) The LMD processed ICA Programs. Although different versions of the LMD filtered out some IBM CSECTs, it was not effective to identify and exclude all IBM CSECTs and other IBM code inserted into the load module through the compilation and link-editing process.

    ii) The LMD process amounted to decompilation, disassembly and/or translation of ICA Programs in breach of clause 4.1.3(a) of the ICA.

    iii) The LMD process was not necessary in order to achieve interoperability between customer applications and the SDM and Article 6 of the Software Directive does not provide a defence to this allegation.

Item 3: CICS Control Blocks Document (Paragraph 11.3 of the Technical Particulars)

346. The allegation is that Winsopia used reverse engineering of the CICS Transaction Server for z/OS to produce a document entitled "CICS Commands and Parameters" Version 1.3 dated 26 February 2020 ("the CICS Control Blocks Document") and provided it to LzLabs in breach of clauses 4.1, 4.1.2(b), 4.1.3(a) and 4.1.3(b) of the ICA.

347. The defendants' case is that EXEC CICS and the Arg0 parameter analysed by Winsopia do not fall within the definition of an ICA Program; they formed part of an interface for the CICS Transaction Server program and were not subject to the restrictions in the ICA. Further, Winsopia was entitled to observe, study and test the EXEC CICS interface pursuant to Article 5(3) of the Software Directive. Alternatively, it was necessary in order to achieve interoperability of customer applications with the SDM and was permitted pursuant to Article 6 of the Software Directive.

348. As set out in Section II of this Judgment, EXEC CICS commands can be incorporated into the source code of an application hosted on CICS, enabling it to request specific services from the CICS Transaction Server and underlying components of the z/OS operating system at runtime. EXEC CICS commands are pre-processed by the CICS translator before compilation and link-editing.

349. The following is a simplified summary of the EXEC CICS process.

    i) A full list of EXEC CICS commands, with their functions, are set out in the CICS Application Programming Reference manual.

ii) The application programmer selects the required EXEC CICS command from the manual options and includes it in the customer source code.

iii) When an EXEC CICS command is incorporated into an application, the language-specific CICS translator creates a new source code program, converting the EXEC CICS command into a CALL statement to a CICS routine, DFHEI1, generating a hexadecimal string of bytes, organised in arguments beginning with Arg0, which forms the parameter list required for execution.

iv) The unique Arg0 format is generated by the CICS translator, based on the specified request parameters in the EXEC CICS command, using the mapping defined in the Language Definition Tables.

v) The application, including the CALL statement, is then compiled into object code and link-edited.

vi) The CALL statement invokes a CICS module or stub ("CICS stub") to call the EXEC CICS interface. The values of Arg0 dictate the design of the CICS stub so that it is compatible with the language of the application program and the location where the program is loaded in memory.

vii) At runtime, a component of CICS reads the load module, including Arg0, which it uses to call the correct EXEC interface program, resulting in execution of the relevant CICS service request.

350. The functions invoked by each EXEC CICS command are listed in the CICS Application Programming Guide manual. However, save for limited samples, details of Arg0 are not published.

351. The CICS Control Blocks Document produced by Winsopia contains a detailed description of hundreds of EXEC CICS commands, identifying the format of Arg0 for each command, including its length in bytes and a table setting out the meaning of each bit or byte in Arg0 for the command.

352. The CICS Control Blocks document was first produced by John Horswill at Winsopia on 23 December 2013. Version 1.0 was completed on 12 November 2015. Subsequently the document was updated by Kevin Hitchings incrementally through until version 1.3 dated 26 February 2020.

353. The document was produced by running test programs containing EXEC CICS commands through the CICS translator and documenting the outputs in a word document. On 31 October 2013, Mr Horswill wrote to Mr Rastall explaining the process used:

> "The input file … is written from the syntax diagrams in the APR.
>
> This is transferred to the mainframe [and] a job is run using the translator. This generates the output which shows the relevant hex codes next to each of the commands.

> From this I generate the two word documents, one contains the HEX codes for each of the commands and the other is a summary of the commands for each control block.
>
> Besides the CICS Application Programming Reference version 5.1, I am referring to the CICS Customization Guide version 5.1 …"

354. Mr Horswill's methodology was to analyse the string of bytes produced by the CICS translator for each CICS command, correlating the meaning of the values generated by the CICS translator against values specified in the source code program, so as to derive an understanding of the DFHEI1 parameter list for the full range of CICS commands.

355. The information in the CICS Control Blocks document was sent to LzLabs and used by LzLabs in development of the SDM.

356. The experts' second joint statement contains the following agreed facts:

i) CICS provides a large number of APIs enabling a program to access CICS services. User programs use the EXEC CICS command to request these services.

ii) Before compiling a CICS program, the EXEC CICS commands are translated to source code in the same programming language as the original program using the CICS translator.

iii) This translation step converts the EXEC command into a string of bytes, organised in arguments.

iv) Argument 0 ("Arg0") is required for every CICS command and contains details about the CICS service requested. IBM has published some information on Arg0 values but has not published a full specification corresponding to each of the available EXEC CICS commands.

v) The CICS control blocks document created by Winsopia is over 1000 pages and contains a detailed description of over 500 different EXEC CICS commands, including Arg0.

vi) Information used to create the document included IBM documentation and viewing the output of the CICS translator showing the source code generated.

vii) LzLabs used this information for developing the SDM.

viii) The level of detail in the CICS Control Blocks document is at a level of granularity which is far more detailed than that published by IBM.

357. The disputed issues are:

i) whether Winsopia's analysis of the CICS translator, Arg0 and/or the Language Definition Tables was in respect of an ICA Program within the meaning of the ICA;

ii) whether Winsopia's analysis amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA;

iii) whether Winsopia's analysis fell within permitted observation, study and testing of the EXEC CICS interface pursuant to Article 5(3) of the Software Directive;

iv) whether Winsopia's analysis was necessary in order to achieve interoperability of customer applications with the SDM and, as such, was permitted by Article 6 of the Software Directive;

v) whether Winsopia's supply of the CICS Control Blocks document to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

358. IBM's case is that the CICS translator, Arg0 and/or the Language Definition Tables are component parts of the CICS Transaction Server, an ICA Program. IBM submits that Arg0 is part of the architecture of CICS Transaction Server for z/OS and EXEC CICS commands are features of applications written to be hosted on this product. The CICS Language Definition Tables that contain the mapping from EXEC CICS for Arg0 and the various CICS routines that parse and process Arg0 in order to deal with CICS requests are supplied with, and form part of, the CICS Transaction Server for z/OS.

359. The defendants' case is that the format of the Arg0 parameter is part of the EXEC CICS interface between the customer application and the CICS Transaction Server, comprising: (i) the EXEC CICS statement in the customer application; (ii) the translated call to DFHEI1 generated by the CICS translator; (iii) the CICS language interface module (CICS stub) inserted into the link-edited machine code; and (iv) the interface program which receives the call from the CICS stub at runtime, providing an entry point to the application interface program load module. The EXEC CICS interface is distinct from the CICS service that is requested by the application and, as such, it is not an ICA Program.

360. IBM publishes a list of available EXEC CICS commands from which a customer can select a specific command to incorporate in its application for the purpose of requesting a CICS service at execution. The process of translating the EXEC CICS command into the source code language of the application and generating the CALL to DFHEI1 with the Arg parameter list, including Arg0, is carried out by the CICS translator (which can be embedded within the compiler) using the Language Definition Tables. The CICS translator and the Language Definition Tables are component parts of the CICS Transaction Server for z/OS. For the reasons set out above, such component parts fall within the definition of an ICA Program for the purposes of the ICA.

361. It is said by the defendants that the CICS translator process occurs at the pre-compilation stage whilst the application is in source code form. That is correct (save where an embedded CICS translator is used) but it does not change the characterisation of the process. The process is the generation of code, including the Arg0 parameter, by component parts of the CICS Transaction Server.

362. It is also said by the defendants that the EXEC CICS interface is not part of the CICS service in that it merely generates the call and parameter list, including Arg0, to the service in the appropriate source code language. That is an oversimplification of the process, which involves complex algorithms with many variables, selected by the CICS translator by reference to the Language Definition Tables. The specific content, parameters, sequence and combination of the instructions are dictated by software which is the intellectual creation of IBM Corp (licensed to IBM and to Winsopia through the ICA). Although this process takes place prior to the call to the required CICS service, it is an essential part of the operation of the CICS Transaction Server.

363. IBM's case is that the defendants derived most of their information about Arg0 not from public sources but through use of Winsopia's mainframe, involving the systematic analysis of outputs of the CICS translator, amounting to reverse engineering.

364. The defendants' case is that its analysis was limited to the EXEC CICS interface; it did not investigate how the CICS middleware processes the API commands, or the underlying processing within the runtime environment to deliver the CICS service requested.

365. Mr Stephens described the process used to create the CICS Control Blocks document in his first report. Winsopia wrote test applications which invoked each of the relevant CICS commands. This involved invoking the IBM compiler with the test applications as input and using standard compiler options to cause the listings to be created as part of the compilation process. Winsopia compiled the test applications on its mainframe and reviewed the compiler listings created by the compiler. From a review of the compiler listings, together with publicly available documentation, the information in the CICS Control Blocks document was created.

366. In Appendix 4 to his first report, Mr Stephens explained the analysis that Winsopia could carry out based on the compiler listings:

    i) The high level language statements generated by the CICS translator would indicate what instructions were passed to the CICS interface.

    ii) The hexadecimal string of bytes would indicate the call command to DFHEI1, together with the parameters to be passed.

    iii) The response code returned by the EXEC CICS command would indicate whether the command was successful for the application in various states and, if not, information on the error.

367. It is clear from the above description that the test programs were designed to produce a compiler listing which would disclose for each test case the hexadecimal code generated by the CICS translator and the parameters determined by the Language Definition Tables. As such, it amounted to reverse engineering of the CICS translator and the Language Definition Tables, component parts of the CICS Transaction Server, in breach of clause 4.1.3(a) of the ICA.

368. LzLabs admitted that reverse engineering was used to create the CICS Control Blocks document. On 22 June 2014 Mr Taylor of LzLabs sent an email to Steve Towns and Mr Broussard at Texas Wormhole regarding two DRs, including the following:

> "…the requests need to be modified further. I have talked this over with Ira. You need to find a way to ask them to discover the API and surface it in their own words. Not just ask them to send you listings or write what they see in the listings.
>
> … the second request is that someone please continue the work started by John Horswill. His API document is excellent and it details most of the information we need to know. I don't want to wait for this however. The document has information on HANDLE ABEND but it is incomplete and does not answer this question.
>
> On DR0068 you need to NOT ask for listings, but instead ask how COBOL implements the handle abend API. I think it is totally legitimate that you state your belief in how it works and attempt to get confirmation. I have perused the literature, i.e. CICS manuals, and it is very scanty on how it treats COBOL. It has a paragraph about how it deals with assembler, restoring the registers, but it is silent on COBOL. Having them write a series of COBOL programs, or possibly getting Martin to do it, if they do not have the time, might be a way to go. Also having them analyze the assembler listings and surfacing the information in the API document (Horswill's PDF), would also be legitimate. If there are edge cases that you think might exist (you mentioned very large COBOL programs as a possible example), then we could get them to write those as well. This would accomplish a couple of things; It would begin to give us the start of a QA suite, that we could run in both environments (MF CICS and LTE); It would validate behavior of COBOL and handle conditions (as well as HANDLE AID and HANDLE ABEND).
>
> Try to keep in mind that the clean room process we are going thru, is that all "reverse engineering" is done in England (or the EU) as the laws in these jurisdictions are clearer. John Horswill's PDF document is the beginning of the remediation that is being done, so that we can legally state that the original work could have been done. "

369. In cross-examination, Mr Taylor agreed that the reference to "reverse engineering" was reverse engineering of Arg0 that was done by Mr Horswill and Mr Hitchings to produce the CICS Control Block document. He also agreed that the remediation referred to was a request for Winsopia to produce a document that could have been used to get the information already deduced by Texas Wormhole or obtained by LzLabs through reverse engineering from compiler listings and used in the SDM.

370. I accept the defendants' submission that IBM publishes the format of the Arg0 data area and the Arg0 values for some EXEC CICS commands, showing the output from the CICS translator for those commands and their parameters. That was demonstrated during cross-examination by reference to published IBM documentation. However, Winsopia did not confine its analysis to the publicly available information. The experts agreed that the level of detail in the CICS Control Blocks document goes to a level of granularity which is far more detailed that that published.

371. The defendants' case is that Winsopia's actions fell within permitted observation, study and testing of the CICS interfaces pursuant to Article 5(3) of the Software Directive. They submit that each of the matters being observed, studied and tested by Winsopia related to ideas and principles underlying the programs, such as interfaces contained in, and used by, the test programs; the division of responsibility between the executing program and the runtime environment; and the nature of the interactions between the compiler, customer application and runtime environment.

372. I reject that submission. Winsopia did not confine its actions to examining the input and output of each test application to determine the functioning of the CICS Transaction Server. Winsopia's analysis involved deciphering Arg0 and the parameter list for each EXEC CICS command and identifying the appropriate EXEC CICS interface program to determine how it operated. Such detailed analysis of the compiler, customer application and runtime environment investigated, not just the output of the program in response to a particular input but how the program achieved its output, that is, expression of the program, rather than its functioning.

373. It matters not that the component parts being analysed by Winsopia included the CICS interface. What was interrogated was not confined to the principles and ideas underlying the interface but rather it extended to the precise instructions and parameters generated by the CICS translator that formed an essential part of the implementation of the CICS Transaction Server Program. Such reverse engineering was not permitted by the terms of the ICA and did not fall within the observation, study and test exception of the Software Directive.

374. The defendants rely on an argument that the analysis of Arg0 was necessary in order to achieve interoperability of customer applications with the SDM.

375. Professor Donaldson agreed with Mr Swanson's understanding that Winsopia's aim was to decipher Arg0 and the remaining parameter list for each EXEC CICS command and match that to the appropriate EXEC CICS interface program such that an alternative could be substituted for the CICS runtime. Professor

Donaldson's opinion is that this information was necessary in order for LzLabs to develop support for customer programs relying on CICS services. Without a replacement interface capable of handling the parameters passed by those programs, it would have been impossible for those programs to interact, or communicate, successfully with the SDM.

376. That is correct from a technical point of view but, of course, it assumes that LzLabs does not simply recompile the customer program, an option always available to it. As stated by Mr Swanson and agreed by Mr Stephens in cross-examination, if LzLabs worked from customer source code and had their own translator capable of parsing EXEC CICS commands, there would be no need to understand the information passed to DFHEI1 or Arg0.

377. It follows that reproduction of the code and translation of its form were not indispensable to obtain the information necessary to achieve interoperability so as to engage the Article 6 exception.

378. Further, Article 6(2)(c) provides that information obtained by reproduction of the code and translation of its form is not permitted to be used for the development, production or marketing of a computer program substantially similar in its expression, or for any other act which infringes copyright. Development of the SDM was intended to provide an alternative program to call equivalent services to those provided by the CICS Transaction Server, using the same CICS commands and parameters, and producing the same responses. This was substantially similar in expression to Arg0.

379. It is not in dispute that the CICS Control Blocks document was supplied by Winsopia to LzLabs for the purpose of development of the SDM by LzLabs. The information supplied included the hexadecimal code for Arg0. That amounted to a direct transfer of part of the CICS Transaction Server software and was in breach of clauses 4.1, 4.1.2(b) and 4.1.3(b) of the ICA.

380. In summary on this issue:

i) Winsopia's analysis of the CICS translator, Arg0 and the Language Definition Tables was in respect of an ICA Program within the meaning of the ICA.

ii) Winsopia's analysis amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA.

iii) Winsopia's analysis did not fall within permitted observation, study and testing of the EXEC CICS interface pursuant to Article 5(3) of the Software Directive.

iv) Winsopia's analysis did not fall within the permitted exception in Article 6 of the Software Directive.

v) Winsopia's supply of the CICS Control Blocks document to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

Item 4: EXEC DLI (Paragraphs 27.18 & 28.19 of RRRAPOC)

381. As set out in the third joint statement prepared by the experts, CICS programs can access IMS APIs using an EXEC DLI command. The CICS translator converts EXEC DLI commands to the relevant programming language source code, including parameters passed to a CICS stub bound with the load module.

382. The claim and issues are the same as those set out above in respect of the Paragraph 11.3 claim at Item 3 above. Accordingly, my findings and conclusions are the same as for Item 3 above.

Item 5: IBM Binder Software (Paragraph 11.4 of the Technical Particulars)

383. The allegation is that Winsopia reverse engineered part of the IBM z/OS Binder, software which assists in transforming object code modules into executable programs. Specifically, it is alleged that Winsopia reverse engineered the mechanism in the Binder for compressing program objects, including disassembly of modules IEWBXZIP and/or IEWBXZP6 ("the Compression Modules").

384. The defendants' case is that no disassembly or decompilation tools were used by Winsopia to perform its analysis of the compression used by the Binder. The analysis carried out by Mr Lynch by searching for the relevant hexadecimal instructions amounted to observation, studying and testing permitted by Article 5(3) of the Software Directive. Alternatively, the information gleaned from any disassembly was necessary to achieve interoperability between the CPX/CPI and compressed program objects.

385. The z/OS Binder is a utility provided with z/OS that converts the output of language translators and compilers into an executable program object, which can either be read directly into virtual storage for execution or stored in a program library. Within the Binder there is a compression option that can be used to compress the additional data that the Binder stores, which reduces the disk storage needed.

386. On 4 September 2015, LzLabs opened DR-1254, seeking information from Winsopia about the program object compression performed by the z/OS Binder.

387. Mr Lynch was unable to find the required information about the Binder compression in published documentation. Therefore, he took an available program attached to an earlier DR and used the Binder to link it, both with and without the compress option, identifying patterns within the data that indicated use of dictionary compression.

388. Mr Lynch then used AMBLIST, a batch utility provided with z/OS, to display a list of the CSECTs in the load module and to show them in hexadecimal form. An analysis of the CSECTs indicated that the Compression Modules were responsible for dictionary compression. Mr Lynch also searched the hexadecimal code of the modules and discovered the specific instruction responsible for dictionary compression within the Binder, verified by decoding

the surrounding instructions. Subsequently, this was confirmed by publicly available information.

389. On 21 September 2015, Mr Lynch responded to DR-1254 with the following comment:

> "What you are missing is the dictionary.
>
> We do not have any source code available as this is a hardware implementation.
>
> Further analysis of the Binder compression process shows that both the dictionary and data to be compressed are held in a Data Space.
>
> The dictionary used appears to be exactly the same as that in the … load module member ...
>
> The output of the compression process results in data consisting of sequences of 9 bits.
>
> I think it is likely that the dictionary is propriet[a]ry to IBM so will need legal clearance before documenting."

390. In his further answer to DR-1254 on 24 September 2015 Mr Lynch stated:

> "The dictionaries used for the compression and expansion are not part of the Program Object data, they are hard coded in [the program]. I am sure therefore that they are proprietary to IBM which is why I am concerned about copying them into this case.
>
> There are in fact two compression dictionaries and two expansion dictionaries in [the module]. I believe there is one each for the two 'zip' areas seen in program … as mentioned in the case description.
>
> I have written a small test program to test these dictionaries against a small section of compressed and uncompressed data … and successfully compressed and expanded it using both the … instruction and with a service routine … So I can be certain that these are the correct dictionaries used by the Binder."

391. The experts' second joint statement includes the following agreed statements:

i) The z/OS binder is a utility provided by IBM with z/OS that prepares load modules. It can create load modules that are compressed. Winsopia performed analysis to determine how the binder compressed load modules and whether the compressed load modules could be processed on the SDM.

ii) Winsopia performed analysis on a compressed load module provided by a customer. They also bound this module as a non-compressed load module to compare the difference.

iii) Winsopia used the AMBLIST utility provided by IBM with z/OS to list the contents of the z/OS Binder modules: CSECTS, entry points and hexadecimal contents. AMBLIST does not provide disassembled output.

iv) From this AMBLIST output and other research, Winsopia believed that a particular z/OS instruction was used for compression. Winsopia disassembled a small number of instructions either side of the material instruction to confirm this belief.

v) Winsopia created a test program to confirm this, compressing and uncompressing data using what appeared to be dictionaries included in the z/OS binder CSECT.

vi) Information from this research was sent to LzLabs, including the compression mechanism used.

vii) There is no evidence that z/OS binder modules, the AMBLIST output or any compression dictionary used by the z/OS binder was sent to LzLabs.

viii) LzLabs decided not to support compressed load libraries on the SDM. Instead, the CPX tool on z/OS used z/OS binder APIs to uncompress load modules.

392. The disputed issues are:

i) whether Winsopia's analysis of the binder modules was in respect of an ICA Program within the meaning of the ICA;

ii) whether Winsopia's analysis amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA;

iii) whether Winsopia's analysis fell within permitted observation, study and testing of the Binder pursuant to Article 5(3) of the Software Directive;

iv) whether Winsopia's analysis was necessary in order to achieve interoperability of CPX and/or CPI with compressed customer applications and, as such, was permitted by Article 6 of the Software Directive.

393. The Compression Modules form component parts of the z/OS Binder supplied by IBM with z/OS Base V2, an ICA Program. For the reasons set out above, the modules within the Binder are ICA Programs for the purpose of the ICA.

394. Mr Lynch's evidence was that he ran the AMBLIST utility against one of the Compression Modules within the Binder program to display the hexadecimal form of the CSECTs contained within the sub-modules. Mr Stephens agreed in cross-examination that the use of AMBLIST gave Mr Lynch the means to look

at the structure of the IBM Binder load module and then look at the code and data in some of the CSECTs.

395. Mr Lynch decoded the instructions surrounding the compression instruction in the load module to ascertain that the Binder relied on hardware compression. In cross-examination he accepted that this amounted to disassembly of part of the code. His evidence that he carried out the disassembly in his head does not change the characterisation of his action as disassembly.

396. Mr Lynch stated in his DR response that he carried out further analysis of the Binder compression process which showed that both the dictionary and data to be compressed were held in a Data Space. From that analysis, he deduced that the dictionary used appeared to be exactly the same as that in the compression load module member. Although he could not recall exactly what he did, he agreed that he must have carried out runtime analysis of the Binder software:

> "Q. The DR itself doesn't say what further analysis you carried out for this and you don't deal with it in either of your statements but in order to do this you presumably must have done one of two things: either you disassembled the relevant part of the Binder code or you used a tool like XDC, or a trace or something, to analyse the IBM Binder software as it was executing?
>
> A. Well, I certainly didn't analyse the Binder as it was executing. It's a large complex piece of software, trying to analyse it would be an impossible task. I don't recall how I did this analysis to determine it was using what's called a data space.
>
> …
>
> Q. In order to conclude that the dictionary being used by the binder compression process is the same as the one in that module we see the name of, you must have compared the data and the binder's working memory as it executed decompression with the hexadecimal in that module?
>
> A. I presume so but I – I just can't recall."

397. Mr Swanson's evidence was that Mr Lynch's further analysis must have involved runtime analysis and decoding the instructions, either by hand, using a tool such as XDC, tracing, or disassembly. In cross-examination, Mr Stephens agreed that, based on Mr Lynch's oral evidence on this issue, he must have performed some sort of runtime analysis and, if so, he could have used tools such as a SLIP trap or XDC.

398. The above steps amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA.

399. The defendants submit that Mr Lynch's analysis was a quintessential example of observing, studying and testing the Binder modules in order to understand their ideas and principles of operation. I reject that submission. It is clear that

Mr Lynch's analysis went far beyond observation, study and testing of the ideas and principles underlying the Binder modules. It extended to a detailed disassembly and decoding of the object code of the Compression Modules, followed by an analysis of the execution of each instruction involved in compression and decompression. Such analysis investigated, not just the input and output of the Binder program but how the Binder program implemented its compression function. That amounted to analysis of the expression of the program, rather than its functioning.

400. I reject the defendants' contention that Winsopia is entitled to rely on Article 6 of the Software Directive because the analysis was not necessary for interoperability. As explained by Mr Lynch in his first witness statement, the information from the analysis was required to avoid the need for customers to recompile their applications and to improve marketability of the SDM:

> "LzLabs needed to decompress the executable application program in order for it to execute on the SDM. Building the decompression functionality into the SDM would avoid the need for customers to recompile their application with the compression feature turned off at the binder stage. To decompress a load module, you need to understand how it was compressed in the first place.
>
> …
>
> Given that a big selling point for the SDM is that it would allow customer applications to be migrated off the mainframe without the need to recompile them, LzLabs did not want to have to ask its customers to do this recompilation to get around the compression."

401. In summary, on this issue:

i) Winsopia's analysis of the Binder modules was in respect of an ICA Program within the meaning of the ICA.

ii) Winsopia's analysis amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA.

iii) Winsopia's analysis did not fall within permitted observation, study and testing of the Binder pursuant to Article 5(3) of the Software Directive.

iv) Winsopia's analysis did not fall within the exception in Article 6 of the Software Directive.

Compiler listings – summary of the dispute

402. The allegation in each case is that LzLabs wrote test programs which were designed to engage particular parts of the IBM COBOL and/or PL/I and or C/C++ runtime functionality but that served no business function in their own right. At LzLabs' request, Winsopia compiled these test programs using the

relevant IBM compiler, analysed the compiler listings, and sent to LzLabs both the output of the test programs and the compiler listings, revealing a full set of pseudo-assembly language instructions, including instructions and data structures inserted by the relevant compiler to set up the relevant Language Environment for the program and to manage the interaction between the program and the various runtime routines supporting it, as well as:

i) the names and purpose of runtime modules used in the execution and support of the relevant program;

ii) the address in memory of each runtime routine in the IBM vector table that the relevant language runtime environment maintains;

iii) the arguments that were passed to each runtime routine;

iv) technical comments explaining what the compiler had done and why;

v) the identity, content, and layout of IBM control blocks used by the relevant application and the runtime; and

vi) fragments of IBM mainframe software inserted by the compiler to handle initialisation of the relevant runtime and interaction between the runtime and the application.

403. IBM's case is that compilation of the test programs, the generation of compiler listings as a result of such compilation, and the analysis of compiler listings by Winsopia amounted to reverse engineering of the high-level language runtime environments. Even where redactions were made, the process did not remove the assembler instructions (fragments of IBM software created by the compiler) or mechanisms used by the relevant IBM language runtime to interact with running applications.

404. The defendants' case is that the acts carried out by Winsopia in relation to the compiler listing allegations did not concern ICA Programs but rather customer applications, and fell squarely within the observation, study and testing rights under Article 5(3) of the Software Directive.

405. In particular, the defendants submit:

i) the relevant computer program was a test program or customer application, not an IBM Program provided to Winsopia by IBM under the ICA;

ii) Winsopia was a lawful user of the relevant computer program;

iii) Winsopia was entitled under the ICA to perform the acts that it did, such as compiling and generating compiler listings for test programs, storing the resulting output from the compiler, running the test programs, and displaying the output from those test programs and the compiler;

iv) each of the matters being observed, studied and tested by Winsopia related to ideas and principles underlying computer programs, such as

> interfaces contained in, and used by, the test programs; the division of responsibility between the executing program and the runtime environment; and the nature of the interactions between the compiler, customer application, and runtime environment.

406. Each specific allegation is considered below.

Item 6: IGZCIVL COBOL runtime module (Paragraph 11.6 of the Technical Particulars)

407. The allegation is that Winsopia carried out reverse engineering of the IBM COBOL v4 compiler and the IGZCIVL runtime module, and transferred data structures from the same to LzLabs for use in development of the SDM.

408. The defendants' case is that Winsopia used compiler listings to study the interface in a customer application that it uses to call a runtime service, IGZCIVL. These activities did not relate to an ICA Program; they related to a test program and fell within the observation, study and test exception right conferred by Article 5(3) of the Software Directive.

409. IGZCIVL is a COBOL runtime routine that compares a data item to a figurative constant and determines whether the data value is equal to, greater than or less than the figurative constant. It is a component of the IBM supplied load module IGZCPAC, which is part of the SCEERUN library provided by IBM as part of z/OS.

410. Where a source code application contains an "IF" statement, the IBM COBOL compiler may produce machine code to evaluate the information and take appropriate action within the compiler program. As Professor Donaldson explained, this is usually confined to relatively simple comparisons; in more complex cases, the compiler will produce machine code to outsource the comparison exercise to the IGZCIVL routine at runtime.

411. On 19 January 2016, DR1485 was opened by Mr Bowler of LzLabs, requesting Winsopia to discover the format of parameters passed by the COBOL program to IGZCIVL when an IF statement compares a data item with a figurative constant. He provided test programs that he had written and supplied the parameters for the same. He asked Winsopia to send to LzLabs the redacted compilation listing, redacted load module and program display output file.

412. The test programs were compiled and executed on the Winsopia Mainframe by Mr Bray of Winsopia, deliberately triggering an error to generate the required output. The compiler listings were sent to LzLabs, with some text redactions but leaving the machine instructions and pseudo-assembler statements visible.

413. From studying this material, LzLabs deduced the circumstances in which the compiler would insert machine code to perform the required comparison itself or insert a call to be made to IGZCIVL during runtime to perform the comparison, together with the order of the parameters to be passed to IGZCIVL.

414. LzLabs developed the SDM equivalent of the IGZCIVL routine, which was present in the SDM Git repository from at least August 2013, prior to Winsopia's acquisition of the mainframe. On 28 January 2016 Mr Bowler modified the SDM code relating to the handling of parameters, inserting code to cause an abend (abnormal termination) in specified circumstances. Mr Bowler's explanatory comment referenced his understanding of the order of the data item and figurative constant as informed by DR1485.

415. The experts' second joint statement includes the following agreed facts:

i) LzLabs provided two test programs to Winsopia to be compiled and executed on the z/OS system.

ii) Winsopia compiled and executed the programs, returning compiler listings with pseudo assembler statements generated by the COBOL compiler, output created by the program when it executed, and scrubbed load modules.

iii) The purpose of this exercise was to determine the parameters input to IGZCIVL and determine the results when these parameters are entered.

iv) IGZCIVL is an IBM provided CSECT included in the runtime library SCEERUN provided by IBM with z/OS.

v) From the information provided, including assembler statements, LzLabs determined that a z/OS language environment model IGZCIVL would not be called in certain circumstances.

vi) LzLabs also discovered that the operands were always in a predictable order.

vii) The SDM IGZCIVL equivalent module was modified to reflect this.

viii) The SDM file does not reproduce source or object code from the IBM IGZCIVL module.

416. The disputed issues are:

i) whether Winsopia's analysis of the interaction between the compiler and the IGZCIVL runtime module was in respect of an ICA Program within the meaning of the ICA;

ii) whether Winsopia's analysis amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA;

iii) whether Winsopia's analysis fell within permitted observation, study and testing of the compiler and/or IGZCIVL pursuant to Article 5(3) of the Software Directive;

iv) whether Winsopia's analysis was necessary in order to achieve interoperability of customer applications with the SDM and, as such, was permitted by Article 6 of the Software Directive.

417. The defendants submit that the object of Winsopia's analysis was the interface used to call the IGZCIVL runtime service, which interface was not itself an ICA Program. I reject that submission. Regardless of whether one could describe the interaction between the compiler and the IGZCIVL runtime module as an interface, what was interrogated by Winsopia was the compiler listing produced during compilation of the test program, which disclosed the parameters input to IGZCIVL, the circumstances in which IGZCIVL would be called and the results when certain parameters were entered. In so doing, Winsopia analysed the internal operation of the compiler and the IGZCIVL runtime module.

418. It is common ground that IBM Enterprise COBOL v4 is an ICA Program. The compiler and its associated IGZCIVL runtime module are both components of IBM Enterprise COBOL v4. For the reasons set out above, they both fall within the definition of an ICA Program for the purpose of the ICA.

419. Winsopia compiled and ran the test programs in order to expose elements of the undocumented internal workings of the compiler and IGZCIVL components, through compiler listings, including pseudo-assembler statements, load modules and the program display output files. The information gleaned was not documented publicly by IBM. As Professor Donaldson agreed in his evidence, the matters analysed reflected design choices by the creators of the compiler and its associated runtime. This amounted to reverse engineering.

420. The defendants submit that Winsopia's acts were observation, study and testing of the compiled test programs while running, and in particular the interface that invoked the IGZCIVL runtime module. The purpose of such exercise was to determine interface information relating to IGZCIVL, namely, the parameters passed to IGZCIVL as input (in particular their order and relationship), the output provided back to the test program and the circumstances in which the interface would be called by the test program.

421. The purpose of Winsopia's analysis was not confined to the functioning of the program to determine underlying ideas and principles; it was to determine how the compiler and IGZCIVL implemented IF statements. It included discovery of the parameters passed to IGZCIVL, the circumstances and order in which the parameters were input, and the results of the same. That amounted to expression of the program, rather than its functioning.

422. The Article 6 exception in the Software Directive is not applicable. It is common ground that LzLabs already had a functioning equivalent of IGZCIVL in the SDM. Further, it was open to LzLabs to adopt a solution based on recompilation of customer source code applications. Therefore, Winsopia's analysis was not necessary in order to achieve interoperability of customer applications with the SDM.

423. In summary, on this issue:

    i) Winsopia's analysis of the interaction between the compiler and the IGZCIVL runtime module was in respect of an ICA Program within the meaning of the ICA.

ii) Winsopia's analysis amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA.

iii) Winsopia's analysis did not fall within permitted observation, study and testing of the components so as to engage Article 5(3) of the Software Directive.

iv) Winsopia's analysis was not necessary in order to achieve interoperability of customer applications with the SDM so as to engage Article 6 of the Software Directive.

Item 7: CICS Translators (Paragraph 20.1-2 of the Technical Particulars)

424. Some IBM compilers contain embedded CICS translators, enabling the dynamic translation of an EXEC CICS command during the compilation process. This is in contrast to external CICS translators, which pre-process the EXEC CICS statement before a separate compiler is used to compile the load module.

425. The allegation is that Winsopia reverse engineered (i) the IBM CICS translators and (ii) the interaction between the CICS translators and the runtime to test whether the Arg0 byte strings generated by the embedded CICS translators differed from those generated by the external CICS translators.

426. The defendants' case is that Winsopia used compiler listings to study the embedded and external CICS translators. They ascertained that they produced the same output and, as a result, no changes were required to the SDM. These activities did not relate to an ICA Program; they related to a test program and fell within the observation, study and test exception right conferred by Article 5(3) of the Software Directive; alternatively, they were necessary for interoperability within the Article 6 exception.

427. It is common ground that in May 2014 Martin Truebner of LzLabs obtained from a colleague, probably at Winsopia, two unredacted compiler listings, including pseudo-assembler lists, for the same test program, one using an external CICS translator, the other using an embedded CICS translator. Mr Treubner sent a note to Mr Taylor, explaining what he had done and the results of his analysis, namely that there was no difference in the output. Mr Taylor explained that his actions were inappropriate and in breach of the Code of Conduct, as recorded in his email dated 14 May 2014 to Ulrich Weibel. Mr Taylor's evidence is that Mr Truebner confirmed that he destroyed the listings.

428. The disputed issues are:

i) whether Winsopia's analysis of the embedded and external CICS translators was in respect of an ICA Program within the meaning of the ICA;

ii) whether Winsopia's analysis amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA;

iii) whether Winsopia's analysis fell within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive;

iv) whether Winsopia's analysis was necessary in order to achieve interoperability of customer applications with the SDM and, as such, was permitted by Article 6 of the Software Directive.

429. IBM's case is that the CICS translators, Arg0 and the Language Definition Tables are component parts of the CICS Transaction Server, an ICA Program, as set out in respect of the Paragraph 11.3 allegation above.

430. The defendants' case is that the information sent by Winsopia to LzLabs took the form of compiler listings and scrubbed load modules. This was interface information, comprising material output by the compiler or pre-processor and not an ICA Program.

431. Winsopia's production of the unredacted compiler listings included the pseudo-assembler language listings, close representations of the machine instructions that the compiler would emit as the object code of the compiled program. This exposed otherwise undocumented IBM code generated by the CICS translators and the parameters determined by the Language Definition Tables. For the reasons set out in respect of the Item 3 allegation above, this amounted to reverse engineering of the CICS translators, Arg0 and the Language Definition Tables, which were ICA Programs within the meaning of the ICA.

432. The defendants' case is that Winsopia's actions amounted to permitted observation, study and testing of the functioning of the two translators in order to determine the underlying ideas and principles in the form of interface information, namely, whether the translated Arg0 calls output by the IBM COBOL compiler were the same as those output by the external CICS translator.

433. I reject that submission for the reasons set out above in respect of the Item 3 allegation. Winsopia did not confine its actions to examining the input and output of the test program to determine the functioning of the CICS translators. Winsopia already knew that the CICS translators would generate a CALL to the relevant CICS routine with the parameter list required for execution. Winsopia's production of the unredacted listings with pseudo-assembly lists disclosed, not just the output of the program but how the program achieved its output, that is, expression of the program, rather than its functioning.

434. For the reasons set out in respect of the allegation in item 3 above, production of the compiler listings was not indispensable to obtain information necessary to achieve interoperability so as to engage the Article 6 exception.

435. In summary, on this issue:

i) Winsopia's analysis of the embedded and external CICS translators was in respect of an ICA Program within the meaning of the ICA.

ii) Winsopia's analysis amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA.

iii) Winsopia's analysis did not fall within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

iv) Winsopia's analysis did not fall within the permitted exception in Article 6 of the Software Directive.

Item 8: Floating point rounding rules (Paragraph 20.3 of the Technical Particulars)

436. The IBM C language runtime contains functionality used to round floating point numbers (non-integers which contain a decimal point). Arithmetic and conversion operations performed by the mainframe produce an intermediate result correct to infinite precision and with unbounded range. Unless that intermediate result can be represented exactly in the target format, it is replaced by a value that depends on the rounding mode used. There are various rounding modes that can be used and the mathematical operations used to perform rounding are set out in a published standard of the Institute of Electrical and Electronics Engineers Inc, IEEE-754.

437. The IBM mainframe provides a floating point control register ("the FPC") that enables users of the IBM C language runtime to specify the rounding mode to format the results of their calculations. The format of the FPC is published by IBM in publicly available documentation, namely, the IBM Principles of Operation manual.

438. In the IBM C language runtime, a data structure called "_FP_fpcreg_t" ("the FPC Data Structure"), is used to identify the rounding mode specified by the customer application, using IBM supplied APIs to view and modify the FPC accordingly. The functions of the APIs are set out in publicly available documentation.

439. The experts agree that the IBM XL C/C++ compiler provides a header file that provides C language source code that maps the FPC data area, which is made available to z/OS customers but which is not publicly documented in full. The z/OS language environment APIs take this data as input and provide it as output. The use of this structure is partially shown in IBM documentation but the internal code is not shown.

440. The allegation is that Winsopia used reverse engineering to test the APIs that modify the FPC, ascertain the effect of different values on the API rounding functions and determine how it performed number rounding.

441. On 17 February 2021 Peeter Joot of LzLabs created DR 520, requesting Winsopia to run test programs to discover the rounding mode selection data structure and the values corresponding to four floating point number functions. Subsequently further versions of the test programs were supplied to be run on the mainframe. Winsopia compiled and executed the programs and provided to LzLabs the program output, compiler listings and scrubbed load modules created.

442. On 18 February 2021 Mr Joot made a commit to the SDM source code, adding parameter structures for the SDM replacement FPC data structure, citing as the

source of the change DR 520. On 1 March 2021 Mr Joot made a further commit, modifying one of the fields of the data structure, again citing DR 520 as the source.

443. The experts' second joint statement contains the following agreed statements:

i) The SDM has its own file mapping the FPC. It is plausible that the contents of this file could have been created from publicly available IBM documentation. However, evidence indicates that information from Winsopia was used.

ii) LzLabs made modifications to the SDM from the information obtained to support the setting and retrieval of rounding mode to match that provided by the IBM z/OS language environment and XL/C compiler.

iii) Source code material from the IBM mainframe was not copied into the SDM.

444. The disputed issues are:

i) whether Winsopia's analysis of the FPC Data Structure was in respect of an ICA Program within the meaning of the ICA;

ii) whether Winsopia's analysis amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA;

iii) whether Winsopia's analysis fell within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive;

iv) whether Winsopia's analysis was necessary in order to achieve interoperability of customer applications with the SDM and, as such, was permitted by Article 6 of the Software Directive;

v) whether Winsopia's supply to LzLabs of the compiler listings, including the format and values of the rounding mode selection data structures, constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

445. IBM's case is that the FPC Data Structure and the IBM supplied APIs are component parts of the IBM C Language runtime, part of z/OS v2 Base, an ICA Program.

446. The defendants' case is that the information sent by Winsopia to LzLabs took the form of compiler listings and scrubbed load modules. This was interface information, comprising material output by the compiler and not an ICA Program.

447. Winsopia provided output data, printing out all the bytes contained in the requested configurations of the FPC Data Structure. As such, this amounted to reverse engineering of the C Language runtime, which was an ICA Program within the meaning of the ICA and a transfer outside Winsopia's Enterprise.

448. The defendants' case is that Winsopia's actions amounted to permitted observation, study and testing of the interfaces that modify the FPC register, in particular, the format and values of the rounding mode selection data structures inserted by the IBM C compiler.

449. I reject that submission. Winsopia did not confine its actions to examining the input and output of the test programs to determine the functioning of the C language runtime. Winsopia's production of the compiler listings disclosed, not just the output of the program but the data structures and how the C language runtime dealt with rounding modes or changes to rounding modes specified in customer applications, that is, expression of the program, rather than its functioning.

450. For the reasons set out above in respect of the compiler listings allegations, production of the compiler listings was not indispensable to obtain information necessary to achieve interoperability so as to engage the Article 6 exception.

451. Although it is common ground that Winsopia did not transfer to LzLabs any IBM source code or CSECTs, it did provide the values and configurations for the FPC Data Structure. This amounted to a transfer of an ICA Program in breach of clause 4.1.3(b) of the ICA.

452. It is common ground that LzLabs requested, and Winsopia supplied, the information required in DR 520 for development of the SDM by LzLabs. This was in breach of clauses 4.1 and 4.1.2(b) of the ICA.

453. In summary, on this issue:

i) Winsopia's analysis of the FPC Data Structure was in respect of an ICA Program within the meaning of the ICA.

ii) Winsopia's analysis amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA.

iii) Winsopia's analysis did not fall within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

iv) Winsopia's analysis did not fall within the permitted exception in Article 6 of the Software Directive.

v) Winsopia's supply of the compiler listings, including the format and values of the rounding mode selection data structures, to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

Item 9: IBM PL/1 compiler (Paragraph 20.4 of the Technical Particulars & Paragraph 27 of the POC)

454. The allegation is that Winsopia carried out reverse engineering of interactions between the IBM PL/I compiler and its corresponding language environment over a number of years.

455. The defendants' case is that Winsopia used the PL/I compiler listings to study the interface in customer applications to understand which Language Environment runtime services were being called by the applications and which runtime calls corresponded to the source code statements in the applications. These activities did not relate to an ICA Program; they related to a test program and fell within the observation, study and test right conferred by Article 5(3) of the Software Directive.

456. LzLabs had access to compiler listings for the PL/I programs but no line numbers for the source code of the program, making it difficult for them to link source code statements with corresponding object code and pseudo-assembly language.

457. On 15 March 2017, Mr McKeough of LzLabs created DR-2305, requesting Winsopia to provide line numbers in order to enable LzLabs to link statements in the source code of the PL/I programs with the pseudo-assembly language and machine code content of the compiler listings.

458. Winsopia supplied the contents of the compiler listings, including pseudo-assembly language representations of compiler generated machine code used to invoke IBM PL/I runtime services. Winsopia also provided a spreadsheet, linking source code statements to the pseudo-assembly language in the compiler listings relating to the PL/I compiler, invoking IBM runtime functions to implement a PL/I language construct or feature.

459. The experts' second joint statement records:

    i) LzLabs had PL/I source code and corresponding load modules of seven programs from a customer.

    ii) LzLabs did not have a complete PL/I compile listing of these programs.

    iii) At LzLabs' request, Winsopia recompiled these programs and returned: (a) compiler listings of these programs, including pseudo-assembler code generated by the compiler; and (b) an excel spreadsheet showing some lines of assembler code generated by the PL/I compiler.

    iv) This information provided LzLabs with information they were unable to obtain from the incomplete listings provided by the customer; in particular, the z/OS language environment runtime modules called for certain PL/I source code statements.

    v) The DR in question was only one of others requesting additional, similar information.

460. The disputed issues are:

    i) whether Winsopia's analysis of the PL/I compiler listings and Language Environment was in respect of an ICA Program within the meaning of the ICA;

ii) whether Winsopia's analysis amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA;

iii) whether Winsopia's analysis fell within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive;

iv) whether Winsopia's analysis was necessary in order to achieve interoperability of customer applications with the SDM and, as such, was permitted by Article 6 of the Software Directive;

v) whether Winsopia's supply of the compiler listings to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

461. IBM's case is that the PL/I compiler is part of Enterprise PL/I for z/OS v4 and the IBM Language Environment runtime is part of z/OS. As such, they both fall within the definition of an ICA Program.

462. The defendants submit that Winsopia's study was limited to analysis of the customer application source code and compiler listings that were generated for each customer application. This did not involve any analysis of any Language Environment runtime module and therefore did not concern an ICA Program.

463. I reject the defendants' submission. Winsopia's analysis was not limited to a study of the customer application; it was to ascertain the means by which the IBM PL/I runtime and compiler implemented PL/I language features used in the source code of the customer programs at runtime. As such, it concerned an ICA Program.

464. The defendants' case is that Winsopia's actions amounted to permitted observation, study and testing of the interfaces of the compiled customer application, to enable LzLabs to document the structure of the runtime function calls in the assembler code of the customer application, so that the SDM could redirect them to the appropriate open-source functions.

465. I reject that submission. Winsopia's production of the compiler listings disclosed the pseudo-assembly language representing the object code generated by the IBM PL/I compiler to invoke PL/I runtime routines. This enabled Winsopia to follow each instruction in sequence to gain an understanding as to the compiler-generated code inserted into a PL/I load module by the IBM PL/I compiler to implement language features used in the PL/I source code of the application. That concerned expression of the program, rather than its functioning.

466. For the reasons set out above in respect of the compiler listings allegations, production of the compiler listings was not indispensable to obtain information necessary to achieve interoperability so as to engage the Article 6 exception.

467. Winsopia transferred to LzLabs pseudo-assembler code generated by the IBM PL/I compiler, together with an excel spreadsheet showing lines of assembler code generated by the PL/I compiler. This amounted to a transfer of an ICA Program in breach of clause 4.1.3(b) of the ICA.

468. As Mr Jaeger explained in his first witness statement, LzLabs requested, and Winsopia supplied, the information required in respect of the PL/I compiler listings for development of the SDM by LzLabs. This was in breach of clauses 4.1 and 4.1.2(b) of the ICA.

469. In summary, on this issue:

i) Winsopia's analysis of the PL/I compiler listings and Language Environment was in respect of an ICA Program within the meaning of the ICA.

ii) Winsopia's analysis amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA.

iii) Winsopia's analysis did not fall within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

iv) Winsopia's analysis was not permitted by Article 6 of the Software Directive.

v) Winsopia's supply of the compiler listings to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

Item 10: XML Parse statements (Paragraphs 33-38 of the Technical Particulars)

470. The allegation is that Winsopia reverse engineered the exception handling mechanism within IBM's COBOL XML Parser and the COBOL runtime, to assist LzLabs in developing support for XML PARSE statements in COBOL programs for the SDM.

471. The defendants' case is that although IBM COBOL XML Parser is provided with Enterprise COBOL v4, v5 and v6, that was not the subject of analysis. Winsopia analysed the XML Parser API and related data structures by which customer applications may call and interface with the XML Parser. These activities did not relate to an ICA Program and fell within the observation, study and test exception right conferred by Article 5(3) of the Software Directive.

472. Extensible Markup Language ("XML") is a standard format that can be used to define data structures. XML PARSE statements enable COBOL programs to parse an XML document, breaking it down into its component parts and processing each part. A key aspect of this functionality is the ability to detect and process syntax errors in a document, together with the start and end of the document.

473. In order to use the XML PARSE functionality, a COBOL application program must include the customer's chosen processing procedure, a function which contains the business logic to handle different types of XML events, together with an XML PARSE statement. During compilation, the COBOL compiler generates a call with a parameter list to invoke the relevant XML PARSE runtime routine and request the XML Parser service. During execution, the XML Parser begins the parsing and establishes the processing procedure with

the Parser. For each XML event that it detects, the Parser sets the event name in a special register and transfers control to the invoking program, where instructions generated by the COBOL compiler as part of the XML PARSE statement direct processing to the processing procedure. After processing the event, compiler generated code determines whether the document end has been reached, or whether control should be returned to the Parser so that it can continue to parse the document.

474. The experts agree that for the SDM to support the execution of compiled COBOL programs, without the programs being rewritten or recompiled, it was necessary for LzLabs to understand the interfaces between the compiler-generated code and the runtime routines. LzLabs obtained such understanding through the DR process: DR-0210 dated 13 August 2014, DR-0218 dated 17 August 2014, DR-4509 dated 25 November 2019 and DR-112 dated 14 June 2020.

475. There is no dispute about what Winsopia did and the experts' second joint statement includes the following agreed facts:

i) LzLabs prepared COBOL programs using XML parsing, and handling exceptions that may occur. LzLabs also prepared XML input for these programs: some of this input was valid, some would generate exceptions.

ii) Winsopia compiled and executed these programs, using the provided XML data. Winsopia sent to LzLabs compiler listings generated by the IBM COBOL compiler for these programs, output produced by the programs when executed by Winsopia, and scrubbed load modules.

iii) LzLabs used this information to understand the interfaces between the compiler-generated code and the IBM runtime routines, and implement replacement SDM routines.

476. The disputed issues are:

i) whether Winsopia's analysis of the interaction between the IBM COBOL compiler and the Language Environment runtime was in respect of an ICA Program within the meaning of the ICA;

ii) whether Winsopia's analysis amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA;

iii) whether Winsopia's analysis fell within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive;

iv) whether Winsopia's analysis was necessary in order to achieve interoperability of customer applications with the SDM and, as such, was permitted by Article 6 of the Software Directive;

v) whether Winsopia's supply of the compiler listings to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

477. The defendants submit that the object of Winsopia's analysis was not the document processing performed by the XML Parser but the interplay between the compiled customer application and the Language Environment runtime when XML events or exceptions are generated during parsing. That concerned only the interface, which is not an ICA Program. I reject that submission. The object of Winsopia's analysis was the complex sequence of inter-dependent instructions implemented by the compiler-generated code in the load module and the runtime routines forming part of the Language Environment. For the reasons set out above, the IBM COBOL compilers, v4, v5 and v6 and the Language Environment runtime fall within the definition of an ICA Program for the purpose of the ICA.

478. Winsopia ran the test programs in order to expose elements of the undocumented interplay between the compiler and runtime components. The compiler listings disclosed compiler-generated machine code and pseudo-assembler representation of such code, with no material redactions. This revealed the input parameters, special registers containing XML related information and control flow to the XML event processing routine. The interaction between the COBOL compiler and z/OS Language Environment runtime for XML parsing functionality is not fully documented publicly. This amounted to reverse engineering.

479. It is common ground that the compiler listings were sent by Winsopia to LzLabs and used to implement replacement routines in the SDM.

480. The defendants submit that Winsopia's acts were observation, study and testing within Article 5(3) of the Software Directive. Winsopia was engaged in characterisation testing of the XML Parse interface. This involved Winsopia compiling and running the COBOL test programs, producing compiler listings and reviewing their output in response to particular processing events. Winsopia was entitled to perform each of these steps as a licensed user of its mainframe. In so doing, it is said that Winsopia was observing, studying and testing characteristics of the XML Parse interface, including its parameters, the output generated in response to errors encountered during parsing and how control was passed between the test program and the runtime environment.

481. The purpose of Winsopia's analysis was not confined to the functioning of the program to determine underlying ideas and principles; it was to determine how the IBM COBOL compilers and the corresponding runtimes worked together to implement XML PARSE statements. It included step-by-step analysis of instructions generated by the compiler and implemented by the runtime routines, the input parameters, special registers and control flow back and forth between the load modules and the Parser. That amounted to expression of the program, rather than its functioning.

482. The Article 6 exception in the Software Directive is not applicable. It was open to LzLabs to adopt a solution based on recompilation of customer source code applications. Therefore, Winsopia's analysis was not necessary in order to achieve interoperability of customer applications with the SDM.

483. In summary, on this issue:

i) Winsopia's analysis of the interaction between the IBM COBOL compiler and the Language Environment runtime was in respect of an ICA Program within the meaning of the ICA.

ii) Winsopia's analysis amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA.

iii) Winsopia's analysis did not fall within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

iv) Winsopia's analysis was not necessary in order to achieve interoperability of customer applications with the SDM and, as such, was not permitted by Article 6 of the Software Directive.

v) Winsopia's supply of the compiler listings to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

Item 11: COBOL initialisation, branching and I/O declaratives (Paragraphs 27.4&27.5 RRRAPOC)

484. The allegation is that Winsopia carried out reverse engineering of features of Enterprise COBOL for z/OS v4, designed to support I/O declaratives, and transferred to LzLabs load modules containing initialisation code fragments and data relevant to I/O declaratives and branching statements in breach of clauses 4.1, 4.1.1(a), 4.1.3(a) and 4.1.3(b) of the ICA.

485. The defendants' case is that these features are interfaces and/or other functionality provided by the relevant programming languages; as such, they are not ICA Programs and Winsopia was entitled to study, test and observe them pursuant to rights conferred by Article 5(3) of the Software Directive.

486. Initialisation is a service provided by the runtime environment when execution of a COBOL program commences, whereby the data structures within the system memory are defined and populated with compiler-generated code necessary to execute the program. The COBOL compiler generates (i) the object code of the application program, (ii) the COBOL initialisation code, a sequence of machine code instructions and data; and (iii) a data structure that is specific to the storage requirements of each compiled customer application. A routine inserted into the load module calls the COBOL runtime to begin initialisation. The initialisation process involves an inter-dependent arrangement of the initialisation code instructions with the application program and data structures in the load module and the runtime.

487. COBOL I/O declaratives are a COBOL exception handling feature that allow I/O errors to be handled according to instructions set out by the application developer in application source code. COBOL branching statements, such as "GO TO DEPENDING ON" and "ALTER GO TO DEPENDING ON" statements permit branching to blocks of code according to a variable set out by the application developer in application source code.

488. During initialisation, the data structures in the runtime are populated with information reflecting the presence of I/O declaratives or COBOL branching statements in the application and their particular characteristics.

489. For the SDM to support the above functionality, it needed to be able to resolve conflicts between nested declaratives. Nested programs, whereby programs are contained within other programs, are a feature of COBOL. Each nested program might contain one or more I/O declarative statements and there are complicated rules for determining which declarative should gain control. Further, the SDM needed to replicate functionality enabling past executions of certain branching statements to be refreshed. Without support for a compatible interface in the SDM, it could not execute COBOL load modules.

490. Mr Jaeger's evidence was that LzLabs' initial development of these functions was carried out before 2013 using an SDM prototype product running on the Hercules emulator. LzLabs developed functionality for the SDM to execute IBM compiler-generated initialisation code by executing customer compiled and link-edited load modules on the prototype and using a Linux tracing tool, GNU, to step through the initialisation and execution process. It developed support for COBOL branching statements using IBM compiler listings in addition to execution of load modules and use of the tracing tool. LzLabs determined how the SDM COBOL run-time needed to interact with COBOL programs containing EXCEPTION/ERROR declaratives by running different test programs on the SDM prototype. An I/O error was forced, and the various control blocks which form part of the compiled object code were inspected to discover how the COBOL program makes the presence of the declaratives visible to the run-time.

491. It was accepted by Mr Swanson in cross-examination that, prior to August 2013, the SDM provided initial support for initialisation, I/O declaratives and branching:

> "Q. And I think it's also common ground that the initial support for all three of these was in place in the SDM code before August 2013?
>
> A. The initial implementation, yes.
>
> Q. And you will agree that support had been developed by LzLabs based on compiler listings, a modified version of the Hercules emulator and an early prototype of the SDM?
>
> A. Along with use of the GNU debugger, yes.
>
> Q. And that was all before Winsopia came on the scene and before Winsopia had any use for its IBM mainframe?
>
> A. Initial development, yes. "

492. By September 2013, LzLabs had a fully functioning COBOL initialisation process which code was added to the SDM Git repository. However, it was still

necessary for LzLabs to gain an understanding and interpretation of the data structures that were produced following COBOL initialisation. Professor Donaldson agreed that such information is not publicly available.

493. In Annex 2 to his first statement, based on a document prepared by Mr Bowler, Mr Jaeger described how SDM support for I/O declaratives was developed through the DR system. Between 2014 and 2015 LzLabs created test programs, which were compiled by Winsopia on the mainframe to generate compiler listings. The compiler listings and load modules were then sent to LzLabs. The compiler listings disclosed the names and addresses within memory of the data structures related to I/O declaratives but not the values that populate them. The load modules included compiler-generated initialisation code and code relevant to I/O declaratives. LzLabs ran the load modules on the SDM, using a tracing tool to step through the processing so as to ascertain the way in which the data structures and fields were used in the IBM runtime.

494. In cross-examination, Mr Jaeger accepted that the development of compatible interfaces to support I/O declaratives was carried out by LzLabs, using Winsopia's work:

> "Q. … it relied upon Winsopia compiling code with the IBM compiler on its mainframe, didn't it?
>
> A. Yes, but the development was done by LzLabs.
>
> Q. I see. So … you mean it's correct to say the development was done by LzLabs but there were acts of Winsopia involved, weren't there?
>
> A. Yes, the DR system.
>
> …
>
> Q. … these DRs involved Winsopia, didn't they?
>
> A. They did, yes.
>
> Q. Yes, and that was part of the development of the I/O declarative support on the SDM?
>
> A. Exactly… the use of Winsopia greatly speeded up our ability to deliver this functionality in a meaningful timeframe.
>
> Q. It was actually more than that, wasn't it, because you needed to know what the compiler was going to do with the code? You had to have a compiler available to you, didn't you?
>
> A. No … or a customer's. Somebody would have to send us a load module, that's absolutely the case … "

495. The third joint statement by the experts includes the following agreements:

i) To support execution of compiled COBOL CSECTs in the SDM, LzLabs needed to understand interactions between compiled code and the IBM COBOL runtime, including related to I/O declaratives, initialisation and branching.

ii) Before August 2013, the SDM provided initial support for COBOL I/O declaratives, initialisation and branching. The development work that led to this initial support cannot have been informed by access to Winsopia's mainframe.

iii) Witness evidence indicates that, before August 2013, LzLabs studied interactions between compiled COBOL code and the COBOL runtime by (a) studying compiler listings, and (b) using a modified version of the Hercules emulator and/or early prototype of the SDM to step through the compiled code of test programs in a Linux x86 environment.

iv) Support for COBOL branching in the SDM is limited, and has not materially changed since August 2013.

v) After August 2013, support for I/O declaratives was further developed and informed by a series of discovery requests.

vi) The SDM source code does not reproduce source or object code from modules of the IBM COBOL runtime.

496. The disputed issues are:

i) whether Winsopia's role in developing support for I/O declaratives was in respect of an ICA Program within the meaning of the ICA;

ii) whether Winsopia's actions in creating compiler listings and compiling the test programs amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA;

iii) whether Winsopia's actions fell within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive;

iv) whether Winsopia's supply of the load modules to LzLabs constituted breach of clauses 4.1, 4.1.1(a) and/or 4.1.3(b) of the ICA.

497. The defendants submit that none of Winsopia's actions related to ICA Programs. Rather, they were features of the COBOL programming language. The COBOL initialisation data structures are an initialisation interface. I/O declaratives involve control blocks that interconnect with the runtime services that support I/O declaratives. Branching involves interconnecting different parts of the customer application. As such, they are features of the COBOL programming language, defined within compiled customer applications and visible in compiler listings for those customer applications, at least from COBOL v5 onwards.

498. I reject that submission. The development of support for COBOL initialisation, involving the creation of test programs, production of compiler listings and

supply of load modules, was not limited to investigation of customer-defined features that acted as connecting interfaces to the runtime. As set out above, the initialisation process involves a complex inter-dependent arrangement of the initialisation instructions, the program and data structures in the compiled load module and the runtime. The purpose of the discovery process was to understand the operation of the COBOL compiler and runtime in generating and implementing code for the initialisation process and the conventions used to resolve I/O declarative conflicts or implement branching procedures. For the reasons set out above, the IBM COBOL compiler and the Language Environment runtime fall within the definition of an ICA Program for the purpose of the ICA.

499. Mr Jaeger's evidence was that the compiler listings were used to discover the declarative data structures for error/exception handlers in programs compiled by COBOL version 4. Winsopia ran the test programs on the mainframe in order to expose elements of the undocumented interplay between the compiler, the program, the data structures and runtime components. The pseudo-assembly instructions in the compiler listings disclosed the names and addresses within memory of the data structures related to I/O declaratives. This amounted to reverse engineering.

500. Load modules supplied by Winsopia were executed on the SDM with forced I/O errors, enabling LzLabs to inspect the contents of memory, or using a tracing tool, to disclose how the data structures were populated. The interaction between the COBOL compiler and z/OS Language Environment runtime for I/O declaratives functionality is not fully documented publicly. This amounted to reverse engineering by LzLabs that was facilitated by Winsopia.

501. It is common ground that the compiler listings and load modules were sent by Winsopia to LzLabs. The load modules included initialisation code, data blocks and code fragments, data structures associated with I/O declaratives and COBOL branching statements.

502. The defendants submit that Winsopia's acts were observation, study and testing within Article 5(3) of the Software Directive.

503. The purpose of Winsopia's DR work was not confined to the functioning of the program to determine underlying ideas and principles; it was to determine how the IBM COBOL compilers and the corresponding runtimes worked together to implement initialisation. It included analysis of instructions generated by the compiler and implemented by the runtime routines. That amounted to expression of the program, rather than its functioning.

504. In summary on this issue:

    i) Winsopia's role in developing support for I/O declaratives was in respect of an ICA Program within the meaning of the ICA.

    ii) Winsopia's actions in creating compiler listings and compiling the test programs for execution on the SDM using tracing tools amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA.

iii) Winsopia's analysis did not fall within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

iv) Winsopia's supply of load modules and compiler listings to LzLabs constituted breach of clause 4.1, 4.1.1(a) 4.1.3(b) of the ICA.

Item 12: PL/I Condition handling (Paragraphs 27.10-27.12 of RRRAPOC)

505. The allegation is that Winsopia carried out reverse engineering of Enterprise PL/I for z/OS v4 and the Language Environment PL/I runtime to develop support for PL/I condition handlers and transferred to LzLabs load modules containing code fragments and data blocks relevant to PL/I condition handlers in breach of clauses 4.1, 4.1.1(a), 4.1.3(a) and 4.1.3(b) of the ICA.

506. The defendants' case is that these features are interfaces and/or other functionality provided by the relevant programming languages; as such, they are not ICA Programs and Winsopia was entitled to study, test and observe them pursuant to rights conferred by Article 5(3) of the Software Directive.

507. Condition handling is a feature of the PL/I programming language that allows a customer application to specify, by the inclusion of "ON" statements, what action should be taken in response to events and errors that might occur during execution of the program.

508. During compilation, the PL/I compiler generates machine code CSECTs in the application representing the presence of condition handlers of a particular type. This code is used to initialise what I will refer to as the "I" and "C" data structures in the dynamic storage area of the runtime. These structures are populated with values representing the PL/I condition handlers that feature in the load module. Another area of the runtime contains pointers to what I will refer to as the "A" data structures, a set of chained structures, each one representing a different ON statement invoked by the application during runtime. When a qualifying condition occurs, the runtime cycles through the "A" data structures to find a relevant statement that matches the condition.

509. Mr Jaeger's evidence is that the SDM's support for PL/I condition handling was primarily written by Tim Sneddon, a PL/I developer at LzLabs. Mr Jaeger states in his second and third witness statements that in order for the SDM to provide alternative support for a PL/I application compiled and link-edited to run on an IBM mainframe, it needed to be able to locate and identify PL/I conditions in the load module. Winsopia produced PL/I compiler listings and load modules through the DR system, which were used to understand and document data structures and constants used in the process. In particular, PL/I compiler listings, containing pseudo-assembly language representation of the "I" and "C" code, were studied to determine the function, variable location and means by which the "I" structure could be identified, the layout for the "C" structure and the constant values or signal codes for both. Further, the "I" initialisation routine in load modules was studied to understand how the relevant area in memory was generated.

510. Mr Jaeger's evidence as to the source of LzLabs' understanding of the "A" structure was unsatisfactory. In his second witness statement, Mr Jaeger stated that the layout of the "A" structure was identified from sample materials in the z/OS Language Environment Debugging Guide, rather than through DRs. However, he was unable to identify the relevant part of the guide that contained such information or explain in evidence how such information was obtained. Although published IBM documentation makes reference to the relevant structures, it does not disclose full or detailed information about location or layout of the same, as is agreed by the experts. In his third witness statement, Mr Jaeger stated that the compiler listings contained references to the "A" structure but he accepted in cross-examination that this was incorrect.

511. The discoveries by LzLabs were reflected in the SDM source code file, which replicates, at least in part: (i) the means by which the "I" and "C" structures are located at runtime; (ii) the layout and structure of the "I" and "C" data structures; and (iii) the role of the "A" structure and its relationship with the "I" and "C" structures. In cross-examination Mr Jaeger agreed that the layout and fields of the "I" data structure were exactly replicated in the SDM.

512. Professor Donaldson agreed in his evidence that it would have been very difficult for LzLabs to derive these details from the compiler listings.

513. The experts' third joint statement includes the following agreed statements:

i) The interaction between the compiler-generated code and the PL/I runtime, and the structures that are involved, are not fully publicly documented, and certain aspects of this interaction are not publicly documented at all.

ii) For the SDM to support the execution of compiled PL/I executables that make use of condition handling, without requiring recompilation, LzLabs needed to understand details of the interaction between compiled binaries and the IBM PL/I runtime.

iii) LzLabs discovered relevant details and implemented support for PL/I condition handling via a series of discovery requests, mainly during the period 2015-2018. The SDM source code references several of these discovery requests.

iv) An important source of information about the details of PL/I condition handling came from compiler listings that were provided as attachments to DRs.

v) LzLabs' understanding of how condition handling is supported by the IBM runtime is not complete.

vi) The SDM features various data structures that are structurally similar to corresponding data structures in the IBM PL/I runtime that relate to condition handling.

vii) The SDM source code related to PL/I condition handling does not reproduce source or object code from modules of the IBM PL/I runtime.

viii) There is no evidence that LzLabs obtained information related to PL/I condition handling from formatted dumps.

514. The disputed issues are:

i) whether the PL/I condition handling structures, "I", C" and "A" fell within the definition of an ICA Program for the purpose of the ICA;

ii) whether Winsopia reverse engineered part or all of the "I", "C" and "A" structures and the associated Language Environment runtime in breach of clause 4.1.3(a) of the ICA;

iii) whether Winsopia's actions fell within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive;

iv) whether Winsopia's transfer of the compiler listings and load modules to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

515. IBM's case is that the PL/I compiler is part of Enterprise PL/I for z/OS v4 and the IBM Language Environment runtime is part of z/OS. As such, they are components falling within the definition of an ICA Program.

516. The defendants submit that none of the data structures falls within the definition of an ICA Program. They are aspects of the PL/I programming language condition handling feature. As such, they are inherently unprotectable data formats and/or programming language features. The object code output by the PL/I compiler gives effect to the programmer's choice of condition handlers in their PL/I source code, their bespoke source code for implementing specific condition handlers, and any custom conditions specified by the programmer. The specific values put into these data structures will reflect the programmer's decisions. The data structures are not alleged by IBM to be supplied in any particular form by IBM to Winsopia. Rather: Structures "I" and "C" are generated dynamically, in response to instructions being given by Winsopia to the PL/I compiler, causing the compilation of test programs; and structure "A" is populated dynamically.

517. I reject the defendants' submission. The "I" and "C" data structures are code generated by the PL/I compiler; the "A" data structure is present in the runtime. It is correct that the specific values generated by the compiler give effect to the programmer's choice of condition handlers in the PL/I source code but that is true of all IBM mainframe processing. The fact that the structures are generated dynamically during compilation or at runtime does not detract from their characterisation as components of the PL/I compiler and Language Environment runtime. As such, each structure is an ICA Program within the meaning of the ICA.

518. Winsopia used compiler listings generated by test programs to disclose the pseudo-assembly language representing the object code generated by the IBM PL/I compiler. The purpose was to expose the location, layout, structure and inter-relationship of the "I", "C" and "A" structures. That amounted to reverse engineering of the PL/I compiler. It is not clear what method was used to obtain an understanding of the "A" data structure. Professor Donaldson confirmed in cross-examination that he agreed with Mr Swanson's conclusion in his first report that the "A" structure was not referenced in compiler listings. Mr Swanson agreed in cross-examination that the "A" structure for an older version of PL/I was documented in a 1973 IBM Manual but Professor Donaldson agreed in cross-examination that, although it represented the same concept, the layout, flag and field names, and overall size of that structure were different to the "A" structure under consideration in this case. The defendants' suggestion that the source could have been the IBM published sample dumps does not stand up to scrutiny because, as Professor Donaldson accepted in cross-examination, the sample dumps relied on did not give any explanation as to the meaning of the disclosed fields, the meaning of the values in the fields, the fact that the data areas related to PL/I condition handlers or their function. In cross-examination, Mr Swanson accepted that LzLabs could have deduced the detail of the "A" structure and constant values from a combination of what was published and observation of compiler listings. Professor Donaldson stated that this was "plausible" but put it no higher than that. Considered in the light of Mr Jaeger's unsatisfactory evidence on this issue, the evidence as to the method used to understand the "I" and "C" structures, and in the absence of a plausible explanation from LzLabs in the Git comments, it is likely that this was also the result of reverse engineering.

519. The defendants' case is that Winsopia's actions amounted to permitted observation, study and testing. The use of test programs to generate compiler listings in order to determine the underlying ideas and principles relating to the relevant data structures contained within compiled customer applications.

520. I reject that submission. Winsopia's production of the compiler listings disclosed the pseudo-assembly language representing the object code generated by the IBM PL/I compiler. This enabled Winsopia to follow each instruction in sequence to gain an understanding as to the compiler-generated code inserted into a PL/I load module by the IBM PL/I compiler to implement language features used in the PL/I source code of the application. That concerned expression of the program, rather than its functioning.

521. Winsopia transferred to LzLabs pseudo-assembler code generated by the IBM PL/I compiler, together with load modules, containing data blocks and code fragments inserted into the application code by the PL/I build toolchains. This amounted to breach of clause clauses 4.1, 4.1.2(b) and 4.1.3(b) of the ICA.

522. In summary, on this issue:

    i) The PL/I condition handling structures, "I", C" and "A" fell within the definition of an ICA Program for the purpose of the ICA.

ii) Winsopia reverse engineered part or all of the "I", "C" and "A" structures and the associated Language Environment runtime in breach of clause 4.1.3(a) of the ICA.

iii) Winsopia's actions did not fall within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

iv) Winsopia's transfer of the compiler listings and load modules to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

Reverse engineering through the systematic use of traces, dumps, slip traps, packet sniffing and other debugging tools techniques – summary of the dispute

523. IBM's case is that Winsopia made systematic and extensive use of diagnostic and debugging tools for the purpose of investigating and analysing components of IBM mainframe software and their interactions, amounting to reverse engineering; further, that Winsopia provided the results of such reverse engineering to LzLabs for use in developing the SDM in breach of the ICA.

524. The defendants' case is that Winsopia used such tools to understand the behaviour of customer applications on the mainframe and to observe, study and test interface information relating to the interfaces of the customer applications with the IBM runtime environments, pursuant to its rights under Article 5(3) of the Software Directive.

Item 13: CICS-to-CICS communications (Paragraph 28.1 of the Technical Particulars)

525. An IBM mainframe can support multiple CICS regions (memory areas), which can be configured to work together, where a mainframe user requires additional capacity for its CICS transactions. The CICS Transaction Gateway ("CICS TG") is software that allows a remote application to invoke services in a CICS region. Communications can occur between two regions of CICS executing on different mainframes, or between a CICS region and a non-CICS region.

526. Where a CICS region is required to communicate with other CICS or non-CICS systems that are not in the same operating system, the connection is effected by using either the Systems Network Architecture ("SNA") protocol, or a TCP/IP protocol. Logical Unit 6.2 ("LU6.2") is an IBM communications protocol, which is used to communicate between two systems using the SNA. IP interconnectivity ("IPIC") is a protocol which enables use of a TCP/IP network for intercommunication between systems.

527. LzLabs developed a substitute for CICS on the SDM, through software referred to as "LTE" or "LzOnline". In order to support communication between a CICS region running on the SDM and a CICS region running on a mainframe, the SDM needed to be able to receive messages from, and send messages to, the mainframe-based CICS regions, using the same network protocols, headers, handshakes and data format as used by CICS.

528. IBM has not published the method that CICS uses to encapsulate the input parameters and output response of an EXEC CICS command that is specified

to be executed on a remote CICS region, or details of the data area used for encapsulation of the relevant parameters, the private headers.

529. It is common ground, as recorded in the experts' second joint statement, that Winsopia compiled and executed test programs so as to generate CICS-to-CICS network traffic, using tracing tools to record such traffic.

530. Winsopia used VTAM buffer traces to record the network traffic over SNA protocols, using GTF to store the data. Virtual Telecommunications Access Method ("VTAM") is a component of the z/OS communication server which provides communication services using SNA based protocols. VTAM is used to carry network data between pairs of CICS regions using LU6.2. General Tracing Facility ("GTF") records the processing or flow of a program or transaction, including network communications, memory content and the interactions between programs and transactions. The VTAM buffer traces included byte streams which were used to identify the data structures, message formats and the constants defined by CICS.

531. Winsopia also used TCP/IP packet traces to record network traffic over TCP/IP protocols. Component trace ("CTRACE") is a z/OS diagnostic utility that can capture in real time detailed technical information about the operation of components of the IBM mainframe software. TCP/IP packet trace uses CTRACE to store the trace data.

532. Winsopia sent the trace data from CICS-to-CICS network traffic to LzLabs. LzLabs used it to develop a solution whereby the SDM could communicate with a z/OS based CICS as if the SDM were another CICS region.

533. IBM's case is that the systematic collection by Winsopia of GTF, VTAM buffer and TCP/IP packet traces constituted reverse engineering of CICS transaction server for z/OS v5. Further, provision of the results of such traces to LzLabs was transfer outside enterprise, use outside enterprise and failure to prevent unauthorised use, in breach of the ICA.

534. The defendants' case is that Winsopia's analysis was necessary to understand the data format of messages sent between different CICS regions during CICS-to-CICS communications, so that it could understand the network communication protocol that the SDM would need to support in order to interoperate with CICS regions.

535. The issues in dispute are:

i) whether Winsopia's analysis of the CICS-to-CICS network traffic fell within the definition of an ICA Program for the purpose of the ICA;

ii) whether Winsopia's use of tracing tools constituted reverse engineering in breach of clause 4.1.3(a) of the ICA;

iii) whether Winsopia's supply of the trace data to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA;

iv) whether Winsopia's actions fell within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive;

v) whether Winsopia's analysis was necessary in order to achieve interoperability of the SDM with CICS regions and, as such, was permitted by Article 6 of the Software Directive.

536. IBM's case is that Winsopia's analysis constituted reverse engineering of the CICS Transaction Server for z/OS v5, an ICA Program.

537. The defendants' case is that Winsopia's analysis concerned the private header fields used by CICS forming part of the packet structure of messages sent between CICS regions. The content of the messages can be distinguished from the format of the messages. The format is interface information and not an ICA Program.

538. I accept that Winsopia was not interested in the content of the messages but it did not confine its investigation to establishing the nature or function of the interface between CICS regions. The components examined, namely the byte streams disclosed in the VTAM buffer traces, had as the object of their analysis, the sequence of steps and rules by which CICS Transaction Server for z/OS v5 effected CICS-to-CICS communications. As such, it concerned an ICA Program.

539. Winsopia used the GTF facility to collect VTAM buffer traces that disclosed details of the data structures, message formats and the constants defined by CICS. As Mr Taylor agreed in cross-examination, the purpose of the buffer trace analysis was to identify the precise message formats and the constants that CICS defines for messages. This enabled Winsopia to ascertain the combination and sequence of instructions, and rules selected by the creators of the CICS-to-CICS communications. Mr Swanson and Mr Stephens agreed in their evidence that information within the private headers is not documented. This analysis amounted to reverse engineering of these component parts of CICS.

540. It is said by the defendants that the data formats, headers and protocols were simply interface information exchanged between two CICS regions for the purposes of communicating. That description does not accurately reflect the fact that the byte streams exposed by Winsopia identified the precise data structures, message formats and constants by which the CICS Transaction Server effected such communications.

541. It is common ground that Winsopia sent the buffer trace data to LzLabs for use in development of the SDM. Although some of the information was redacted, Mr Taylor accepted in cross-examination that the streams of bytes could be read by anyone who could read the EBCDIC format. Further, during his secondment from LzLabs to Winsopia, Mr Taylor had access to the traces on the mainframe without any redaction, which he used to develop code on the SDM:

> "A. … I did not have unfettered access. There was security rules in place that would not allow me to do certain things, change certain datasets, for example. But to all intents and purposes, I

> could use the mainframe tools to carry out the testing in the area in which I was allowed to carry out testing it.
>
> …
>
> Q. But what you did have is the access that you needed to do whatever you wanted to do?
>
> A. Yes, that is true.
>
> Q. And you could see raw GTF traces, so no reductions, no legal input?
>
> A. Correct.
>
> …
>
> Q. Another advantage is you were able to work on developing the SDM code simultaneously with your mainframe access, weren't you?
>
> A. Yes I was.
>
> Q. … and that allowed you to look at traces and make changes as a direct and immediate result of what you were seeing on the mainframe?
>
> A. Yes it is.
>
> Q. And to be clear, it wasn't just configuration files that you were tweaking on the SDM, you were making substantive changes to SDM source code while at Winsopia, weren't you?
>
> A. Yes I was. ”

542. The defendants submit that Winsopia’s activities fell within Article 5(3) of the Software Directive. Winsopia was entitled to compile, load and run test programs on the mainframe, cause network transmissions to occur between two CICS regions it had set up and perform traces using the tools supplied by IBM, and storing, loading and displaying the results of those traces. I reject that argument. There is no suggestion that Winsopia used the tracing tools for the purpose they were designed, namely, debugging. The format, sequence and combination of hexadecimal code exposed by Winsopia enabled it to gain an understanding as to the detailed design choices used in the CICS Transaction Server to implement the CICS-to-CICS communications. That concerned expression of the program, rather than its functioning.

543. The Article 6 exception in the Software Directive is not applicable. Mr Taylor’s evidence was that LzLabs sought to develop LTE/LzOnline so that it would enable customer applications written for CICS to run on the SDM without the need for recompilation and to simulate the behaviour of a CICS region when it communicates with a mainframe CICS region. As Mr Stephens accepted in

cross-examination, it was open to LzLabs to adopt a solution based on recompilation of customer applications. Therefore, Winsopia's analysis was not necessary in order to achieve interoperability of the SDM with CICS.

544. In summary on this issue:

i) Winsopia's analysis of the CICS-to-CICS network traffic fell within the definition of an ICA Program for the purpose of the ICA.

ii) Winsopia's use of tracing tools for this analysis constituted reverse engineering in breach of clause 4.1.3(a) of the ICA.

iii) Winsopia's supply of the trace data to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

iv) Winsopia's actions did not fall within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

v) Winsopia's analysis was not permitted by Article 6 of the Software Directive.

Item 14: AMBLIST analysis of CICS Stubs (Paragraph 28.2 of the Technical Particulars)

545. AMBLIST is a batch utility provided with z/OS that can provide information about a load module. Such information includes the parameters and options used when creating the load module, a list of the CSECTs in the load module and the assembler used to create the load module. It can be used to print a formatted listing of the contents of the load module, including the object code and EBCDIC text, but it cannot disassemble the object code.

546. On 28 January 2016 Mr Taylor of LzLabs opened DR-1502, requesting Winsopia to use a tool such as AMBLIST on CICS stub modules that form part of the CICS Transaction Server to identify the entry points and their corresponding offsets used in COBOL, PL/I and Assembler programs. Four CICS stubs were analysed by Winsopia. Mr Lynch stated in his witness statements that he would have run AMBLIST on the IBM CSECTs to respond to the DR.

547. The experts agree that Winsopia used AMBLIST to derive information about the size of the CICS stubs, the entry points of the stubs, their offsets within the stub modules, the addressing mode specified and eyecatchers associated with the stubs. From this information, Winsopia created its own replacement CICS stubs, which were included in the load modules sent to LzLabs.

548. IBM's case is that Winsopia analysed the CICS stubs with AMBLIST in order to create replacements for these stubs, amounting to reverse engineering a component part of the CICS Transaction Server.

549. The defendants' case is that the CICS stubs are not ICA Programs, there was no reverse engineering and Winsopia's acts fell within the scope of its rights under Article 5(3) of the Software Directive.

550. The issues in dispute are:

i) whether Winsopia's analysis of the CICS stubs fell within the definition of an ICA Program for the purpose of the ICA;

ii) whether Winsopia's use of AMBLIST constituted reverse engineering in breach of clause 4.1.3(a) of the ICA;

iii) whether Winsopia's actions fell within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

551. The IBM CICS stubs were generated by, and form component parts of, the CICS Transaction Server. The defendants submit that they are not ICA Programs because they are not self-contained components of CICS and are interfaces containing none of the commercially valuable processing of CICS. I reject that argument. The CICS stubs are an essential part of the CICS Transaction Server process, without which the CICS services would not be called at runtime. For the reasons set out above, I accept that they fall within the definition of ICA Programs within the meaning of the ICA.

552. Although AMBLIST is available as a debugging tool, Winsopia did not use it for such purpose. It was used to expose the names, lengths and offsets of the IBM CSECTs in the load modules, so that Winsopia could provide placeholders for LzLabs to insert equivalent replacement CICS stubs. This amounted to reverse engineering of the CICS Transaction Server v5.

553. The defendants submit that Winsopia's activities fell within Article 5(3) of the Software Directive on the ground that Winsopia was entitled to investigate interface information relating to the CICS stubs. I reject that argument. There is no suggestion that Winsopia used AMBLIST for the purpose it was designed, namely, debugging. The information produced about the load modules enabled Winsopia to gain an understanding as to the sequence and combination of code used by the CICS Transaction Server to implement the CICS services. That concerned expression of the program, rather than its functioning.

554. In summary on this issue:

i) Winsopia's analysis of the CICS stubs fell within the definition of an ICA Program for the purpose of the ICA.

ii) Winsopia's use of AMBLIST constituted reverse engineering in breach of clause 4.1.3(a) of the ICA.

iii) Winsopia's actions did not fall within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

Item 15: Colesoft z/XDC and COBOL initialisation (Paragraph 28.3 of the Technical Particulars)

555. The allegation is that Winsopia used the z/XDC tool to analyse the COBOL v 4.2 and COBOL v 5.2 compilers, amounting to reverse engineering of the same. The defendants' case is that the compilers are not ICA Programs and the

analysis was permitted observation, study and testing under Article 5(3) of the Software Directive.

556. As set out above, z/XDC is a commercially available debugging tool that enables a breakpoint to be set at a specific point in a load module, stopping execution and transferring control to XDC. It allows the user to step through the code examining the execution of each instruction, view data areas and registers, disassemble object code and display dumps of memory.

557. Mr Lynch's evidence is that Mr Rastall requested him to investigate and report on how executable load modules compiled under the COBOL compiler are constructed. The first report was produced on about 6 August 2015 in respect of the COBOL compiler v4.2 and included the following:

> "This document contains the findings from the analysis of the object code generated by the COBOL compiler.
>
> This analysis was undertaken in support of the Load Module Compiler project which aims to write a product to analyse COBOL load modules and construct new C language source code based upon the contents of the COBOL load module.
>
> All of the analysis of the COBOL compiler's object code was done using COBOL 4.2. Other versions and releases may produce different structures or generate different object code.
>
> Research into the latest COBOL compiler, version 5, has discovered a new runtime parameter … To properly analyse and understand the code this new compiler generates we will need to install the latest version of the compiler, version 5.2, as soon as reasonably possible.
>
> The COBOL language contains many different language constructs not all of which have been analysed thus far. The contents of this document will almost certainly change as more programs are analysed… "

558. Mr Lynch described the methods used to analyse the compiler, namely, compiler listings together with XDC:

> "Additional analysis was done using a bespoke Assembler program that loads a specified COBOL program into storage without executing the program, and then invokes the XDC the Debugging product to analyse the COBOL program. This allows the analyst to view the COBOL program in storage and set execution break points to stop execution of the COBOL program at specified points. Once an execution break point is then reached storage contents and registers can be analysed and the COBOL program can be stepped through one instruction at a time if required."

559. In addition to a description of the compiler options, the report contained a detailed analysis of program initialisation, addresses for storage areas, base register usage and branch instruction sequences. In particular, the report included details as to:

i) COBOL initialisation routines;

ii) the COBOL entry code and various other details relevant to initialisation;

iii) COBOL data structures relevant to initialisation;

iv) the physical locations of COBOL initialisation routines; and

v) the components of the compiler-generated code relevant to initialisation, its functions and how it interacts with COBOL initialisation routines.

560. The second report was produced by Mr Lynch on 24 November 2015 and involved a similar analysis in respect of COBOL compiler v 5, including a detailed explanation of the changes between versions 4.2 and 5 of the compiler.

561. The experts' second joint statement included the following agreed statements:

i) The Lynch reports included information such as: the effect of compiler options on the object code created by the compiler; information about data areas created by the compiler in a user CSECT; and code inserted by the compiler in a user CSECT, including initialisation code.

ii) This information was obtained by analysing COBOL compiler listings and XDC, a debugging tool that can be used to step through an executing program, viewing its processing and data areas as the program executes.

iii) XDC was not used to resolve or debug a problem but to obtain information from a user program.

iv) Winsopia created a test COBOL program and used XDC to analyse the program, including code inserted by the compiler into the user CSECT.

v) There are no IBM supplied utilities designed to identify or remove code and data areas inserted by the compiler into a user CSECT.

vi) The information obtained by Winsopia was sent to LzLabs.

562. The allegation is that Winsopia used XDC to reverse engineer Enterprise COBOL v4 and v5.

563. The defendants' case is that the COBOL runtime and code generated by the compilers are not ICA Programs, there was no reverse engineering and Winsopia's acts fell within the scope of its rights under Article 5(3) of the Software Directive.

564. The issues in dispute are:

i) whether Winsopia's analysis of the COBOL compiler, versions 4 and 5 concerned an ICA Program for the purpose of the ICA;

ii) whether Winsopia's use of z/XDC constituted reverse engineering in breach of clause 4.1.3(a) of the ICA;

iii) whether Winsopia's actions fell within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

565. The Enterprise COBOL v 4 and v 5 are ICA Programs. The compilers were provided as components of the ICA Programs. For the reasons set out above, the compilers fall within the definition of ICA Programs within the meaning of the ICA. The defendants submit that Winsopia's use of XDC was limited to its analysis of the interfaces contained within a test program and the user CSECTs in that test program, none of which is an ICA Program. I reject that submission. It is clear from the Lynch reports that the analysis was not concerned with the test program; in each case, the test program was prepared so that Mr Lynch could analyse the code generated by the compiler. The purpose of such analysis was to understand the structure and content of the same, the instructions generated that are relevant to initialisation and how they interact with the initialisation routines at runtime.

566. Mr Lynch was clear in cross-examination that he used XDC, not as a debugging tool, but to examine the operation of the COBOL compilers. He viewed a disassembled representation of each instruction in sequence during program execution, including branching instructions. He used XDC to step through the initialisation code that the COBOL compiler generates and inserts into the load module to initialise COBOL applications. He analysed the interaction between the initialisation code and IBM CSECTs in the load module during initialisation. He analysed a number of runtime data structures, including how the initialisation code and IBM CSECTs manipulate these structures during initialisation. He also confirmed that XDC displays assembler code on the screen. This amounted to disassembly and reverse engineering of COBOL v4 and v5.

567. The defendants submit that Winsopia's activities fell within Article 5(3) of the Software Directive on the ground that Winsopia was performing characterisation testing of two versions of the COBOL compiler: to understand how different compiler options affected the object code that was output; how the resulting user CSECTs in customer application are structured; and to study the initialisation interfaces. I reject that argument. It is clear from the reports that Mr Lynch carried out a detailed analysis of the execution of each instruction within the load module. Mr Stephens agreed in cross-examination that Mr Lynch was using XDC to study the initialisation process. Such analysis investigated, not just the output of the program but how the program achieved its output, that is expression of the program, rather than its functioning.

568. In summary on this issue:

i) Winsopia's analysis of the COBOL compiler versions 4 and 5 concerned an ICA Program for the purpose of the ICA.

ii) Winsopia's use of XDC constituted reverse engineering in breach of clause 4.1.3(a) of the ICA.

iii) Winsopia's actions did not fall within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

Item 16: XDC and IMS (Paragraph 28.4 of the Technical Particulars)

569. The allegation is that Winsopia used the XDC tool to analyse the IMS ACB data structure, amounting to reverse engineering of the same. The defendants' case is that the IMS modules analysed are not ICA Programs and the analysis was permitted observation, study and testing under Article 5(3) of the Software Directive.

570. IMS is a hierarchical database system which supports transaction processing. The databases and the information that is stored in them must be predefined using database descriptors ("DBDs") that are coded as assembler macro statements. Although it is a load module, a DBD is not an executable program; it contains metadata that is used by IMS to document the format of data in the user's database. Before an IMS database can be accessed by a program, the program must also be defined to IMS, using assembler macros. This is processed by an IBM utility to generate a program specification block ("PSB"), which describes the characteristics of the program, including the data structures required. When a program executes in IMS, IMS must obtain the DBD and PSB, and merge them into an application control block ("ACB"). ACBs are load modules created using an IBM utility, ACBGEN. ACBGEN comprises a number of modules, including DFSUAMB0 and DFSUACB0, which are supplied with IMS as part of the SDFSSRC library.

571. Winsopia needed to develop support for IMS databases in CPX to facilitate the export of customer data from IMS databases. On 12 August 2016 Mr Palmer at Winsopia contacted Mr Harper, a developer at LzLabs, to discuss potential methods of identifying the DBD libraries required by the IMS unload utility for the purpose of exporting data using CPX.

572. Following this exchange, Mr Palmer wrote:

> "I was able to track down and build the [ACBGEN] utility so I have access to the listings etc where the key write module is DFSUAMB0. I did stick an XDC hook in there and was able to identify where the DBD and PSB ACB library members are written and discovered (I had mistakenly thought there would be some sort of ACB control block) that the DBDs are written as a DMB (Data Management Block) that is mapped in the DFSDMB macro. There are a number of different mapping DSECTs in there but so far it looks good."

573. Subsequently, Mr Palmer sent an email to Mr Harper, copied to Mr Payne and Mr Rastall, explaining the work that he had undertaken:

> "Compiled and linked IMS code to the extent that I can insert an XDC hook in the code that performs an ACBGEN; thus allowing a better understanding of the critical ACB member attributes I will need to handle."

574. In his third witness statement, Mr Palmer explained what he was trying to achieve:

> "I was seeking to examine the application control blocks that contain definitions for customer IMS databases. These database definitions are stored in load modules generated by the user as part of the IMS system generation process. I needed to understand the member attributes of these definitions so that I could determine how to "unload" (export) the customer databases as part of the CPX migration process. In order to examine these definitions, I had to first locate them within the load modules."

575. Mr Palmer gave the following account as to what he did in his written and oral evidence. First, he located DFSUAMB0, by reviewing the source code of IMS modules which had been supplied to Winsopia as part of IMS. Second, he compiled and link-edited the DFSUAMB0 source code into a load module. Third, he used an Assembler listing to identify the particular offset that he was interested in. Fourth, he inserted an XDC hook at the point of interest, thereby modifying the program binary so that it would invoke and pass control to XDC during execution. Fifth, he executed the modified version of the DFSUAMB0, which was paused at the point that it was about to write the DBDs to a load module. Sixth, he used XDC to inspect the working memory and registers to identify the location of the DBD and ACB library members so that he could understand the attributes of such member definitions.

576. The experts' second joint statement includes the following agreed statements:

i) Winsopia were developing the CPX component to migrate IMS databases to the SDM. To do this, they decided to use an IMS utility that required the DBDs defined for the databases being processed to be input. ACBs include DBDs and Winsopia investigated if the ACBs could be used. To do this, Winsopia needed to understand the format of the IMS ACBs.

ii) Winsopia assembled the DFSUACB0 and related DFSUAMB0 source and used z/XDC to analyse their operation. From this analysis, it was determined that macros supplied with IMS in the SDFSMAC library could be used in a user program (CPX) to process ACBs.

iii) XDC was used to analyse and study DBDs and other static objects, such as an ACB or DBD. XDC was not used to analyse any IBM supplied code or programs.

iv) This information was used in the development of CPX.

577. The issues in dispute are:

i) whether Winsopia's analysis of the DFSUAMB0 module concerned an ICA Program for the purpose of the ICA;

ii) whether Winsopia's use of z/XDC constituted reverse engineering in breach of clause 4.1.3(a) of the ICA;

iii) whether Winsopia's use of z/XDC was in breach of clause 4.1.1(d) of the ICA;

iv) whether Winsopia's actions fell within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

578. IMS v14 is an ICA Program. The ACBGEN utility, including the DFSUAMB0 and DFSUACB0 modules, were provided as components of IMS. Those modules are supplied to IMS users in source form as part of the SDFSSRC library. For the reasons set out above, they fall within the definition of ICA Programs within the meaning of the ICA.

579. The defendants submit that the allegations relate to the use of XDC on modified versions of DFSUACB0 and DFSUABM0, which were created and compiled by Winsopia and were not ICA Programs. I reject that argument. Regardless of whether Winsopia modified the modules as part of its analysis of the same using XDC, they were component parts of IMS and, therefore, constituted an ICA Program.

580. Mr Palmer's evidence established that Winsopia modified the DFSUACB0 and DFSUABM0 modules by inserting an XDC hook, using it to stop execution at the selected point, so as to display in assembly form the material working memory and registers, from which he could identify the location of the DBD and ACB library members. This amounted to reverse engineering of components of IMS.

581. It is common ground that the DFSUACB0 and DFSUABM0 modules were supplied by IBM in source code form. It is not suggested by the defendants that the analysis carried out on the modules was for the purpose of resolving problems related to use of IMS. Although the defendants seek to argue that the analysis was to modify IMS so that it would work with other projects, Mr Palmer's evidence did not support such a case. His investigation was not to modify IMS to enable it to work with other projects; rather, it was to enable CPX to extract and export data from IMS to the SDM. He confirmed in cross-examination that the purpose of his work was to automate the process of matching up IMS database definitions to their corresponding databases, so as to speed it up. It follows that Winsopia's use of the modules was outside permitted use, in breach of clause 4.1.1(d) of the ICA.

582. The defendants submit that Winsopia's activities fell within Article 5(3) of the Software Directive on the ground that the purpose was to understand interface information relating to the ACB data structure. I reject that argument. It is clear from Mr Palmer's evidence that his examination of the working memory and

registers in the modules during execution was to understand how IMS defined databases. Such analysis investigated, not just the output of the program but how the program achieved its output, that is expression of the program, rather than its functioning.

583. In summary on this issue:

i) Winsopia's analysis of the DFSUAMB0 module concerned an ICA Program for the purpose of the ICA.

ii) Winsopia's use of z/XDC constituted reverse engineering in breach of clause 4.1.3(a) of the ICA.

iii) Winsopia's use of z/XDC was outside permitted use in breach of clause 4.1.1(d) of the ICA.

iv) Winsopia's actions did not fall within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

584. This use of XDC to interrogate IMS was not an isolated incident, as evidenced by the email dated 16 March 2018 sent by Bob Maddison of Winsopia to Richard Parkinson, in which he referred to the practice of using XDC to "rummage around" in IMS. The court accepts that XDC was used by a number of Winsopia employees to interrogate the ICA Programs.

Additional examples

585. IBM seeks to rely on additional examples considered by the experts in their second joint statement, to demonstrate that Winsopia used XDC to analyse various different elements of mainframe software. However, although IBM pleaded a general allegation of extensive use of XDC, it was required by the Court to set out in its pleadings full particulars of the technical breaches alleged. The additional examples now identified are not pleaded allegations and it is not necessary for the court to resolve them for the purpose of determining the key issues in the case.

Item 17: SLIP Traps and CICS (Paragraph 28.5 of the Technical Particulars)

586. IBM's case is that Winsopia used the SLIP trap diagnostic tool to reverse engineer CICS data structures.

587. SLIP (Serviceability Level Indication Processing) is a diagnostic aid that can be configured to intervene during and interrupt the execution of a program and trigger a specified action, typically, a memory or system dump, enabling the user to have a snapshot of a system state or storage value at the time of the event, a Program Event Recording ("PER").

588. The experts agree that PER SLIP traps have a performance impact on the operating system and therefore are used sparingly. Further, as SLIP traps can affect z/OS processing, they are usually set only by the z/OS administrator (systems programmer).

589. On 29 June 2014, LzLabs opened DR0111, requesting Winsopia to identify the four byte value of a data area created and stored in a register by CICS, at the moment that control is passed to a COBOL program during execution. It was suggested that Winsopia could use SLIP to obtain this information.

590. Simon Payne of Winsopia carried out the requested task by loading a COBOL program into CICS storage and setting a PER IF SLIP trap to produce a dump at the required offset when executing. Analysis of the dump enabled him to identify the value of the four byte STKLANG field in the register save area at that point and provide the information to LzLabs.

591. The issues in dispute are:

    i) whether Winsopia's analysis of the SLIP trap dump concerned an ICA Program for the purpose of the ICA;

    ii) whether Winsopia's use of the SLIP trap constituted reverse engineering in breach of clause 4.1.3(a) of the ICA;

    iii) whether Winsopia's actions fell within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

592. IBM's case is that Winsopia's analysis involved either CICS Transaction Server v5 or Enterprise COBOL v4, and their associated runtimes. Both are ICA Programs. The defendants' case is that the register save area is an interface; a data area that interacts with the COBOL Language Environment. As such is it not an ICA Program.

593. Mr Swanson's evidence is that CICS is responsible for creating the register save area when control is passed from CICS to a COBOL program. Mr Stephens agreed in cross-examination that the data area is populated either by CICS or the Language Environment runtime. As submitted by IBM, both are ICA Programs.

594. Winsopia used a SLIP trap to produce a dump at the required offset so that it could inspect the register save area and identify the value of the four byte field at a specific point during execution. This interrogated the internal implementation of the operation of the CICS Transaction Server and amounted to reverse engineering. As Mr Stephens agreed in cross-examination, although the format of the register save area is published by IBM, the values that occupy the field in the data area at different times during runtime execution and the purposes served by such values are not publicly documented.

595. The defendants submit that Winsopia's activities fell within Article 5(3) of the Software Directive on the ground that the purpose was to observe the input that is passed to the test program from CICS, at the point when the test program receives control, while it is executing. I reject that argument. The investigation went beyond mere observation, study or testing the input and output of the program. It entailed a detailed examination as to how control is passed between CICS and COBOL applications during execution. That concerned expression of the program, rather than its function.

596. In summary on this issue:

i) Winsopia's analysis of the SLIP trap dump concerned an ICA Program for the purpose of the ICA.

ii) Winsopia's use of the SLIP trap constituted reverse engineering in breach of clause 4.1.3(a) of the ICA.

iii) Winsopia's actions did not fall within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

Item 18: SLIP Traps and COBOL (Paragraph 28.6 of the Technical Particulars)

597. IBM's case is that Winsopia used SLIP traps to reverse engineer COBOL data structures.

598. On 2 July 2014, LzLabs opened DR-0127, requesting Winsopia to compile and link-edit four test programs, and to identify register values and COBOL structures at the point at which one of the modules calls another during program linkage.

599. Mr Lynch (or another employee of Winsopia) carried out the requested task by running the test programs, setting up a SLIP trap to produce a dump at the entry point to the module. Analysis of the memory content disclosed by the dump enabled him to identify the register values as initialised by the COBOL runtime environment on entry to the program. This information, together with compiler listings and redacted load modules were sent to LzLabs.

600. The issues in dispute are:

i) whether Winsopia's analysis of the SLIP trap dump concerned an ICA Program for the purpose of the ICA;

ii) whether Winsopia's use of the SLIP trap constituted reverse engineering in breach of clause 4.1.3(a) of the ICA;

iii) whether Winsopia's actions fell within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

601. The pleaded case invited the court to draw an adverse inference that load modules were sent unscrubbed to LzLabs based on heavy redactions to the DR response. The redacted parts of the DR response are subject to a claim for privilege which has not been challenged. In those circumstances, it would not be appropriate for the court to draw such an inference. Further, in his second report, Mr Stephens examined the file that was attached to the DR response (with the assistance of Professor Donaldson and Mr Wilkinson). From that examination, he was able to confirm that the file sent to LzLabs had been scrubbed.

602. IBM's case is that Winsopia's analysis involved Enterprise COBOL v4 and its associated runtime, an ICA Program. Mr Swanson's evidence is that the focus of Winsopia's analysis was how the IBM COBOL runtime provides support for

COBOL CALL statements, used to request services from another COBOL application. The defendants' case is that the analysis was carried out in respect of test programs which are not ICA Programs. It is clear that Winsopia was not asked to investigate the test programs; it was asked to use the test programs to analyse the way in which control is passed from one COBOL program to another in the Language Environment. This involved analysis of Enterprise COBOL v4 and its runtime, an ICA Program.

603. Winsopia used a SLIP trap to produce a dump during execution of a test program so that it could inspect the registers and memory as initialised by the COBOL runtime environment. Mr Stephens agreed in cross-examination that the purpose of this examination was to understand the mechanism selected by the designer of Enterprise COBOL and the Language Environment for passing certain values from one program to another. This amounted to reverse engineering. Although the general role of the registers is documented publicly, that does not include the specific values used to populate the registers.

604. The defendants submit that Winsopia's activities fell within Article 5(3) of the Software Directive on the ground that the purpose was to identify interface information comprising the parameter values stored in memory registers and passed to the test program. I reject that argument. The investigation went beyond mere observation, study or testing the input and output of the program. It entailed a detailed examination as to how the Language Environment COBOL runtime passed control between COBOL programs. That concerned expression of the program, rather than its function.

605. In summary on this issue:

i) Winsopia's analysis of the SLIP trap dump concerned an ICA Program for the purpose of the ICA.

ii) Winsopia's use of the SLIP trap constituted reverse engineering in breach of clause 4.1.3(a) of the ICA.

iii) Winsopia's actions did not fall within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

Macros and Copybooks - introduction

606. Multiple programs may write a record of the same format to a file or access a common data area. Rather than requiring each program to repeat the same instructions in full, short cut commands in programming languages, such as macros and copybooks, can be used to incorporate common source code into the program.

607. A macro is a rule, pattern, or template that specifies how a certain input should be mapped to a replacement output. Macro language is an extension of assembler language. It provides a convenient way to generate the same sequence of assembly language statements many times in one or more programs. A macro instruction included in source code is automatically expanded into an assembly language program by the assembler. The macro comprises:

i) the macro definition, a set of machine-readable statements that defines the name, format and conditions for generating a sequence of language statements;

ii) the macro instruction for calling or invoking the macro, which provides the assembler with the name of the macro definition to process, and the information or values to pass to the macro definition; and

iii) the macro expansion or generation, whereby the assembler replaces the macro call with the macro definition statements and inserts them into the source module at the point of the call.

608. Macros can be used to generate executable code and also to create DSECTs, data structure templates used to define the layout of data held in memory.

609. Publicly available IBM documentation provides some information about macros, including identification and use of the macros. Such published information does not include the macro assembly code contained within the macro definition that is produced on expansion by the assembler. The experts agree that the published information is not sufficient to enable LzLabs to create the relevant SDM equivalent source code.

610. A copybook provides a similar role to a macro for COBOL programs. It contains source code which can define common data structures (or control blocks) used by a number of different application programs. A copybook is invoked by incorporating the relevant copy statement in the source code of a program. When the COBOL program is compiled, the compiler pre-processor copies the contents of the relevant copybook into the source code of the application before it is compiled.

611. The experts agree that the copybooks are partially documented publicly but variables are not identified.

612. It is common ground that the macros and copybooks were supplied by IBM to Winsopia, as licensee under the ICA, in source code form. The macros were provided as part of IMS v14, IMS v15, z/OS Base v2 or Db2 v10 for z/OS. The copybooks were provided as part of the CICS Transaction Server for z/OS v5. The experts agree that macros and copybooks are intended to be used by programmers to save time, simplify the coding process and reduce the chance of programming errors.

Macros (Paragraphs 32.1-32.9 of the Technical Particulars) – summary of the dispute

613. The allegation by IBM is that Winsopia reverse engineered the macro expansion tables and supplied the contents of the same to LzLabs. The defendants' case is that macros and copybooks are provided in source code and, when included in a customer application, become part of the customers's source code and compiled program. As such, they do not form part of any ICA Program. The information sent by Winsopia to LzLabs was not source code but rather information derived from observing, studying and testing the macros or copybooks. Further, the macros and copybooks in question define data formats

and values needed by customer applications to access and store their data, accept input from the keyboard, and send commands to other devices. Winsopia's activities were within its interoperability rights.

614. The pleaded examples relied on by IBM are considered below. In each case, there is no dispute as to the material facts.

Item 19: DR-3246 (Paragraph 32.1 of the Technical Particulars)

615. By DR-3246 dated 13 March 2018, Mr Harper of LzLabs requested Winsopia to provide the layout of fields in DFSVC000, an IMS macro definition load module. On 15 March 2018 Mr Hitchings of Winsopia sent to LzLabs a word document, containing the layout of each field in the load module.

Item 20: DR-10237 (Paragraph 32.2 of the Technical Particulars)

616. By DR-10237 dated 23 June 2014, Mr Bond of LzLabs requested Winsopia to provide the layout of the AMDSB data block, the structure of which was mapped in an IMS macro, referred to in an earlier Jira ticket, CORE-96. In response, Mr Payne of Winsopia produced a table describing each field in the data block, including length values, data types and values.

617. In cross-examination Mr Bond agreed that the information in the Winsopia table was the same as the IBM macro:

> "Q. … a plain text table is never going to look like a macro, is it?
>
> A. No.
>
> Q. But that's because a macro definition obviously has to be written in a specific parsable format and it has to conform to the macro language semantics and syntax?
>
> A. Exactly.
>
> Q. But as a matter of substance, as opposed to form, the two are the same, aren't they? This is a translation, in essence, of the macro?
>
> A. Yes, okay.
>
> Q. Well, do you agree with that?
>
> A. Yes.
>
> ...
>
> Q. It does rather look, doesn't it, as what Mr Payne has done is got the macro source up on the screen and essentially copied it into the table?

> A. It does seem likely.
>
> Q. And so, although you say -- and we looked at it a moment ago at paragraph 92 of your statement – that the Winsopia team used their own words to describe the layout, I mean, really they used IBM's words, didn't they?
>
> A. I certainly did not know that when I did my witness statement because I had not seen the IBM macro.
>
> Q. But looking at it now, do you accept that's the position?
>
> A. Yeah, I can see that, yes.
>
> …
>
> Q. … you say "I recall working on the SDM code, and I accept that the SDM code is based on and incorporates information relating to the fields in the AMDSB data structure from [the file]." That's what you say, isn't it?
>
> A. It was not written from that, but it of course mirrors it exactly.
>
> Q. That's what you meant there, is it?
>
> A. Yes."

Item 21: DR-2753 (Paragraph 32.3 of the Technical Particulars)

618. By DR-2753 dated 4 September 2017, LzLabs requested Winsopia to provide details of the CSVAPF macro definition. In response, on 6 September 2017 Mr Hitchings produced a table setting out the contents of the macro definition, including offsets, routine names, values and identifiers.

Item 22: DR-2771 (Paragraph 32.4 of the Technical Particulars)

619. By DR-2771 dated 12 September 2017, LzLabs requested Winsopia to provide the offset, length and description of all fields in the generated DSECTs for the macro definitions DFSDMBL and IDLI JCBBASE=0. In response, on 14 September 2017 Mr Hitchings sent a word document showing the expanded macro definitions, including the offsets, routine names, values and identifiers.

Item 23: DR-2796 (Paragraph 32.5 of the Technical Particulars)

620. By DR-2796 dated 18 September 2017, LzLabs requested Winsopia to provide the offset, length and description of all fields in the generated DSECTs for three further macros: DFSIPSBA, DFSIPSBB and DFSIPSBX. In response, on 20 September 2017 Mr Hitchings sent a word document showing the expanded macro definitions, including the offsets, routine names, values and identifiers.

Item 24: DR-3280 (Paragraph 32.6 of the Technical Particulars)

621. By DR-3280 dated 28 March 2018, LzLabs requested Winsopia to provide the offset, length and description of all fields in the generated DSECTs for six further macros: ICLI CLBBASE=0, ICLI CTBBASE=0, ICLI CNTBASE=0, ICLI CTTBASE=0, ICLI CIBBASE=0 and ICLI PCIBASE=0. In response, on 2 May 2018 Mr Hitchings sent a word document showing the expanded macro definitions, including the offsets, routine names, values and identifiers.

Item 25: DR-4281 (Paragraph 32.7 of the Technical Particulars)

622. By DR-4281 dated 19 August 2019, LzLabs requested Winsopia to provide the offset, length and description of all fields in the generated DSECTs for four further macros: DSPDBHRC, DSPPTNRC, DSPDSHRC and DSPDGRC. In response, on 15 August 2019 Mr Hitchings sent a word document showing the expanded macro definitions, including the offsets, routine names, values and identifiers.

Item 26: DR-4322 (Paragraph 32.8 of the Technical Particulars)

623. By DR-4322 dated 28 August 2019, LzLabs requested Winsopia to provide the offset, length and description of all fields in the generated DSECTs for further macros: DFSURGUF and DFSURGUP. In response, on 29 August 2019 Mr Hitchings sent a word document showing the expanded macro definitions, including the offsets, routine names, values and identifiers.

Item 27: DR-0847 (Paragraph 32.9 of the Technical Particulars)

624. By DR-0847 dated 26 March 2015, David Janicek of LzLabs requested Winsopia to provide the offset, length and description of all fields in the generated DSECTs for the IFCID 3 DB2 accounting trace records. In response, in May 2015 Mr Hitchings sent a word document showing the expanded macro definitions, including the offsets, routine names, values and identifiers, some of which were re-named by him.

Macros - discussion

625. It is not in dispute that in each of the above examples, LzLabs used the DR system to request Winsopia to document various IBM macros whose definitions created DSECTs. The experts agree that, in each case, it is likely that Winsopia created an assembler program that included a call to the macro in question. Winsopia obtained a listing from the assembly of the program, including an expansion of assembler code provided by the macro. From such listing, Winsopia created a word document, containing information about the data area, namely, the offset of the DSECT field, field name, field value and identifier describing the field. The document was sent to LzLabs through the DR system. LzLabs used the information received from Winsopia to create its own versions of the data structures and values in the SDM code, with an equivalent data structure to that defined by the IBM macro.

626. I accept Professor Weissman's opinion that for the SDM to reproduce the functionality of the macros, it was necessary for LzLabs to ascertain the DSECT layout of particular data areas so that the SDM could access data fields in a

manner equivalent to the IBM mainframe. Contrary to Mr Jaeger's evidence, the DSECTS generated by expansion of the above macros were not customer specific and did not contain customer data. They were keys that could be used to decipher or interpret customer application data for its use in the SDM.

627. The parties agree that each of the above allegations raises common issues of principle. The disputed issues are:

i) whether the above macros were ICA Programs within the meaning of the ICA;

ii) whether Winsopia's activities amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA or misuse of source code in breach of clause 4.1.1(d) of the ICA;

iii) whether Winsopia's supply of the word documents to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA;

iv) whether Winsopia's analysis fell within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive;

v) whether Winsopia's analysis was necessary in order to achieve interoperability of customer applications with the SDM and, as such, was permitted by Article 6 of the Software Directive.

628. IBM's case is that the macros were provided in source code as part of IMS, z/OS Base or Db2. They are machine-readable instructions and data, or related licensed materials; as such, they are ICA Programs within the ICA definition.

629. The defendants submit that the macros in question were provided to Winsopia in libraries provided with IMS, z/OS Base or Db2. Each macro is not a discrete unit so as to constitute an ICA Program. Rather, each macro is a sample program or Other IBM Program for the purpose of the ICA.

630. I reject the defendants' submission. The macros are supplied with, and form component parts of, IMS v14, IMS v15, z/OS Base v2 or Db2 v10 for z/OS. They are packaged as libraries and supplied in source code form but that does not detract from the fact that they are part of those programs. IMS v14, IMS v15, z/OS Base v2 and Db2 v10 for z/OS are ICA Programs licensed to Winsopia under the terms of the ICA. It follows that the macros fall within the definition of an ICA Program for the purpose of the ICA.

631. In each case, Winsopia created an assembler program that included a call to the macro in question in order to disclose the expanded assembler code provided by the macro. The resulting assembler listing revealed the structure of the data sets, including the offset of the DSECT field, facilitating location of the DSECT field in memory, the field name, field value, length and type. These details are not published and are not necessary for an application programmer to use the macro but they are needed to implement the DSECT. This amounted to reverse engineering of those parts of the ICA Programs.

632. The experts agree that the original IBM assembly code was not copied to the SDM, nor provided to LzLabs by Winsopia. Therefore, there was no breach of clauses 4.1, 4.1.2(b) or 4.1.3(b) of the ICA.

633. However, it is common ground that the information obtained from the expanded assembly code in the form of the source code of each of the expanded macros was sent by Winsopia to LzLabs and used in the development of at least parts of the replacement code in the SDM. The experts agree that for each of the data mapping macros there is a close correspondence between the information represented by the fields of the relevant data structures defined in the SDM code and the fields of the data structure mapped by the relevant IBM macro. Mr Bond's evidence is that it was used primarily to check information that had already been obtained from an IBM manual but that some of the information was essential for processing the data sets by the SDM. It is clear from the number of DRs seeking similar types of information, that LzLabs needed the expanded assembly code details for implementation or development of the SDM.

634. The experts agree that the IBM macro data was not replicated in the SDM. However, they also agree that in each case the layout of the SDM C data structure corresponded to the data structure defined by the IBM macro in order to map the same data area. Professor Weissman compared the IBM source code defining some of the macros with the information in the Winsopia tables and its implementation in the SDM code. From this comparison, he found a high degree of similarity between the information in the Winsopia table and the SDM implementation. Based on this comparison exercise, his view is that, although written in different programming languages, a semantic equivalence could be established between portions of the IBM DSECTs and SDM C data structures.

635. Professor Weissman clarified what he meant by semantic equivalence in cross-examination:

> "Q. … So by "copying", you mean semantic equivalence?
>
> A. I mean the same IBM materials are in the SDM.
>
> Q. No, you said "semantic equivalence"?
>
> A. That's what I mean by semantic equivalence.
>
> Q. Semantic equivalence, either you mean copying as in slavishly, which is syntactic equivalent to identity, or you mean semantic equivalence, which is it?
>
> A. As I state clearly in the report, it is not syntactic, and that is not surprising given they're different systems, I make it clear that this is semantic equivalence."

636. Professor Donaldson did not disagree with Professor Weissman in any material respect:

> "What I see here is that the SDM code has clearly been informed by the information in the table. It's also clear to me that the information in a table has arisen from study of the IBM code in whatever form. What I don't see is reproduction of the IBM code directly in the SDM source code.
>
> …
>
> In my opinion, the information about this structure has been carried across and that's reflected in the SDM source code. I do agree that names have been carried across and very similar names appear in the IBM source code … but for some of the fields, but not all of them.
>
> …
>
> When I say that none of the IBM source code is reproduced in the SDM, I do mean literally reproduced. However, what I'm not saying is that the IBM code is basically there in the SDM, it's just been tweaked to look a bit different. The code in the SDM is fundamentally different from the IBM code, being written in a different programming language with different syntax. But I do agree that the information used to create that code has evidently been derived via this table.
>
> …
>
> What we see in the table is a representation of the information, so I think it would be fair to say that the IBM code had been turned into, or rather aspects of the IBM code had been put into tabular form, and then that table has been used to inform the creation of a data structure that does have the same structural layout as the original IBM data structure."

637. The defendants submit that Winsopia's acts were observation, study and testing within Article 5(3) of the Software Directive. Winsopia was entitled to assemble a program that included a call to the relevant macro and production of an assembler listing of the macro expansion as a licensed user of its mainframe. In so doing, it is said that Winsopia was observing, studying and testing interface information, namely the data fields, lengths, offsets, data types and descriptions of the data areas.

638. That submission is rejected. The purpose of Winsopia's analysis was not confined to the functioning of the macros to determine underlying ideas and principles. It already knew the function of each macro; this was information that was publicly available. Winsopia's analysis was to ascertain details of the DSECT field in memory, the field name, field value, length and type so that the same details could be replicated or mirrored in the SDM. That amounted to expression of the program, rather than its functioning.

639. The Article 6 exception in the Software Directive is not applicable. It was open to LzLabs to adopt a solution, whereby they designed their own macros and copybooks, using different data structures, and recompiled customer source code applications using the replacement macros and copybooks. Therefore, Winsopia's analysis was not necessary in order to achieve interoperability of customer applications with the SDM. Further, although the replacement code mapping in the SDM was written in a different programming language to that used in the ICA Programs, it used Winsopia's translation of the macro code to achieve semantic equivalence in the SDM code. That was substantially similar in expression to the IBM mapping of the data areas for the purpose of Article 6(2)(c).

640. In summary on this issue:

i) The above macros were ICA Programs within the meaning of the ICA.

ii) Winsopia's activities amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA and/or misuse of source code in breach of clause 4.1.1(d) of the ICA.

iii) Winsopia's supply of the word documents to LzLabs did not constitute breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

iv) Winsopia's analysis did not fall within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

v) Winsopia's analysis was not permitted by Article 6 of the Software Directive.

Copybooks (Paragraphs 2.1.1.3 and 32.10-32.12 of the Technical Particulars) – nature of the dispute

641. The allegation is that Winsopia reverse engineered the copybooks and supplied the contents of the same to LzLabs. The defendants' case is that, as set out above, copybooks do not form part of any ICA Program, the information sent by Winsopia to LzLabs was not source code but rather information derived from observing, studying and testing the copybooks, and Winsopia's activities were within its interoperability rights.

642. The pleaded examples relied on by IBM relate to the COBOL copybooks in SDFHCOB and are considered below. Again, there is no dispute as to the material facts.

Item 28: DR-715 (Paragraph 32.10 of the Technical Particulars)

643. CICS includes a feature called basic mapping support used by user application programs to present screens to an IBM 3270 terminal user and obtain information provided to that screen by a user. User programs, when using CICS APIs to present a 3270 screen, can provide attributes for fields within that screen, specified as numbers. CICS provides a resource that can be used by user

programs written in languages such as Assembler, COBOL, C and PL/I, source code for each programming language that assigns a variable to each number.

644. As helpfully explained by Mr Stephens, using variable names rather than a number can make it easier to program; programmers do not need to research the number for each variable (such as a colour). Variables also make it easier for other programmers to read and to understand the program code.

645. The CICS copybook, DFHMBSA, is IBM supplied source code that defines the standard field attributes and printer control characters used by 3270 terminals (or terminal emulators).

646. CICS documentation provided by IBM lists the names of each of the variables and a description of the attribute to which it relates but does not specify the actual number assigned to each variable.

647. LzLabs created a COBOL program that displayed the value of each of the variables and attached it to DR 715 dated 23 July 2021, requesting Winsopia to provide the numeric values for the CICS DFHBMSCA copybook constants.

648. Winsopia compiled and executed the program and, on 26 July 2021, Mr Lynch sent the test results back to Mr Joot of LzLabs Canada. This did not include the compiler listings or scrubbed modules.

649. Initial implementation of the SDM version of DFHBMSCA was created by LzLabs on 19 March 2019. Following the response to DR 715, Mr Joot updated the SDM file, by replacing all placeholder values with the copybook constant values supplied by Winsopia. Against the Git commit dated 26 July 2021, Mr Joot identified DR 715 as the source of the constant values.

Item 29: DR-753 (Paragraph 32.11 of the Technical Particulars)

650. CICS includes features used by user application programs to present screens to, and obtain information from, a 3270 user. A 3270 terminal user has several attention keys, each one identified by a number, or 'AID'. CICS provides features for user programs to identify the attention key pressed and determine processing depending on the key used. As for the DFHMBSA copybook above, CICS provides a resource that can be used by user programs written in languages such as Assembler, COBOL, C and PL/I, source code for each programming language that assigns a variable to each number.

651. LzLabs created a COBOL program that attempted to dump all the DFHAID copybook fields and attached it to DR 753 dated 20 August 2021, requesting Winsopia to provide and/or verify the numeric values for the DFHAID copybook constants.

652. Winsopia compiled and executed the program and, on 23 August 2021, Mr Lynch sent the test results back to Mr Joot of LzLabs. This did not include the compiler listings or scrubbed modules.

653. Initial implementation of the SDM version of DFHAID was created by LzLabs in March 2019. Following the response to DR 753, Mr Joot updated the SDM file, by correcting and/or adding the copybook constant values supplied by Winsopia. Mr Joot identified DR 753 as the source of the constant values.

Item 30: DR-756 (Paragraph 2.1.1.3 of the Technical Particulars)

654. CICS creates a data area called the Exec Information Block ("EIB"), which is used for direct communication between command level programs and CICS. CICS provides APIs to allow user application programs running in CICS to access the data area. CICS also provides resources to allow user application programmes to determine the format of the EIB for different programming languages, including Assembler, PL/I and COBOL. IBM fully documents the format of the EIB data area in CICS documentation.

655. The IBM supplied DFHEIBLK and DFHEIBLC copybooks defines the layout of an EIB.

656. LzLabs created two programs that displayed the location of fields in the EIB (their offset from the beginning) and attached them to DR 756 dated 24 August 2021, requesting Winsopia to provide and/or verify the layout of the structure of the DFHEIBLK copybook.

657. Winsopia compiled and executed the programs and Mr Lynch sent the offsets for the DFHEIBLK copybook fields to Mr Joot at LzLabs on 24 August 2021. Information on field names was included but not compiler listings or scrubbed modules.

658. Versions of the COBOL copybooks mapping the EIB were created in the SDM prior to DR 756. They were updated using information obtained in DR 756 and used to support customer COBOL applications that rely on the copybooks. Although there is an apparent discrepancy in dates, in that the relevant Git commit is dated 23 August 2021, prior to the DR response on 24 August 2021, DR 756 is recorded by Mr Joot as the source of the Git commit.

Copybooks - discussion

659. In each case, it is common ground that LzLabs used the DR system to request Winsopia to document the values/constants defined in various IBM copybooks. Winsopia printed the requested values and sent the information to LzLabs. SDM code was created or developed using the values provided by Winsopia.

660. It is common ground that the same issues of principle arise in relation to the macros and copybook allegations. For the reasons set out above in relation to macros, I find that:

    i) The above copybooks were ICA Programs within the meaning of the ICA.

ii) Winsopia's activities amounted to reverse engineering in breach of clause 4.1.3(a) of the ICA and/or misuse of source code in breach of clause 4.1.1(d) of the ICA.

iii) Winsopia's analysis did not fall within permitted observation, study and testing pursuant to Article 5(3) of the Software Directive.

iv) Winsopia's analysis was not permitted by Article 6 of the Software Directive.

Transferring "unscrubbed" materials

661. The allegation is that Winsopia transferred to LzLabs or third parties load modules containing IBM CSECTs and other proprietary material in breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA. Although Winsopia purported to "scrub" the IBM CSECTs and other proprietary material, the methods adopted were flawed and incomplete, with the result that not all such material was removed from the load modules.

662. The defendants' case is that the IBM CSECTs and other proprietary materials were not ICA Programs and, once bound into a customer application during the compiler or link-editing process, became part of the customer application. Additionally, many of the IBM CSECTs the subject of these allegations did not relate to material licensed to Winsopia under the ICA but were licensed to third parties, licensed for distribution under Licensed Program Specifications or sample program licence terms, or historic IBM CSECTs relating to old versions of products not licensed to Winsopia.

663. The disputed issues are:

i) whether materials sent by Winsopia to LzLabs included ICA Programs within the meaning of the ICA;

ii) whether Winsopia's supply of such materials to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA;

iii) in respect of limited allegations, whether Winsopia's actions were necessary in order to achieve interoperability of customer applications with the SDM and, as such, were permitted by Article 6 of the Software Directive.

664. During the compilation and/or link-editing processes, IBM CSECTs, code fragments, routines and other IBM proprietary materials are inserted into the application, as explained earlier in this judgment. Clause 4.1.3(b) of the ICA prohibits Winsopia from transferring any ICA Program outside Winsopia's Enterprise. Further, clauses 4.1.1(b) and 4.1.2(b) restrict the use of any ICA Program to the permitted use identified in the ICA.

665. Winsopia transferred load modules and other materials to LzLabs for the purpose of developing the SDM and to assist third party customers in migrating their applications and data to the SDM. Prior to such transfer, it was necessary

for Winsopia to remove IBM CSECTs and other IBM proprietary materials from the load modules. There were two material purposes served by such removal: first, it would ensure that IBM proprietary material would not be transferred out of Winsopia's business enterprise in breach of the ICA; and second, it would enable the transferred load modules to be processed and run on the SDM, using instructions inserted by the SDM loader invoking alternative API calls to compatible runtime services.

666. Prior to 2015, a manual process of "stubbing" was used, whereby Winsopia replaced IBM CSECTs with empty "Winsopia stubs", dummy routines that would indicate to LzLabs where "wormholes" (instructions to call replacement runtime routines) should be inserted. This process did not involve any removal of IBM CSECTs. At the link-editing stage, Winsopia specified a NOCALL ("NCAL") option that prevented the binder or linkage editor from calling runtime libraries other than the Winsopia stub dataset. This process prevented the inclusion of IBM CSECTs at the link-editing stage but did not remove any CSECTs already inserted, for example, at compilation stage. Further, it required Winsopia to re-link-edit load modules supplied by customers, referred to by Simon Payne of Winsopia as "a real pain" in his email dated 27 October 2016.

667. Between 2015 and 2016, Winsopia developed a tool to migrate data, including load modules, initially known as the Data Migration Assistant ("the DMA"), later referred to as Centerpiece Export ("CPX"). CPX, which ran on the mainframe, used a LZMPDSE batch program to scan the load modules, deconstruct them into individual CSECTs, automatically remove the IBM CSECTs and replace them with placeholder binary zero code.

668. Mr Palmer of Winsopia wrote the CPX code and explained its various functions in his first witness statement:

i) First, it exports data, associated metadata and user-specified application logic from customers' mainframe databases and applications such as IMS, Db2 and CICS using the IBM-provided export functions for those products.

ii) Second, CPX generates a JavaScript Object Notation or JSON file which describes the data and contains metadata identifying the structure of the customer data extracted from the mainframe and schematic data such as data types. This information is then imported by the SDM along with the data itself.

iii) Third, CPX also prepares customer applications, which are exported from the mainframe as load modules. These are packaged up within CPX into a bundle that can then be taken across and imported into the SDM.

iv) Fourth, a further and very important function of CPX is scrubbing the load modules before they are exported from the mainframe. This involves removing control sections or CSECTs inserted by the IBM linkage-editor, as well as other IBM or third-party material which is identified. CPX scrubs load modules by replacing selected CSECTs with binary zeros. If the size of the CSECT being scrubbed is large enough

then a character string is written over that area that indicates the code section was removed.

v) Fifth, similarly to scrubbing, CPX excludes certain load library members from its exports, by referencing an exclusion list of member names stored in a text file called LZXLIST.

669. Mr Palmer's evidence was that LZSLIST, the load module scrub list, was created by him through the discovery request process, with input from developers at LzLabs, including Mr Bowler. Initially, it was based on the full names of CSECTs that were supported by the SDM, that is, where there were replacement wormholes.

670. In cross-examination, Mr Rastall stated that CPX identified the CSECTs to be redacted by reference to the scrub list. The scrub list was compiled based on Winsopia's understanding at any given point in time of the IBM CSECTs which needed to be redacted. It was updated manually as and when Winsopia became aware of new CSECTS to be added to the list but if the precise name of a CSECT was not on the scrub list, it would not be scrubbed by CPX.

> "Q. It was inherent in the process that there would be certain IBM CSECTs which were not contained on the manually updated scrub list?
>
> A. Yes, I think we've established that in previous points.
>
> Q. And if those CSECTs were contained in a load module being passed to LzLabs, it would follow, would it not, that those CSECTs would be transferred in unscrubbed form?
>
> A. Yes."

671. Mr Whittingham, a Senior Software Developer at Winsopia, took over Mr Palmer's role on CPX. He made changes to the scrub list that would prompt the batch program to look for specific code signatures and sequences within the modules rather than simply looking for names. The scrub list was expanded on an incremental basis as he was asked by various people at Winsopia and LzLabs to add items. In addition, he conducted checks to maintain the scrub list and requested LzLabs and Winsopia staff to notify him of any new CSECTs discovered that should be scrubbed.

672. In addition to the scrub list, Winsopia maintained LZXLIST, an exclusion list of module names, that would be excluded from migration to the SDM but not replaced by wormholes, and LZSRULE, a list of CSECTs that were not subjected to any scrubbing.

673. Mr Whittingham's evidence was that CPX contained algorithms that could have been used to facilitate wild cards, using search patterns to match generic names, such as "*" to capture all variations of CSECT names that required scrubbing. From the outset, it was suggested to Mr Palmer on a number of occasions that

wild cards should be used to identify and remove IBM CSECTs from the load modules but this suggestion was not adopted until after he left in 2020.

674. The allegations are that there were a number of scrubbing failures or exclusions, resulting in IBM CSECTs and other proprietary materials being sent by Winsopia to LzLabs.

Item 31:Epiphany

675. In its closing submissions, it is accepted by IBM that the relevant transfer of unscrubbed Language Environment CSECTs by Winsopia to a third party was permitted by the z/OS Licensed Program Specifications. Therefore, this allegation is no longer pursued.

Item 32: Db2 Catalog table metadata

676. Db2 is a relational database software product that provides a database management system in which customers can store information. It is common ground that Db2 versions 10, 11 and 12 were ICA Programs supplied by IBM to Winsopia for use with z/OS.

677. The Db2 catalog tables are a collection of database tables that are created as part of the Db2 installation process and keep track of objects and resources available within a Db2 system. The Db2 materials include: (i) data provided with the Db2 product or created during installation of Db2 which is identified by the IBMREQD field; (ii) stored procedures in the form of Structured Query Language ("SQL") statements, which allow programmers to write scripts and internal procedures that are stored in the Db2 catalog; (iii) default parameters and values in the table definitions; and (iv) IBM supplied packages and plans stored as records in the Db2 catalog tables. Mr Bray agreed in cross-examination that a Db2 catalog will always contain some material supplied by IBM.

678. CPX software copies information from the Db2 catalog when migrating Db2 data to the SDM. The evidence of Mr Palmer and Mr Bray was that Winsopia developed a set of filters so that, when migrating customer metadata and records stored in a Db2 relational database, CPX would exclude any information in the Db2 tables that might relate to IBM or other third-party products.

679. The experts agreed in their second joint statement that, at least from April 2016, CPX software included filters that were designed to exclude IBM provided objects, such as objects with the IBMREQD field set, when copying information from the Db2 catalog.

680. IBM's case is that contemporaneous email exchanges show that, prior to April 2016, CPX filters were ineffective in excluding all ICA Programs and other IBM proprietary material when exporting data to the SDM.

681. On 27 April 2016 Mr Palmer of Winsopia sent an email to Mr Bray and Mr Rastall at Winsopia, and to Mr Garfield and Mr Wehrli at LzLabs, setting out discussion items for a DMA (CPX) meeting.

682. In response, Mr Bray sent an email to Mr Palmer, stating:

> "To add to your list, there is a changed set of Winsopia DB2 Catalog Table Filters! Previously we were not excluding some IBM tables/programs."

683. Mr Palmer updated the items for discussion at the meeting, including the following:

> "… DB2 unload JCL catalog table filter syntax changes are required due to some IBM tables not being excluded from the migration payload. These changes will have to be identified and incorporated into the DMA."

684. Mr Bray confirmed in cross-examination that before he identified the above issue and created the updated list of filters, the Db2 tables were not filtered and therefore they would be transferred from Winsopia to LzLabs whenever Db2 catalogs were exported.

685. The Db2 catalog tables contained data that was part of the Db2 supplied program and/or generated by the Db2 program on installation. Such categories of data were component parts of Db2 and constituted an ICA Program for the purpose of the ICA. Further, data stored in the Db2 catalog tables included IBM proprietary material.

686. It is clear that, when the CPX filter was first introduced, it was ineffective in excluding IBM data in the Db2 catalog tables from the materials intended to be exported to the SDM. However, as the defendants submit, there is no evidence that any such materials were actually transferred outside Winsopia. Mr Swanson confirmed in cross-examination that the experts searched for, but were unable to find, any evidence of any transfer of Db2 catalog resources from Winsopia prior to April 2016.

687. IBM relies on an email dated 30 March 2016, sent by Mr Palmer to Mr Rastall and Mr Bray at Winsopia and to LzLabs, referring to the migration of a package, including Db2 catalog data. However, the email expressly refers to such data being subject to legal approval to exclude all IBM proprietary rows. This suggests that, at least in this particular case, a manual review was used to remove IBM proprietary data, rather than any reliance on the CPX filter. It does not indicate that IBM Db2 material was sent by Winsopia to LzLabs.

688. It follows that IBM have not established any breach of the ICA in respect of this item.

Item 33: DSS dump

689. The allegation is that Winsopia sent an unscrubbed DSS dump, containing code fragments forming part of an ICA Program, to LzLabs.

690. By email dated 16 May 2016, Mr Rastall raised a query regarding the export of IBM stubs by Winsopia:

> "I have been asked to understand and collate some information on DMA output files (payloads) that have been extracted by Winsopia staff and transmitted that contain modules with unchanged IBM stubs. I am not interested in Payloads that contain Winsopia stubs only.
>
> I am making the assumption that all such DMA payloads transmitted have been sent to Christian only, please correct me if I am wrong.
>
> I am also making the assumption, that with most (if not all) of such payloads the destination has been either the Winsopia PizzaBox or a Winsopia NUC. (but via Christian).
>
> I need to know if any DMA Payloads containing IBM stubs has been loaded on other systems outside Winsopia."

691. Mr Wehrli of LzLabs responded on the same day as follows:

> "What do you mean with other? I always check that only stubbed libraries are included. The[y] libraries also always have the same "stubbed", on it. If there has been an incident where I slipped something through which should not be in there I need to know.
>
> Every request that ends up in a DMA dump has a PRB corresponding to it, i also always request stubbs and leave a note that only stubbs to be included.
>
> During the [customer] exercise one of the DSS dumps slipped the normal library in by accident. I have informed Dev's to not use this and deleted it from SVN, and replaced it with a stubbed one, and also for this we only use the stubbed one."

692. The experts were unable to locate the material DSS dump because, as stated by Mr Wehrli above, it was deleted on 16 May 2016. As a result, they were not able to determine what the unscrubbed load modules comprised.

693. There is insufficient evidence on this item for the court to determine whether the material sent by Winsopia to LzLabs contained an ICA Program, or part thereof. Even if it did, on the very limited evidence available, it is likely that any slip would have been inadvertent and temporary. There is no evidence that the unscrubbed load modules were used by LzLabs. It follows that IBM have not established any breach of the ICA in respect of this item.

Item 34: Kednos

694. The allegation is that Winsopia failed to scrub properly a suite of PL/I test programs and, as a result, IBM CSECTs were transferred outside Winsopia's Enterprise and used on non-designated machines.

695. Kednos test programs are a PL/I compiler-verification test suite. In early 2015 Winsopia was instructed to convert a suite of Kednos PL/I test programs in

source code form into executable programs capable of running on z/OS that could be exported to the SDM to verify that the SDM implementation of PL/I produced the same results as an IBM mainframe. This required Winsopia to modify the source code for the Kednos test programs, so that they could be compiled using the IBM PL/I compiler, and then link-edit them with IBM-supplied PL/I modules on its mainframe.

696. On 19 December 2016, Mr Bleach of LzLabs UK sent an email to Mr Rastall at Winsopia and others, concerning PL/I Kednos tests sent to OnTarget in St Petersburg. On Target had been sent two versions of the Kednos library, namely, the old stubbed library and the new CPX scrubbed library. Mr Bleach was unable to explain why the old stubbed tests run by On Target worked but the new CPX scrubbed tests did not work.

697. On 20 December 2016, during an instant messaging exchange between Mr Bleach and Mr Bray, Mr Bray stated:

> "BTW I do not think that the Scrubbing process for PL1 is complete! In fact I think it is letting test through with Copyright IBM in it!! … I have told Mr Palmer.
>
> …
>
> Chris has come up with another enhanced Scrubb list --- LZM5.WINCP.JCL(LZSLIST) so maybe the CPX Needs doing again??"

698. On 20 December 2016 Mr Bray sent an email to Mr Palmer, explaining that he had compared the PL/I z/OS executable load modules using the CPX scrubbing process with executables linked with Winsopia stubs; the CPX version contained a number of IBM copyright statements that did not appear in the stubbed version. In response, Mr Palmer provided an updated and enhanced LZSLIST (scrub list). That proved to be more effective although, as noted by Mr Bray, still it did not scrub all modules containing IBM copyright notices.

699. In cross-examination Mr Bray agreed that, at this time, the CPX scrubbing process was ineffective in that IBM CSECTs in the PL/I load modules were sent by Winsopia to OnTarget:

> "Q. … there may be a question of degree, but I think you're accepting, aren't you –
>
> A. Yes.
>
> Q. -- that the scrubbing process had failed to the extent that at least some IBM –
>
> A. Yes, yes.
>
> Q. -- CSECTs had gone from Winsopia to OTG, possibly via Lz; do you agree?

A. Yes."

700. Mr Palmer's evidence was to like effect:

> "Q. Now, if we just take those two emails we've been looking at together, what we see is that OnTarget Group has been sent a scrubbed copy of the KEDNOS library, that it wasn't scrubbed successfully because every module contained copyright statements, but the version that was done the old way was fine, and what the copyright statements that were sent to OnTarget in the scrubbed version showed was that IBM CSECTs hadn't been scrubbed; do you agree?
>
> A. Yes, from the evidence, that's true."

701. The link-editing process carried out by Winsopia necessarily introduced into the PL/I modules IBM CSECTs, including z/OS Base version 1 or 2 and/or IBM Enterprise PL/I for z/OS. Those IBM CSECTs were ICA Programs for the purpose of the ICA.

702. In summary on this item:

i) materials sent by Winsopia to LzLabs and/or OnTarget included ICA Programs within the meaning of the ICA;

ii) Winsopia's supply of such materials to LzLabs and/or OnTarget constituted breach of clauses 4.1.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

Item 35: CSECTs deliberately omitted from scrubbing

703. The allegation is that Winsopia deliberately omitted from the scrubbing process certain CSECTs when load modules were exported to LzLabs for use in development of the SDM. There is a large measure of agreement between the experts as to the relevant technical facts as set out in their second expert joint statement.

704. CEESTART is an initialisation CSECT comprising both data and code. For COBOL programs, CEESTART is taken from the SCEELKED library distributed with z/OS Base and link-edited into load modules by the z/OS Binder. In PL/I programs, CEESTART is inserted by the IBM Enterprise PL/I compiler. In either case, CEESTART is populated at bind time with values relative to the load module in which it is contained.

705. Mr Palmer confirmed in cross-examination that CEESTART was originally removed from COBOL programs and replaced by Winsopia stubs. However it was not removed from PL/I programs because the Winsopia stubs only replaced link-edited CSECTs; not those inserted at compilation stage. Therefore, initially, CEESTART was not removed from PL/I load modules sent from Winsopia to LzLabs. The introduction of the CPX scrubber enabled the removal of CEESTART from both PL/I and COBOL programs but, from 22 February 2017, CEESTART was taken off the scrub list. Thereafter, as Mr Palmer and

Mr Stephens confirmed, any PL/I or COBOL load modules sent to LzLabs would contain CEESTART.

706. IGZUOPT is a data only CSECT that can be used by a mainframe user to specify COBOL options. IGZUOPT is generated by a mainframe user by creating an assembler program that calls the assembler macro IGZXOPT provided by IBM in the SCEEMAC library supplied with z/OS. The mainframe user specifies required parameter values for the IGZXOPT macro in the assembler code. The experts agree that IGZUOPT is not scrubbed by CPX; it is likely that it has been sent by Winsopia to LzLabs.

707. CEEROPT is a data only CSECT that can be used by a mainframe user to specify z/OS Language Environment options. CEEROPT is generated by a mainframe user by creating an assembler program that calls the assembler macro CEEXOPT provided by IBM in the SCEEMAC library supplied with z/OS. The mainframe user specifies required parameter values for the CEEXOPT macro in the assembler code. The experts agree that CEEROPT is not scrubbed by CPX; it is likely that it has been sent by Winsopia to LzLabs.

708. CEEMAIN and CEEFMAIN are data only CSECTs bound with COBOL and PL/I programs. They are obtained from the SCEELKED library provided with z/OS for COBOL programs and bound or inserted by the compiler for PL/I programs. The experts agree that they are not scrubbed by CPX; it is likely that they have been sent by Winsopia to LzLabs.

709. DFS* modules are data only CSECTs relating to IMS. The format of the modules and default values populating the data structures are built by IBM-provided macros. Some of the data in these CSECTs is dependent on parameters provided by the IMS administrator during generation. These modules are placed in a dataset called 'RESLIB' or 'SDFSRESL' library. The RESLIB / SDFSRESL library where these modules are placed may also contain modules supplied by IBM with the IMS product. The experts agree that these modules are not scrubbed by CPX; it is likely that they have been sent by Winsopia to LzLabs.

710. The defendants submit that none of the CSECTs the subject of this allegation is an ICA Program. The CSECTs are custom-built by the customer, based on: (i) the individual configuration and/or parameters that the customer has selected based on their business requirements, using IBM-supplied macros and sample programs to generate the relevant CSECTs; and/or (ii) statements made by the programmer in the source code of their customer application, or metadata relating to the entry point specific to their executable customer application. Further, they submit that the CSECTs are subject to different licensing terms, which permit distribution, with or without modifications for the purpose of developing, using, marketing or distributing application programs.

711. I reject that submission. The CSECTs are selected by the customer based on their business requirements but implementation of the customer's stated requirements in an application is determined by IBM, through its design of the CSECTs, its choice as to when and how code is inserted and/or generated, and its decision as to the mode of execution. The Licensed Program Specification

and other sample program terms and conditions provided that distribution of such programs would be subject to IBM copyright notices and for the purposes of developing, using, marketing or distributing application programs "conforming to the application programming interface for the operating platform for which the sample programs are written" that is, for use with z/OS. Therefore, such permission did not extend to distribution of load modules containing such CSECTs for the purpose of developing an alternative operating system such as the SDM.

712. CEESTART is a component of z/OS Base and IBM Enterprise PL/I for z/OS, both of which are agreed to be ICA Programs. As such, for the reasons set out earlier in this Judgment, it is an ICA Program. Mr Swanson agreed in cross-examination that the functional content of this CSECT, whether inserted at the compilation or link-editing stage, is materially the same.

713. Likewise, CEEMAIN and CEEFMAIN are supplied by IBM as part of the z/OS Language Environment SCEELKED dataset. As such, for the reasons set out earlier in this Judgment, they are ICA Programs.

714. IGZUOPT and CEEROPT are data-only CSECTs which can be used by a mainframe user to override default runtime operations in z/OS. Mr Stephens explained in his evidence that IBM supplies, as part of the z/OS Language Environment, a job control language ("JCL") script, assembler source code to invoke a macro, and the macro itself. The user adds its details to the JCL script and amends the assembler source code to select the required default option from the range of options set out in JCL samples compiler dataset (.SCEESAMP) or the z/OS Language Environment Programming Guide. Running the JCL causes the Assembler to assemble the source code, invoke the macro and generate the CSECT. The macros were designed by IBM and provided in source code as part of z/OS Base. They are machine-readable instructions and data; as such, they are ICA Programs within the ICA definition.

715. DFS* CSECTs are data only modules generated as part of the IMS installation process. Mr Swanson and Mr Stephens concurred that during IMS system generation, IBM-supplied macros can be customised by users and assembled by running the IBM-supplied JCL to generate the load modules. Alternatively, the user can simply select unmodified load modules supplied by IBM. The load modules contain code, data and structures designed by IBM. Although the user can specify parameters and values that are unique to its business requirements, such configuration does not change the fact that the CSECTs are essential components of IMS. IMS versions 12, 14 and 15 are ICA Programs. As such, the CSECTs are ICA Programs within the ICA definition.

716. For the reasons set out earlier in this Judgment, the above findings are not affected by the fact that the CSECTs are incorporated into the customer applications or generated by software that itself is an ICA Program. The CSECTs remain subject to the terms of the ICA.

717. The defendants submit that the CSECTs were deliberately omitted from the scrubbing exercise because they were necessary to enable customer applications to run on the SDM. The expert evidence did not support that argument.

718. The experts agreed in their second joint statement that the SDM does not need CEESTART for COBOL programs. Mr Jaeger stated that LzLabs worked out an alternative way to obtain the relevant information so that CEESTART became unnecessary for PL/I programs.

719. Mr Swanson's opinion was that, although the CSECT information might be needed to execute a load module on the SDM, Winsopia could have analysed the load module prior to scrubbing to identify the relevant information required and scrubbed the load module before transfer to LzLabs. It would have been open to LzLabs to obtain the information from published documentation and develop replacement functionality on the SDM. Mr Stephens agreed that what was necessary was not the CSECTs themselves but the information that was contained within them.

720. In summary on this item:

i) Load modules sent by Winsopia to LzLabs included CSECTs that were ICA Programs within the meaning of the ICA.

ii) Winsopia's supply of such modules to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

iii) Winsopia's actions were not necessary in order to achieve interoperability of customer applications with the SDM and, as such, were not permitted by Article 6 of the Software Directive.

Items 36 and 42: Unscrubbed CSECTs

721. In July 2017 LzLabs opened DR-2581, requesting Winsopia to process customer applications on its mainframe. As recorded on the DR, initially, Winsopia encountered difficulties because the data supplied was incomplete but subsequently, it used CPX to scrub and package the load modules and sent them to LzLabs.

722. On 26 July 2017 LzLabs opened DR-2625, identifying an IBM CSECT, IGZCBSN, that had not been scrubbed from the load modules and asking Winsopia to rescrub those libraries to remove it.

723. Mr Palmer and Mr Bowler discovered that the CSECT was not specified on the CPX scrub list, as recorded in Mr Palmer's email dated 2 August 2017. By further email dated 14 August 2017, Mr Bowler stated:

> "Chris Palmer sent me the attached "scrub list" of csects which Winsopia remove from load modules and replace by dummy csects. Chris has asked Lzlabs to review the list and report back additions or deletions.
>
> To my mind the list appears at first sight somewhat sparse. I would prefer to see a blanket exclusion of everything beginning with IGZ, CEE, etc. with specific exceptions like CEESTART, CEEUOPT. This would require a modification to the Winsopia

> scrub utility but Chris indicated he is willing to make that enhancement."

724. On 4 September 2017 DR-2747 was opened, in which Mr Bowler stated that the load library delivered in response to DR-2625 to LzLabs was found to contain 43 listed CSECTs, which were then added to the CPX scrub list. Yet further unscrubbed CSECTs were discovered in this load library as recorded by Mr Palmer in Bugzilla Ticket 347-A on 5 September 2017.

725. The defendants' submission is that none of the relevant materials originated from Winsopia's mainframe; they came from versions of IBM products that were not licensed to Winsopia, and were not provided by IBM to Winsopia. That is an oversimplification of the process; Winsopia did not just act as a post-box for customer applications. The CPX process involved loading the application and calling the CPX LZMPDSE program, which in turn would call the z/OS binder to read the load module, break it into CSECTs and perform the scrubbing exercise. The question in respect of this allegation is the extent to which, if at all, the relevant load modules were in fact subject to CPX processing.

726. The experts agree that a number of the unscrubbed CSECTS related to historic versions of COBOL and PL/I that could not have emanated from Winsopia's mainframe. However, others are still supplied by IBM with z/OS and are available to Winsopia on its mainframe. Both Mr Swanson and Mr Stephens examined the load modules on the zPDT machine. They identified that a number of the unscrubbed CSECTs were not contained in the z/OS library and therefore unlikely to have been available on Winsopia's mainframe but they also identified a number of the unscrubbed CSECTs that are available in z/OS and therefore available on Winsopia's mainframe. There was not complete agreement between the experts. Mr Swanson and Mr Stephens agreed that 19 of the unscrubbed CSECTs are part of the SCEELKED dataset forming part of z/OS Base. They disagreed on a further 4 CSECTs, with Mr Stephens identifying them as CSECTs present in a different version to those available to Winsopia.

727. DR-2625 suggests that Winsopia might have re-compiled, assembled and/or link-edited some of the applications on the mainframe, thus indicating that at least some of these unscrubbed CSECTs could have been inserted into the load modules at that time. Mr Stephens agreed in cross-examination that it was plausible that Winsopia re-bound the modules but he stated that he had seen no evidence to that effect.

728. Drawing together the strands of evidence on this issue, despite the best efforts of the experts, it is not clear what was done by Winsopia when sending these load modules to LzLabs. The contemporaneous documents demonstrate that CPX did not remove CSECTs effectively from these load modules. However, there is inadequate evidence as to whether the versions of the CSECTs which Winsopia could access under the ICA contained the same code as the older versions in the load modules supplied by the customer. Therefore, it is unclear whether Winsopia transferred out of its enterprise any CSECTs that were subject to the terms and conditions of the ICA.

729. It follows that IBM have not established any breach of the ICA in respect of these items.

Items 37 and 40: IMS PROCLIB & DLIBATCH

730. These allegations concern the transfer by Winsopia of JCL procedures to LzLabs. IMS procedures ("PROCs") are JCL files that are used to configure elements of the user's IMS installation, including the way in which IMS tasks will run. "Sample" procedures are provided with the IMS software product for this purpose in the SDFSPROC library and include an IBM copyright statement.

731. DR-4184 was opened on 8 July 2019 and contained a request by LzLabs for Winsopia to modify a customer program as an IMS application, package it up using CPX and send it to LzLabs. In the answer on 9 August 2019, Winsopia explained that the customer application was unsuitable for this purpose; instead, Winsopia had built its own sample, supplied via DR-4244. This indicates that the SDFSPROC library sent by Winsopia originated on its mainframe.

732. DR-4244 was opened on 31 July 2019 and contained a request by LzLabs for Winsopia to send H2R and DL2 versions of a test program developed by LzLabs. A comment was added on 22 August 2019 stating:

> "As I processed DR4244 I noticed it has explicit JCL members with the IBM explicit copyright notification in one member I have seen and I did not want to open anything further. This member is the DLIBATCH member in the IMS version of the CPX package…
>
> Since that was the only member I looked at, I did not view any other. Turned out our exercise did not need that proc specified in the original JCL.
>
> I am deleting these resources and requesting another CPX package that hasn't any copyrighted material."

733. By email dated 3 October 2019, Conley Shepherd at LzLabs UK notified Mr Maddison at Winsopia that he had identified IBM copyrighted material in the DR-4184 CPX, in particular, an IMS procedure. In response, Mr Maddison stated:

> "SDFSPROC is being automatically selected by CPX as datatype (HDB PROCLIB) as requested by the IMS/HDB Specifications for CPX-CPI. Should we exclude all members that contain Copyright text? There also members in HDB RESLIB that contain Copyright statements, though these being load modules we should be scrubbing them already.
>
> As regards issues with the JCL I will test this on my SDM and build a new package when your next DR requesting a replacement for DR4184 arrives. I will run my new Copyright scanning job against it before sending it out. This does however

> raise the question as to whether CPX itself should perform this check for non-load module material?"

734. In further emails sent to Mr Rastall on this issue, Mr Maddison stated:

> "We clearly have a serious issue sending you the IMS PROCLIB dataset because it contains all manner of stuff, much of it either irrelevant to the IMS region of interest and/or proprietary to IBM.
>
> …
>
> The way that CPX automatically discovers artefacts to export is complex and we need to investigate where we might be exposed. Of the 4 IMS packages I built, these 2 contained sample PROCLIB members marked "Copyright IBM" and have been deleted from the FTP site.
>
> …
>
> I don't know about any other IMS packages that may have been created in the past. Looking forward, when exporting IBM supplied source libraries required to support an application, we need to either apply a filter to identify copyright or extract only specific user customisable members. This likely applies to CICS and DB2 also. In the meantime I have figured out a workaround that will cause CPX to discover a redacted IMS PROCLIB and will rebuild DR4380 and DR4184."

735. Mr Whittingham responded as follows:

> "There definitely appears to be a need for some form of detection of copyrighted material being included into CPX packages, particularly when the offending datasets are automatically selected."

736. In cross-examination, Mr Jaeger accepted that part of the IMS dataset must have been created on the Winsopia mainframe because the file contained the presence of a Winsopia qualifier. Mr Stephens agreed that DR-4244 related to an IMS test application created in-house by Winsopia and sent to LzLabs, including the IMS procedure library containing DLIBATCH. Mr Swanson and Mr Stephens both examined datasets with names that matched those referred to in the above emails, from which it was apparent that Winsopia had deleted the IBM copyright statements and comments but made no other substantial changes.

737. From the above evidence, I find that during this period, Winsopia created test applications, using IMS procedures and datasets, which were sent to LzLabs through the DR process. The CPX tool was not designed to scrub JCL scripts and therefore IBM copyright material in the procedures and datasets was not removed.

738. SDFSPROC is an IMS PROCLIB dataset supplied by IBM as part of the IMS version 15 software product, an ICA Program. Although referred to as "samples" in IBM documentation, the procedures are clearly marked as IBM licensed materials. Their status as components of an ICA Program is not changed by the fact that the procedures can be customised. For the reasons set out in paragraph [711] above, the terms on which sample programs may be distributed do not permit export to LzLabs for the purpose of developing the SDM. They are designed to be run on a mainframe operating system and remain subject to the terms of the ICA.

739. In summary on this item:

i) IMS procedures and datasets sent by Winsopia to LzLabs were ICA Programs within the meaning of the ICA.

ii) Winsopia's supply of such materials to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

Item 38: DFHEI1 module

740. The allegation is that Winsopia failed to scrub properly IBM module DFHEI1 and, as a result, such modules were transferred outside Winsopia's Enterprise and used on non-designated machines.

741. CICS programs are bound with a CICS stub which has an entry point called DFHEI1. IBM supplies the DFHEI1 CSECT with CICS as part of the SDFHLOAD library; another version of the DFHEI1 CSECT is supplied with IMS within the DFSLI000 load module.

742. A third-party software product, DL/2, allows programs designed to use IMS databases to use Db2 databases without application charges. Winsopia used DL/2 processing by converting customer IMS load modules to a DL/2 compatible format when migrating them to the SDM.

743. The experts agreed in their second joint statement that, for applications using the EXEC DLI API, the DL/2 process provided a replacement 'DFHEI1 stub' but did not remove the original 'DFHEI1 stub'; it simply renamed it 'DFHEI1X'. Prior to November 2018 CPX was not programmed to scrub 'DFHEI1X.' As a result, load modules transferred by Winsopia to LzLabs contained both the replacement DFHEI1 stub and the original CICS or IMS DFHEI1X stub.

744. This issue was highlighted by Mr Spencer of LzLabs in an email dated 24 November 2018:

> "It looks like DFHEI1X is the real IBM DFHEI1 that has been renamed to X by the DL2 conversion process. DFHEI1 is the CIRCLE intercept module for that entry point ...
>
> What we need is for … DFHEI1 to be renamed … and NOT redacted (it is circle code), it is not being renamed in the CPX

> we receive. DFHEI1X should not be renamed and SHOULD be redacted since it contains the IBM DFHEI1 code now.
>
> We can add a "wormhole" on our side for DFHEI1X to treat it just like we would DFHEI1."

745. In response, Mr Palmer stated that he had updated the CPX rules file as requested, although in a further email dated 29 November 2018 Mr Spencer noted that the CPX renaming problem had not yet been resolved.

746. There is evidence that the renaming problem occurred in respect of load modules transferred by Winsopia to LzLabs in 2018, as set out in DR-3435, DR-3552 and DR-3107. There was difficulty in analysing some of the DR material as a result of corrupted files. However, the experts agreed in their second joint statement that DR-3107 showed evidence that the DFHEI1X module was sent to LzLabs. In cross-examination, Mr Stephens agreed that it showed that DL/2 applications contained CICS renamed stubs. The DR-3552 evidence was inconclusive but it is likely that the IMS DFHEI1X module was sent to LzLabs given that the relevant program was the subject of Mr Spencer's November email.

747. On the basis of the above evidence, I find that on the balance of probabilities Winsopia transferred load modules to LzLabs containing the renamed DFHEI1X module. Regardless of whether the programs originated as customer applications, they were processed by Winsopia on its mainframe, using DL/2. As set out earlier in this judgment, the acts of loading, displaying, running and storage of applications containing IBM CSECTs involve reproduction of the IBM software and are subject to authorisation under the terms of the ICA.

748. It is common ground that IMS Versions 12, 14 and 15 are ICA Programs. The IMS version of DFHEI1 is a CSECT in the DFSLI000 load module provided with IMS. It follows that the IMS DFHEI1 CSECT is a component of an ICA Program, and therefore an ICA Program. For the reasons set out in paragraphs [185] to [193] above, the terms of the Licensed Program Specification for IMS do not permit export to LzLabs for the purpose of developing the SDM.

749. It is also common ground that CICS Transaction Server for z/OS Version 5 is an ICA Program. The CICS version of DFHEI1 is provided by IBM in the SDFHLOAD library supplied with CICS. It follows that the CICS DFHEI1 CSECT is a component of an ICA Program, and therefore an ICA Program.

750. In summary on this item:

    i) IMS load modules sent by Winsopia to LzLabs contained DFHEI1 CSECTs that were ICA Programs within the meaning of the ICA.

    ii) Winsopia's supply of such materials to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

Item 39: IGZXANE

751. IGZXANE is a 24-byte CSECT that is bound into a user load module as a stub routine and provided by IBM in the SCEELKED library with z/OS. As such, it is an ICA Program for the purpose of the ICA.

752. On 11 January 2019 DR3802 was opened, requesting Winsopia to compile all NIST Programs in the IC suite using COBOL version 6 with compiler listings. Winsopia duly sent the requested load library and compiler listings. A comment added on 15 February 2019 noted that a couple of the modules contained a CSECT called IGZXANE which should have been redacted but was not.

753. It is common ground that a CPX bug allowed IGZXANE to remain in a scrubbed package and the load module was sent by Winsopia to LzLabs. This bug was identified and subsequently CPX was modified to fix it. The experts agreed in their second joint statement that there is no evidence that IGZXANE was used by LzLabs in the development of the SDM.

754. It follows that the transfer of this ICA Program out of Winsopia to LzLabs amounted to a breach of the ICA but the breach was inadvertent and the problem was swiftly resolved. On that basis, it was inconsequential.

Item 41: IGZXNE3N

755. IGZXNE3N is an IBM-supplied CSECT that is bound into a user load module as a stub routine and provided by IBM in the SCEELKED library with z/OS. As such, it is an ICA Program for the purpose of the ICA.

756. On 8 November 2019 DR-4472 was opened, requesting Winsopia to compile NIST NC tests with the COBOL 6.2 compiler. Winsopia duly sent the requested load modules and compiler listings. On 9 March 2020 DR-4752 was opened, noting that one of the modules contained a CSECT called IGZXNE3N that should be redacted during the CPX export process and therefore should be added to the scrub list. Subsequently, the CSECT was added to the scrub list.

757. It is common ground that the unscrubbed load module was sent by Winsopia to LzLabs. The experts agreed in their second joint statement that the SDM created its version of the IGZXNE3N runtime on 11 March 2020, two days after the DR date. Professor Donaldson carried out a review of the relevant SDM source code file. He concluded that the SDM version did not contain any relevant IBM material and appeared to have been independently produced by LzLabs. He accepted that DR-4472 was useful in determining the format of parameters to be provided as input to the function but he found no evidence that the DR was of any further use. However, in cross-examination, Professor Donaldson accepted that, in addition to the compiler listings supplied through DR-4472, by decompiling the unscrubbed module or stub, one could glean information about the parameters.

758. The proximity of time between the identification of the unscrubbed CSECTon 9 March 2020 and the SDM implementation of the IGZXNE3N runtime routine on 11 March 2020 indicates that it is likely that it did inform LzLabs in its development of an equivalent to the IBM runtime on the SDM.

759. It follows that the transfer of this ICA Program out of Winsopia to LzLabs amounted to a breach of the ICA.

*Item 43: CEEBETBL, CEEBLLST, IBMPINPL & CEESG**

760. This allegation concerns the removal of Language Environment CSECT data areas from the scrub list.

761. CEEBETBL specifies the addresses of other CSECTs, including CEEBLLST and the termination stub routine address. CEEBLLST declares the language list for the load module or program object. The CEESG* CSESCTs specify the addresses of CSECTs specific to each language. They are provided by IBM in the SCEELKED library with z/OS. As such, they are ICA Programs for the purpose of the ICA.

762. On 22 June 2017 Tim Sneddon of LzLabs sent an email to Mr Jaeger, asking whether certain of the above CSECTs could be included in the load modules sent from Winsopia:

> "According to the manual they aren't executable code, but parameters lists and item lists that give details about how a load module should be handled and which language run-times should be initialised."

763. On the same day, Mr Bowler of LzLabs identified a number of additional CSECTs that should not be redacted because they could provide useful information.

764. On 12 September 2017 as recorded in the Bugzilla Ticket 354-A, Winsopia removed these CSECTs from the scrub list.

765. On 15 September 2017, following receipt of Mr Bowler's preliminary analysis of program objects produced by the IBM C and PL/I compilers, Mr Bond sent the following email to Mr Bowler, Mr Sneddon and Mr Jaeger:

> "Roger's analysis was very informative, as was Tim's pointer to the relevant documentation in the Language Environment Vendor Interface manual.
>
> My conclusion is that we will need most of the contents of the CEEBETBL, CEEBLLST and CEESGnnn sections. I don't know if we need the contents of xxxINPL - I defer to Roger and Tim for those. Each of the sections that I listed contains pointers to things that we will need to initialize the environment…
>
> There really isn't any other way to get this information except to have these sections. This is because information external to the loaded program, such as information about names, is not always available…"

766. After receiving these CSECTs from Winsopia in unscrubbed form, LzLabs' developers used the Language Environment Signature CSECTs CEESG003 and

CEESG011 to create data structures in the SDM source code that were equivalent to those contained within the original unscrubbed IBM CSECTs.

767. The creation of those data structures, together with modifications to the SDM loader, enabled the SDM to reconstruct these CSECTs in load modules from which they had been scrubbed, with the result that Winsopia could then return them to the scrub list.

768. The experts agreed in the second joint statement that some, but not all, of the format of the relevant CSECTs is available in public documentation. Mr Bond's initial evidence was that LzLabs developed the SDM loader solution using information from public sources but, having been shown contemporaneous document in cross-examination, he accepted that Winsopia supplied LzLabs with load modules containing unscrubbed versions of the relevant CSECTs, and that Mr Bowler used information in those CSECTs to determine the length and format of the data structures.

769. On 25 September 2017 Mr Bowler made a Git commit, recording the addition of structure definitions to the SDM implementation of the relevant CSECT data structures.

770. When shown a copy of the Git commit, Mr Bond agreed in cross-examination the following:

> "Q. So, on the face of it, what Roger Bowler has done has investigated the contents of the CSECTs, the data structures in the CSECTs, using a combination of IBM documentation and, where that wasn't complete, actual CSECTs produced unscrubbed by Winsopia?
>
> A. It does appear to me that way.
>
> Q. Then he copies the structure, doesn't he, into the SDM?
>
> A. He copies the definition of the structure into the SDM."

771. In summary on this item:

i) Load modules sent by Winsopia to LzLabs contained data only CSECTs that were ICA Programs within the meaning of the ICA.

ii) Winsopia's supply of such materials to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

Item 44: DR-4617

772. There is no dispute as to the underlying facts giving rise to this allegation. A potential customer of LzLabs used CPX with a specific scrub list to package up load modules and sent them to Winsopia. DR-4617 was opened on 21 January 2021, by which LzLabs requested Winsopia to carry out checks on the load modules to ensure that the redaction process had been performed correctly. When Winsopia analysed the files, it detected unscrubbed IBM CSECTs and

endeavoured to remove them using CPX. Mr Whittingham explained in his evidence that when initially scrubbing for IBM material, CPX detected the "LZPDSE" component and assumed (wrongly) that it did not need scrubbing. This was discovered by Winsopia, the file was renamed and CPX effectively removed the IBM CSECTs. Further third-party confidential information was identified, the customer was required to modify its CPX processing steps and re-send the files to Winsopia.

773. The files containing unscrubbed materials were initially uploaded by Winsopia on the Winsopia LzLabs FTP Server, used to make CPX packages available to LzLabs. They were deleted when the issues were discovered. Subsequently, the fully scrubbed files were sent by Winsopia to LzLabs.

774. The issue is whether there is evidence that unscrubbed or insufficiently scrubbed load modules were sent by Winsopia to LzLabs. Mr Stephens' evidence was that he had seen no evidence that any unscrubbed IBM modules, stubs or objects were sent to LzLabs. Although unscrubbed modules were placed on the FTP server, they were subsequently deleted and there is no evidence that the materials were accessed by LzLabs. Mr Swanson's evidence was that it was unclear what was sent to LzLabs, although at least one CPX package was uploaded to the FTP server. In cross-examination, he agreed that Mr Stephens' conclusion was a fair one.

775. On the limited evidence available, I find that the unscrubbed materials were uploaded initially to the FTP server but that they were deleted without LzLabs having access to them. On that basis, it could not be said that they were transferred out of Winsopia's Enterprise.

776. It follows that IBM have not established any breach of the ICA in respect of this item.

Item 45: DR-171

777. DR-171 was opened on 13 August 2020, in response to which Winsopia packaged up material from a customer, scrubbed it using CPX and sent it to LzLabs. However, these load modules included 22 PL/I CSECTs from the PL/I or Language Environment runtime libraries that were not removed by CPX because they were not on the scrub list, as reported by Mr Bowler in DR-282 opened on 9 October 2020.

778. On 16 October 2020 Mr Bowler added the following comment on DR-282:

> "I discovered some more unredacted csects in that library … It's likely there are more that I haven't found yet. What we really should be doing is to scrub all csects that begin with CEE, IGZ, or IBM, with the exception of IGZUOPT, CEEUOPT, CEEROPT, CEESTART, CEEMAIN, and CEEFMAIN. I understand from Chris Palmer when I discussed this with him three or four years ago that it would need some CPX development to be able to specify prefixes instead of complete names, but I don't know if this is still the case."

779. By a comment in response on 19 October 2020, Mr Whittingham stated that he had raised an enhancement request to cover this issue. In cross-examination, Mr Whittingham accepted that this was a scrubbing failure by Winsopia:

> "Q. Yes, so there had been a scrubbing failure by Winsopia; do you agree?
>
> A. Yes.
>
> Q. And Mr Bowler had picked it up?
>
> A. Yes.
>
> Q. And that is why you updated the scrub list on 22 October with a whole bunch of CSECTs?
>
> A. Yes."

780. The experts agree that the CPX scrub list was enhanced shortly after DR-282 to implement pattern-based entries in LZSLIST, which resolved this issue.

781. From the above evidence, it is clear that the inadequately scrubbed load modules, containing IBM PL/I CSECTs, were sent by Winsopia to LzLabs in response to DR-171.

782. The experts agree in their second joint statement that these CSECTs are provided by IBM in the SCEELKED and SCEEBIND libraries provided with z/OS. As such, they are ICA Programs as defined by the ICA.

783. It follows that:

i) Load modules sent by Winsopia to LzLabs contained PL/I CSECTs that were ICA Programs within the meaning of the ICA.

ii) Winsopia's supply of such materials to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

Item 46: Scrubbing failures

784. This allegation concerns three alleged scrubbing failures.

785. The first, recorded in a Bugzilla ticket dated 21 October 2020, concerned an incident in which Winsopia used CPX to process a number of load libraries in response to DR-191 but scrubbing failed in relation to certain C programs with the result that IBM copyright material was found by LzLabs in one of the load modules.

786. In cross-examination Mr Whittingham agreed that this was an example of a scrubbing failure:

> "Q. So what's happened here, see if you agree with me, is that Winsopia has attempted to scrub a number of load libraries in

> response to a DR, DR-191, but scrubbing has failed in relation to some C programs with the result that what's described as IBM copyright material has been found by LzLabs in one of the load modules; do you agree with that?
>
> A. Yes."

787. As a result of this scrubbing failure, Mr Whittingham made a commit to the Git repository, adding a substantial number of additional CSECTs to the scrub list.

788. Mr Althen of LzLabs UK raised this issue with Mr Rockmann and others at LzLabs by email dated 21 October 2020:

> "We have discovered an issue in the CPX Load Module scrubbing process. Not all Copyright Material is being removed. We have obviously imported those load modules on a number internal and [a customer] based SDM nodes. Do we have to do something, remove them, clear instances out ?"

789. At the request of Mr Rockmann, Mr Althen sent an SDM AMBLIST output, identifying many IBM CSECTs containing IBM copyright statements that had been missed from the scrubbing exercise. The output included, in relation to the IBM CSECTs: (i) a hexadecimal dump of the binary content code of the CSECT, (ii) a plain text interpretation of any EBCDIC encoding and (iii) a plain text interpretation of any ASCII encoding.

790. On the following day, Mr Rockmann sent an email to Mr Althen, stating:

> "Given the tight timelines of this engagement we currently have to continue. We will need to clean up later on."

791. Mr Swanson and Mr Stephens agree that all but four of the unscrubbed CSECTs were members of the SCEELKED or SEZACMTX libraries distributed with z/OS. As such, they were ICA Programs as defined by the ICA. Although they appear to have originated with a third-party customer, Mr Whittingham acknowledged that the load modules were processed by Winsopia using CPX and sent to LzLabs.

792. The second allegation concerns OS/VS COBOL CSECTs. On 22 October 2020 Mr Bowler commented in a Jira ticket:

> "I am sorry but the load libraries in DR-189 package … will have to be sent back to Winsopia and scrubbed again because they contain some proprietary csects. Please can you request a redelivery and advise Winsopia that all csects whose names begin with ILBO must be added to the scrub list. "

793. The following morning, Mr Vitale of LzLabs UK sent an email to Mr Whittingham regarding CSECTs that were outside the conventional three letter prefix and needed to be added to the scrub list:

> "In a recent CPX package we had some OS/VS COBOL programs it appears they still contain some proprietary IBM csects. Can you update the scrub list to include ILBO*,
>
> The load library in question on WINSP system is …
>
> Once you have tested and the csects are scrubbed correctly, can you recreate a CPX package with the scrubbed library."

794. In cross-examination, Mr Whittingham accepted that the relevant CSECTs in fact used a standard IBM prefix, albeit a four letter prefix, and would have been caught by the use of wild cards, not yet implemented. He added all CSECTs he could find that shared the relevant OS/VS COBOL prefix to the scrub list.

795. On 23 October 2020, Mr Whittingham circulated an email, stating:

> "As mentioned in the meeting on Thursday, it has become apparent that the LZSLIST entry in CPX has become out of date. I have spent the last couple of days adding all the CSECTs I could discover that were missing.
>
> I would like to request that, for the time being, scrubbed libraries, especially from newly arrived customer files, are scanned for copyrighted CSECTs using … You can see the JOB I have been using … brazenly plagiarised from something John put together (thanks JB!) Please notify me of any new CSECTs which need to be added.
>
> I see this as an interim solution until I can implement something more permanent."

796. Mr Swanson and Mr Stephens agree that the unscrubbed CSECTs were members of the SCEELKED or SEZACMTX libraries distributed with z/OS. As such, they were ICA Programs as defined by the ICA. As above, although they appear to have originated with a third-party customer, Mr Whittingham acknowledged that the load modules were processed by Winsopia using CPX and sent to LzLabs.

797. The third allegation concerns IMS RESLIB CSECTs. DR-216 was opened on 8 September 2020, requesting Winsopia to process and scrub a customer dataset, and create a new CPX package for the SDM. Winsopia duly created a new CPX package and sent it to LzLabs using the LzLabs FTP server.

798. By email dated 25 October 2020 Mr Vitale informed Mr Maddison of Winsopia that the CPX package created by Winsopia included the IMS SDFSRESL load libraries, in which some modules had been scrubbed but others still contained IBM copyright messages.

799. It is common ground that IMS Versions 12, 14 and 15 are ICA Programs. SDFSRESL is a library that contains standard IMS resources supplied as part of

IMS. It follows that these IMS CSECTs are components of an ICA Program, and therefore an ICA Program.

800. In summary on this issue:

i) Load modules sent by Winsopia to LzLabs contained IBM CSECTs that were ICA Programs within the meaning of the ICA.

ii) Winsopia's supply of such materials to LzLabs constituted breach of clauses 4.1, 4.1.2(b) and/or 4.1.3(b) of the ICA.

Item 47: @@TRGLOC CSECT

801. The experts agreed the technical facts. A customer used CPX on their mainframe to process their application load modules and sent a CPX package to Winsopia. The package contained an unscrubbed CSECT @@TRGLOC, a stub provided by IBM in the SCEELKED library with z/OS. Once identified, Winsopia deleted the original package and modified CPX to scrub that CSECT and similar 'trampoline' modules. A further scrub of the load modules revealed modules with third-party copyright in them. The customer then modified their CPX processing and re-sent the CPX output to Winsopia, who undertook further scrubbing and forwarded the same to LzLabs.

802. IBM relies on an internal LzLabs email sent by Mr Bond on 6 November 2020, noting that the @@TRGLOC section was not redacted by CPX. However, it is not clear that this particular third-party program was processed by Winsopia. An earlier internal LzLabs email sent by Mr Bowler on 22 October 2020 referred to confusion as to which version of the module was under consideration. When asked about this in cross-examination, Mr Bond stated that, although he did not have a clear recollection, he believed that the customer scrubbed the module, transferred it to the customer's copy of the SDM and attempted to executed it. It is clear that the DR process was used but not clear that the inadequately scrubbed version of the load module was sent by Winsopia to LzLabs.

803. On the evidence available, IBM has not established this allegation.

Item 48: PARMLIB & PROCLIB

804. PARMLIB comprises datasets that enable programmers to select parameters for IBM or other third-party software. PROCLIB is a dataset containing IBM provided sample JCL procedures that can be customised by programmers.

805. A third-party customer load module containing its specified PARMLIB and PROCLIB was included in a CPX package sent to LzLabs. On 15 December 2020 DR-415 was opened by LzLabs, requesting Winsopia to download files and make them available to LzLabs, containing PARMLIB and PROCLIB members. Mr Lynch responded, explaining that the files would contain proprietary material. He explained in his witness statement that the files in question contained the PARMLIB reference; from that, he knew without

looking at the content of the files that they would include IBM material relating to z/OS parameters, and so they could not be sent back to LzLabs developers.

806. The experts agree in their second joint statement that there is no evidence that PARMLIBs or PROCLIBs from Winsopia were sent to LzLabs. Mr Stephens' view is that the above email chain and Mr Lynch's witness statement indicate that PARMLIB and PROCLIB were not made available to SDM developers. In cross-examination, Mr Swanson accepted that he had not seen any evidence to the contrary.

807. On that basis, IBM has not established this allegation.

Use outside Enterprise and beyond Designated Machine

808. IBM's case is that Winsopia allowed employees of LzLabs and LzLabs UK to access its mainframe, and used licensed ICA Programs on machines other than the Designated Machine as defined in the ICA.

809. The defendants' case is that the secondments and use of the ICA Programs were permitted in accordance with the terms of the ICA.

810. A number of the allegations were abandoned by IBM before and after the hearing, namely Paragraphs 44.1, 44.3, 44.4 and 44.7 of the Technical Particulars. There remain only three live issues, namely, secondments to Winsopia (Brad Taylor and Justin Bendich) and remote use of the Winsopia "Pizza Box".

Item 49: Brad Taylor (Paragraph 44.2 of the Technical Particulars)

811. The allegation is that Mr Taylor, a Senior Software Developer at LzLabs, accessed the Winsopia mainframe and used ICA Programs, including (i) writing REXX programs which were stored on the mainframe; and (ii) initiating GTF traces between a CICS instance on the mainframe and a NUC at Winsopia, for the purpose of developing CICS support within the SDM.

812. Mr Taylor was the technical lead at LzLabs for development of the design, coding and testing of Legacy Transaction Environment ("LTE"), LzLabs' implementation of CICS, in a Linux rather than z/OS environment. CICS allows applications and resources to be distributed across CICS regions which behave like servers. ClCS applications may require communications with other CICS regions that are either located within the same operating system or in remote systems. For CICS applications to run smoothly on LTE, it was necessary for LTE to behave like a remote CICS region and to implement the same intercommunication functionality. In cross-examination, Mr Taylor stated that he was trying to achieve 80% of the functionality offered by CICS and to simulate the behaviour of a real CICS instance when communicating with CICS on the mainframe. He wanted the LTE software to be able to send and receive messages to mainframe-based CICS instances and for CICS to understand and respond appropriately to those messages.

813. Mr Taylor was seconded to Winsopia on 27 to 29 August 2014, 16 to 18 September 2014 and 29 September to 2 October 2014, pursuant to the Supply of Staff Agreement dated 14 August 2014 between Winsopia and LzLabs and a letter of secondment dated 20 August 2014.

814. Before his first visit, Mr Taylor sent a New Unit of Computing device (Brad's "NUC"), which had an early SDM version installed on it, to Winsopia to be installed and configured within the Airlock. The Airlock was a secure network which allowed restricted communication with certain devices located at Winsopia. Once inside the Airlock, Mr Taylor was able to connect to the NUC using VPN and a connection from LzLabs in Zurich. He brought with him a hard drive with a copy of the SDM on a virtual machine, on which he had installed "SNAP-IX", a tool licensed to LzLabs by a third party company that provided an implementation of the SNA LU6.2 protocol, enabling connectivity to Linux systems using the TCP/IP communication protocol. Winsopia engineers set up CICS regions on the mainframe so that Mr Taylor could carry out LTE-to-CICS testing and achieve a connection between the NUC and a CICS region on the mainframe.

815. Mr Taylor agreed in cross-examination that it was possible for devices in the Airlock to communicate with the mainframe and he was given access to the mainframe by Mr Rockmann. The NUC had to communicate with the mainframe to perform its CICS tests; indeed, the purpose for which the NUC was installed in the Airlock was so that Mr Taylor could use it to communicate with the mainframe at Winsopia.

816. Mr Taylor's evidence as to the work he carried out at Winsopia was as follows:

> "I would run tests, review the visible communications between LTE and CICS via SNAP-IX and CICS using SNA tunnelling over IP, make changes to the SDM source code based on those communications, recompile the SDM, and then rerun further LTE-to-CICS test communications and transactions. In more detail, I would start up CICS regions on the mainframe, start up the prototype version of the LTE on my VM, run a transaction on CICS and have it try to communicate with LTE, analyse any errors or issues that occurred by running and then reviewing a GTF trace on the mainframe, and fix any bugs I identified on the SDM. I would then try again, effectively repeating the process, or try something different…
>
> I recall that by the end of my first secondment I had established a functional LTE-to-CICS connection using SNAP-IX. I then returned to LzLabs to continue developing LTE. During this period, after my first secondment, I established further SSH connections with the Airlock to configure and install software on the NUC, e.g. further test versions of LTE…
>
> I wrote a REXX program tool on the Winsopia mainframe. It extracted all of the EXEC CICS and EXEC SQL commands out of what I think was a customer's source code…

> Ultimately the secondments allowed me to achieve LTE-to-CICS interoperability using both SNAP-IX and IPIC."

817. In cross-examination, Mr Taylor explained that the secondments enabled him to set up his own tests on the mainframe and view the results of unredacted GTF traces, so that he could develop LTE more efficiently and more quickly:

> "Q. Yes. So I mean, ultimately, what you wanted to do, and in fact what you ended up doing was setting up your own tests on the mainframe and seeing the results of traces directly yourself on the mainframe?
>
> A. Correct. That was the purpose of the secondment.
>
> Q. Yes. So I mean, upper level, I mean, the purpose of your secondment was so that you could develop LTE more efficiently and more quickly?
>
> A. Correct.
>
> Q. And it may be obvious, but it wasn't to deal with, for example, a customer emergency? Things weren't breaking at the customer side?
>
> A. No, no customer had this code.
>
> Q. No. And you weren't assisting in running the DR system?
>
> A. I'm not sure what you mean by "assisting in running".
>
> Q. You weren't going over to Winsopia to help them run the DR system?
>
> A. No, I was not.
>
> Q. Or to assist with a support issue relating to one of -- with Lz's customers?
>
> A. No, I was not.
>
> Q. It was all about you and your work on LTE, wasn't it?
>
> A. Yes, it was."

818. Mr Taylor accepted in cross-examination that, while on secondment, he was able to work on developing the SDM code simultaneously with the mainframe access. He was able to look at unredacted GTF traces and make substantive changes as a direct and immediate result of what he saw on the mainframe. The REXX program he wrote extracts, lists and counts EXEC CICS commands and EXEC SQL commands. Although he was not permitted to make any SDM source code changes whilst at Winsopia, as soon as he returned from secondment he committed the changes he had made to the Git repository.

819. The experts agreed in their second joint statement that:

i) while seconded to Winsopia, Mr Taylor accessed the Winsopia mainframe to write REXX programs, and to run GTF traces;

ii) following Mr Taylor's secondment to Winsopia, LzLabs requested and received the REXX source code and redacted versions of the GTF traces via the DR system; and

iii) there is no evidence that Mr Taylor's use of the mainframe, and the subsequent transfer of the materials that he created, led to IBM materials being provided to LzLabs.

820. The defendants' case is that whilst he was at Winsopia, Mr Taylor was part of Winsopia's Enterprise; therefore, his use of the mainframe was for the Customer's authorised use and complied with IBM's terms regarding ICA Programs. Although he remained employed by LzLabs, he was reporting to Winsopia, using Winsopia equipment, and working with other Winsopia staff in order to complete tasks, that fell within Winsopia's responsibility under the Services Agreement but for which it lacked the necessary expertise.

821. I reject the defendants' submissions for the following reasons. Firstly, the Supply of Staff Agreement provided that each secondee would remain employed by LzLabs for all purposes and would continue to be paid by LzLabs. Although it also provided that each secondee should report to Mr Rastall at Winsopia and carry out work at his direction, in fact, as Mr Taylor confirmed in cross-examination, this did not accurately reflect the conditions under which he worked at Winsopia or the work that he carried out. Mr Rastall stated in cross-examination that Mr Taylor was issued with an electronic pass, which allowed him to come and go as he pleased and gave him access to the mainframe room at Winsopia. He continued working in furtherance of his development of the LTE, a project for LzLabs. Therefore, this was not a genuine secondment.

822. Secondly, even if treated as a consultant, Mr Taylor did not carry out work that Winsopia was entitled to do under the terms of the ICA. As set out in respect of Item 13 above, use of the GTF facility to collect VTAM buffer traces that disclosed details of the CICS-to-CICS communications amounted to reverse engineering. If Mr Taylor had been acting as an employee or agent of Winsopia during the secondments, his activities would have amounted to reverse-engineering of the CICS Transaction Server for z/OS, v.5, an ICA Program. Further, Mr Taylor's activities were not permitted pursuant to Articles 5(3) and/or 6 of the Software Directive for the reasons set out in respect of Item 13 above. Therefore, the defendants' arguments in reliance on the decision in *SAS* do not provide any defence.

823. Thirdly, the defendants correctly point out that the GTF traces were reviewed by an external lawyer, Mr Hedley, and redacted before being sent to LzLabs through DR0235. But that ignores the fact that Mr Taylor, an LzLabs' employee, was given access to study the unredacted CICS-to-CICS traces at Winsopia, so that he could understand how the communications worked from one CICS region to another CICS region; in particular the data formats in which

CICS sent and received instructions and data once a connection (or handshake) was established.

824. Fourthly, Mr Taylor's evidence was that he took a copy of the SDM into Winsopia, which he worked on whilst he had access to the mainframe. The Git commit records show that Mr Taylor made a number of commits to the SDM Git repository shortly after each of his visits to Winsopia. Professor Donaldson examined some of the Git commits and concluded in his third report that they mainly related to establishing, testing and debugging support for inter-system communications ("ISC") between the LTE running on the SDM and a remote CICS region and integration of the SNAP-IX tool. The changes made in the Git commits included (i) the addition of a new code file implementing CQP, an IBM protocol; and (ii) the addition of code implementing XLN, an IBM feature whereby, when an ISC session is established between two CICS regions, the name of the system log being used on each system is passed to the other system. A screenshot of the Git record for 10 October 2014 shows that Mr Taylor made 517 changes to CQP and 938 changes to XLN. In cross-examination Professor Donaldson agreed that Mr Taylor's examination of the mainframe at Winsopia could have informed his understanding of how the ISC worked, so as to prompt those Git commit changes. Given the nature and timing of the changes made, I find that on the balance of probabilities Mr Taylor used the knowledge gleaned from his study of the CICS-to-CICS communications on the Winsopia mainframe to develop the LTE on the SDM.

825. In summary, regardless of whether he was formally seconded to Winsopia, Mr Taylor remained an employee of LzLabs and his activities entailed development of the LTE for LzLabs as part of the SDM. Access to the Winsopia mainframe and permission to use the ICA Programs given by Mr Rockmann to Mr Taylor amounted to a breach of clauses 4.1 and 4.1.2(b) of the ICA.

Item 50: Winsopia Pizzabox (Paragraph 44.5 of the Technical Particulars)

826. The allegation is that Mr Christian Wehrli of LzLabs, employees of LzLabs UK, and members of the "QA team" had access to the Winsopia mainframe and unscrubbed IBM proprietary material through use of the "Winsopia Pizzabox", which was not a Designated Machine, in breach of clause 4.1 of the ICA.

827. The Winsopia Pizzabox was a data centre server, with dimensions similar to a pizza box, stored at Winsopia's premises. From around February 2015, the SDM was installed on the Winsopia Pizzabox and was used by Winsopia to conduct testing.

828. The experts second joint statement includes the following matters of agreement:

i) Certain LzLabs employees, including Mr Wehrli and members of the "QA team", had access to the Winsopia Pizzabox through the Customer Support Centre ("CSC") network. Such access was recorded using a Shell Control Box ("SCB").

ii) The Winsopia Pizzabox was connected to the Winsopia corporate network. The Winsopia mainframe was also connected to the Winsopia

corporate network. By virtue of being on the same network, communication between the Pizzabox and the mainframe was possible.

iii) Emails and audio files suggest that communication between the Winsopia Pizzabox and the Winsopia mainframe may have been possible via NJE or MQ.

iv) Emails suggest that the Winsopia Pizzabox was used to store unscrubbed load modules.

829. In cross-examination, Mr Wehrli stated that the Pizzabox could access the disks in the mainframe and that unscrubbed modules from the mainframe were found on the Pizzabox. He agreed that he used his access to the Pizzabox to provide material from Winsopia to LzLabs developers, Texas Wormhole and OnTarget and that some of that material was created on the Winsopia mainframe. However, he stated that he habitually checked the artefacts that were sent out, particularly from the Pizzabox or from the FTP server to the developers, to ensure that no proprietary materials were included. There is no evidence that any of the unscrubbed modules were sent by Mr Wehrli from the Pizzabox to LzLabs developers.

830. The defendants submit that any remote connections from the Pizzabox to the mainframe would have been recorded by the Shell Control Box ("SCB") logs. There is no evidence in the SCB logs that the Pizzabox was ever used by any LzLabs or LzLabs UK employee to connect to or use the Winsopia mainframe. Mr Werhli's evidence was that he was able to connect to the Winsopia Pizzabox with a workaround, when the VPN tunnel was down, but that access was still obtained through the secure connection to the Winsopia network. Professor Donaldson's opinion, based on his analysis of the SCB logs, was that there was no evidence that the Pizzabox was used as a means of connecting to the Winsopia mainframe. Although some doubts were raised as to the reliability of the SCB logs, there was no alternative data that undermined the analysis relied on by Professor Donaldson. On that basis, I find that there is no evidence that LzLabs or LzLabs UK used the Pizzabox to access the Winsopia mainframe.

831. In summary, I find that the admitted storage of unscrubbed modules on the Pizzabox constituted a breach of clause 4.1.1(a) of the ICA; otherwise, this allegation is not established.

Item 51: Justin Bendich (Paragraph 44.6 of the Technical Particulars)

832. The allegation is that during a visit to Winsopia in March 2019, Mr Bendich, a developer at LzLabs, was permitted by Winsopia to use its mainframe, in breach of the ICA.

833. Mr Bendich did not provide a witness statement or give oral evidence. His email of 24 March 2019 describes his visit to Winsopia, during which he described how GTF traces were run on the Winsopia mainframe and compared with similar traces on the SDM:

> "GTF (the Generalized Trace Facility) will simply stop after a while, even if you give it plenty of disk space…
>
> Each load module is created from COBOL source code. COBOL's implementation requires each produced load module to include IBM-copyrighted run-time code (CSECTs). By our Code of Conduct, this code is not allowed to enter LzLabs. Therefore, if we want to analyze these traces at LzLabs, we must reliably remove these instructions from the trace. Because the hardware allows only one range, this removal must be performed after the trace is obtained.
>
> The instruction traces also contain IBM-copyrighted DB2 code, some of which resides in the home address space, and some of which resides in the DBM1 address space. I believe the code in the DBM1 address space can be pre-filtered by a suitable SLIP option. We didn't try to do this. With the post-processing software i developed while at Winsopia (see below), it's easy to remove these instructions."

834. Professor Weissman concludes from Mr Bendich's email that he ran GTF traces using the Winsopia mainframe. Mr Stephens agrees with Professor Weissman that Mr Bendich was shown the GTF trace output, and appears to have written a C program to help 'scrub' it, but that neither of these would necessarily require access to a mainframe. Neither Professor Weissman nor Mr Stephens were able to find any datasets or user credentials disclosed on the zPDT server that indicated that Mr Bendich had access to the IBM Mainframe.

835. Mr Bendich sent an email dated 3 May 2019 to Mr Hedley in which he gave an account of his visit to Winsopia, suggesting that the GTF traces might have been run on the mainframe by Mr Maddison of Winsopia, rather than Mr Bendich:

> "Bob Maddison, a Winsopia employee, took GTF traces from the mainframe, put them on the thumb drive, and handed it to me so that i could transfer them onto the NUC. I wrote a lot of code which stayed on the Winsopia NUC. I used this code to process the traces, mostly to eliminate extraneous information, including all IBM-written code (see included e-mail, below). None of this code ever left Winsopia. None of the traces, processed or otherwise, ever left Winsopia."

836. In cross-examination Professor Weissman fairly accepted that, having regard to the above email, it is possible that Mr Maddison, rather than Mr Bendich, ran the GTF trace on the mainframe.

837. The evidence is not clear on this issue. It is unfortunate that Mr Bendich did not provide any witness evidence, which prevented IBM from having an opportunity to challenge it. On balance, I am not satisfied that this allegation is established.

Conclusions on technical breaches

838. In conclusion, my findings on the alleged technical breaches are as follows.

839. Winsopia was in breach of the ICA in that it carried out reverse engineering of the ICA Programs by disassembly, decompilation and translation:

i) Item 1 – IGZCUST – breach established;

ii) Item 2 – LMD – breach established;

iii) Item 3 – CICS Control Blocks Document - breach established;

iv) Item 4 – EXEC DLI - breach established;

v) Item 5 – IBM Binder Software - breach established.

840. Winsopia was in breach of the ICA in that it carried out reverse engineering through the systematic creation and analysis of compiler listings:

i) Item 6 – IGZCIVL COBOL runtime module – breach established;

ii) Item 7 – CICS Translators – breach established;

iii) Item 8 – Floating point rounding rules – breach established;

iv) Item 9 – IBM PL/I Compiler – breach established;

v) Item 10 – XML Parse statements – breach established;

vi) Item 11 – COBOL initialisation, branching and I/O declaratives – breach established;

vii) Item 12 – PL/I condition handling – breach established.

841. Winsopia was in breach of the ICA in that it carried out reverse engineering through the systematic use of traces, dumps, slip traps, packet sniffing and other debugging tools techniques:

i) Item 13 – CICS to CICS communications – breach established;

ii) Item 14 – AMBLIST analysis of CICS stubs – breach established;

iii) Item 15 – Colesoft XDC and COBOL initialisation – breach established;

iv) Item 16 – XDC and IMS – breach established;

v) Item 17 – SLIP traps and CICS – breach established;

vi) Item 18 – SLIP traps and COBOL – breach established.

842. Winsopia was in breach of the ICA in that it carried out reverse engineering by copying of IBM source code, macro expansions and copybooks:

i) Item 19 – DR -246 – breach established;

ii) Item 20 – DR-10237 – breach established;

iii) Item 21 – DR-2753 - breach established;

iv) Item 22 – DR-2771 - breach established;

v) Item 23 – DR-2796 - breach established;

vi) Item 24 – DR-3280 - breach established;

vii) Item 25 – DR-4281 - breach established;

viii) Item 26 – DR-4322 - breach established;

ix) Item 27 – DR-0847 - breach established;

x) Item 28 – DR-715 – breach established;

xi) Item 29 – DR-753 – breach established;

xii) Item 30 – DR-756 - breach established.

843. Winsopia was in breach of the ICA by transferring "unscrubbed" and/or partially "scrubbed" materials containing IBM mainframe software:

i) Item 31 – Epiphany – allegation not pursued;

ii) Item 32 – Db2 catalog table metadata – breach not established;

iii) Item 33 – DSS dump – breach not established;

iv) Item 34 – Kednos – breach established;

v) Item 35 – CSECTS omitted from scrubbing – breach established;

vi) Items 36 & 42 – unscrubbed CSECTs – breach not established;

vii) Items 37 & 40 – IMS PROCLIB & DLIBATCH – breach established;

viii) Item 38 – DFHEI1 module – breach established;

ix) Item 39 – IGZXANE – no material breach established;

x) Item 41 – IGZXNE3N – breach established;

xi) Item 43 – CEEBETBL, CEEBLLST, IBMPINPL & CEESG* - breach established;

xii) Item 44 – DR-4617 – breach not established;

xiii) Item 45 – DR-171 – breach established;

xiv) Item 46 – scrubbing failures – breach established;

xv) Item 47 - @@TRGLOC CSECT – breach not established;

xvi) Item 48 – PARMLIB & PROCLIB - breach not established.

844. Winsopia was in breach of the ICA through use outside its Enterprise and use beyond the Designated Machine:

i) The allegations in Paragraphs 44.1, 44.3, 44.4 and 44.7 of the Technical Particulars were abandoned;

ii) Item 49 – Brad Taylor – breach established;

iii) Item 50 – Winsopia Pizzabox – breach established in part;

iv) Item 51 – Justin Bendich – breach not established.

Section VI - Wrongful procurement of breach

845. The pleaded case is that LzLabs, LzLabs UK, Mr Cresswell, Mr Rockmann and/or Mr Moores procured the breaches of the ICA. It is alleged that each and every breach by Winsopia of the ICA was undertaken at the direction, instruction or request of LzLabs (with the assistance of LzLabs UK), Mr Cresswell, Mr Rockmann and/or Mr Moores, each of whom knew and intended that the breaches should occur, or were reckless as to the same.

846. It is alleged that Mr Cresswell and Mr Rockmann are liable for procuring breaches of the ICA by virtue of their capacity as directors of Winsopia. It is said that they knew that Winsopia had acquired an IBM mainframe and entered into the ICA for the purposes of developing and operating the SDM and that Winsopia intended to reverse assemble, reverse compile or otherwise reverse engineer parts of the software or to allow others to do so. It is alleged that it is inconceivable that they did not review the terms of the ICA and appreciate that the proposed activities amounted or would give rise to breach of its obligations, or they were reckless in respect of the same.

847. Further, it is alleged that Mr Moores, the ultimate beneficial owner of LzLabs, Winsopia and LzLabs UK, exerted control over the corporate defendants, so that they followed his instructions in relation to the development of the SDM. It is said that Mr Moores directed and coordinated the development of the SDM, participating in the detail of its development, testing and marketing. In so doing, Mr Moores gave instructions and took steps that knowingly and intentionally induced and/or facilitated breaches of the ICA.

848. The allegations of wrongful procurement of breach of the ICA are denied. The defendants' case is that Winsopia and LzLabs went to considerable lengths to maintain operational separation, to establish and follow formal processes for the exchange of information and documents and to monitor and segregate communications between them, in order to ensure that the provision of services by Winsopia to LzLabs was at all times fully compliant with the ICA.

849. Reliance is placed by the defendants on the clean room process. The Services Agreement between LzLabs and Winsopia provided that the Codes of Conduct

would prevail in all cases and must be complied with by both parties at all times. The Codes of Conduct imposed restrictions to ensure separation of the work by LzLabs and Winsopia and respect for third party intellectual property rights, including restrictions on communications between Winsopia staff and LzLabs, restrictions on access by LzLabs to Winsopia's premises and the mainframe, and external legal oversight. Pursuant to the DR procedure, a request from an LzLabs developer for information or documents from Winsopia could only be made and responded to via a supervised response procedure. Any response given by Winsopia to such requests was stored and processed in an electronic DR system. Pending review by external lawyers, responses were redacted as necessary, to ensure that they did not contain any materials whose supply to LzLabs would be prohibited by the Code of Conduct, including any ICA Program or part thereof.

850. The defendants' case is that they relied reasonably and in good faith on the clean room processes and thereby believed that Winsopia would not breach any of its obligations under the ICA. LzLabs UK did not exist until 23 September 2015 and did not assist Winsopia in relation to the acts giving rise to breach of the ICA. Mr Cresswell believed at all material times that all activities undertaken by Winsopia complied with the terms of the ICA and there was never any activity of which he was aware which he thought might have breached the ICA. Mr Rockmann did not in his capacity as a director and executive officer of Winsopia ever direct, instruct or request Winsopia or any of its employees to engage in any activities that amounted to or gave rise to breaches of the ICA. Mr Moores did not know of or intend any breach of the ICA to occur nor was he reckless about any such breach. He did not instruct or take steps that knowingly and intentionally induced and/or facilitated breaches of the ICA. At all times, Mr Moores understood Winsopia to be acting in compliance with the ICA and the applicable law.

851. Further, Mr Cresswell and Mr Rockmann each rely by way of defence on the principle in *Said v Butt* [1920] 3 KB 497, namely: (i) they were officers and/or employees of Winsopia; (ii) they were acting in good faith in the performance of their duties to Winsopia; and (iii) they were acting within the scope of their authority in respect of the activities of Winsopia about which complaint is made.

Applicable legal principles

852. The tort of procuring a breach of contract requires: (i) a breach of contract; (ii) conduct to procure or induce that breach; (iii) knowledge or recklessness as to the existence of the relevant term in the contract; and (iv) intention that the conduct induced or procured would result in a breach of that term: *OBG v Allan* [2007] UKHL 21 per Lord Hoffmann at [39]-[44]; per Lord Nicholls at [172], [191]-[193] & [202].

853. The test was helpfully clarified by the Court of Appeal in *Kawasaki Kisen Kaisha Ltd v James Kemball Ltd* [2021] EWCA Civ 33 per Popplewell LJ at [20]-[21], citing with approval *Global Resources Group v Mackay* [2004] CSOH 149, 2009 SLT 104 at [11]-[14] per Lord Hodge: "A commits the delict or tort of inducing a breach of contract where B and C are contracting parties and A, knowing of the terms of their contract and without lawful justification

induces B to break that contract." The ingredients of the tort were identified as follows:

i) there must be a breach of contract by B;

ii) A must induce B to breach the contract with C by persuading, encouraging or assisting them to do so;

iii) A must know of the contract and know that their conduct will have that effect;

iv) A must intend to procure the breach of contract either as an end in itself or as the means by which he achieves some further end;

v) if A has a lawful justification for inducing B to break his contract with C, that may provide a defence against liability.

854. For the reasons set out in the technical breaches section of this judgment, I have found that Winsopia was in breach of the ICA. There is no defence of lawful justification. Therefore, the material questions in respect of each defendant are whether the necessary elements of (ii) inducement, (iii) knowledge and (iv) intent are established so as to give rise to liability.

855. The breach of contract must be induced by the defendant. Liability for procuring a breach is an accessory liability where the primary wrong is the breach of contract. Mere prevention of contractual performance will not found liability. There must be a sufficient causal connection between the defendant's act of persuasion, encouragement or assistance and the breach of contract: *OBG* per Lord Hoffmann at [36]; Lord Nicholls at [178]-[180]; *Kawasaki* per Popplewell LJ at [32]-[33].

856. The defendant must know that his action will result in a breach of contract. It is not sufficient to show that the defendant knows that he is procuring an act which, as a matter of law or construction of the contract, amounts to a breach. The defendant must know the essential facts which make the act unlawful. Mere negligence or ignorance is not sufficient. However, recklessness or deliberate disregard of available knowledge, including turning a blind eye, is sufficient: *OBG* per Lord Hoffmann at [39]-[41]; Lord Nicholls at [191]-[192].

857. The issue whether a defendant's state of mind was sufficient to establish relevant knowledge was considered recently in *Lifestyle Equities CV v Ahmed* [2024] UKSC 17 per Lord Leggatt (giving the judgment of the court) at [101] and [107]-[109]:

> "[102] Liability for procuring a breach of contract does more than provide an analogy with liability for procuring another person to commit a tort. As Lord Hoffmann and Lord Nicholls (with whom the other law lords agreed on this issue) made clear in *OBG Ltd v Allan*, both forms of liability rest on the same underlying principle. This principle was stated in *Lumley v Gye* by Erle J, at p 232:

> "It is clear that the procurement of the violation of a right is a cause of action in all instances where the violation is an actionable wrong ... he who procures the wrong is a joint wrongdoer, and may be sued, either alone or jointly with the agent, in the appropriate action for the wrong complained of."

Although in this statement Erle J did not use the word "malicious", it is clear from the context and from his judgment as a whole that - in common with all the members of the court - Erle J was concerned with cases where someone "maliciously" procures a wrong and that his statement of the law tacitly assumes that the defendant's act is "malicious". It is also apparent, as Lord Watson observed in *Allen v Flood* [1898] AC 1, 96, that the judges in *Lumley v Gye* regarded "malice" as "signifying in law, not that the defendant had been actuated by a bad motive, but that he had procured the commission of an act which he knew to be illegal." At the subsequent trial Lumley's claim in fact failed, as the jury found that Gye was not aware, when he engaged Wagner, that she had no right to terminate her contract with Lumley: see SM Waddams, "Johanna Wagner and the rival opera houses" (2001) 117 LQR 431, 455-456. Although as a matter of causation Gye had procured the commission of an act which was a breach of contract, he did not know that the act which he procured was a breach of contract. He therefore lacked the state of mind required for accessory liability.

…

[108] A further distinction needs to be drawn. In accordance with the principle that ignorance of the law is no excuse, liability cannot depend on whether the defendant knows that the act done by the primary wrongdoer is against the law. When courts refer to a requirement of knowledge that an act is wrongful, they must generally be taken to mean, not that knowledge of the law is required, but that the defendant must know the essential facts which make the act unlawful. The same applies to references to intention. Lord Templeman's reference to a defendant who "intends and procures ... that infringement shall take place" should be understood in this sense. Lord Templeman should not be taken to mean that the defendant must have a sufficient knowledge of copyright law to know that the act which he intends to bring about will be a breach of copyright; only that the defendant must know the facts which make that act a breach of copyright.

[109] Although in *MCA Records* Chadwick LJ adopted and relied on Lord Templeman's statement of principle quoted at para 106 above, he did not discuss what precisely the defendant must intend in order to be liable for procuring an infringement. It does not appear that any argument was specifically directed to this point. Chadwick LJ assumed that it was sufficient to make

> Mr Young liable for procuring infringements of copyright that "it was Mr Young's purpose and intent, in doing what he did, that the Chess recordings should be copied and marketed through [the company]": para 36 (and see also para 61). That, in my opinion, was incorrect. Applying the principle of accessory liability stated by Erle J in *Lumley v Gye*, it was not enough that Mr Young knew and intended that the relevant recordings should be copied and issued to the public. It was also necessary to establish that Mr Young knew (or deliberately turned a blind eye to) all the essential facts which made copying and marketing the recordings a breach of copyright. Those facts included the fact that the company controlled by Mr Young did not either own the copyright or have a licence from the owner to copy and issue copies to the public of the relevant recordings."

858. Thus, the knowledge element of the tort is satisfied if the defendant was, or should have been, aware of the essential facts that amount to a breach of contract, including the material terms of the contract, even if he was not aware of the applicable rules and principles of law.

859. As to the mental element of the tort, the defendant must intend to bring about the breach of contract, either as an end in itself, or at least as the means to some further end. Mere causative participation is not enough: *OBG* per Lord Hoffmann at [42]; Lord Nicholls at [191]; *Kawasaki* per Popplewell LJ at [34].

860. In principle, it is open to a defendant to show that he did not have the requisite intention to cause economic harm by evidence of a firm belief, based on legal advice, that he was entitled to act as he did: *Meretz Investments v ACP Ltd* [2007] EWCH Civ 1303, per Arden LJ at [122]-[127].

861. Generally, in order to support such a defence, it would be incumbent on the defendant to disclose the relevant legal advice and adduce evidence that he relied on such advice: *Digicel (St. Lucia) Limited v. Cable & Wireless Plc* [2009] EWHC 1437 (Ch) per Morgan J at [25]; *Generics (UK) Ltd v Competition and Markets Authority* [2018] CAT 4 per Roth J at [200].

862. I now turn to consider the case of wrongful procurement against each defendant.

LzLabs

863. In this case the evidence is clear that every act on the part of Winsopia which constituted a breach of the ICA, in accordance with my above findings, was an act undertaken at the direction of LzLabs or those controlling LzLabs.

864. Firstly, Winsopia's sole purpose was to provide software development and testing services for LzLabs. LzLabs was incorporated in 2011 to develop what became the SDM. Mr Rockmann agreed that in order to develop and test the SDM LzLabs needed access to a mainframe. Winsopia was set up in 2013, at the suggestion of Mr Rockmann and funded by Mr Moores, to provide access to a mainframe to assist LzLabs with development and testing of the SDM.

865. Notwithstanding Mr Rastall's protestation that Winsopia was set up as an autonomous business, his evidence was clear that this was done at the suggestion of Mr Moores, following a recommendation by Mr Broussard of Texas Wormhole, in furtherance of the SDM project. Arrangements for setting up the company, including funding the purchase of a mainframe by LzLabs, were made under the direction of Mr Moores' legal and financial team and in collaboration with Mr Rockmann, as evidenced by their email exchanges during this period.

866. Secondly, Winsopia was a wholly-owned subsidiary of LzLabs. Winsopia was incorporated on 15 March 2013. By mid-June 2013, Mr Rockmann told Mr Moores that he had agreed a price for the purchase of Winsopia with Mr Rastall. Mr Rastall's evidence was that he considered that he had no alternative but to sell to LzLabs. On 5 July 2013, LzLabs acquired the shares of Winsopia. Mr Rockmann, then the CEO of LzLabs, also became the sole director of Winsopia, effectively controlling both companies.

867. Thirdly, Winsopia had no customers apart from LzLabs and no fixed scope of work. The Services Agreement entered into by LzLabs and Winsopia required Winsopia to provide services that were defined in very broad and general terms in Schedule 1: (i) discovery and QA services to LzLabs as described in the Winsopia Code of Conduct; (ii) software development services to LzLabs in respect of the Agents; and (iii) managerial and operational support and co-operation as necessary, including attendance at meetings and conference calls. The degree of control exerted by LzLabs was exemplified by clause 12 of the Services Agreement, which entitled LzLabs to fund and take conduct of any defence to a third party claim against Winsopia in respect of intellectual property rights or misuse of confidential information.

868. Fourthly, the Services Agreement was signed twice by Mr Rockmann, once on behalf of Winsopia and again on behalf of LzLabs, negating any suggestion of independent operation on the part of Winsopia.

869. Fifthly, Mr Rastall's evidence was that Winsopia used the mainframe and IBM licensed software for the sole purpose of assisting LzLabs in its software development. All of the work that Winsopia did was work which LzLabs instructed it to do.

870. An example of control over Winsopia's activities is Mr Moores' evidence that the SDM development work was allocated from Texas Wormhole to Winsopia because it had access to a mainframe:

> "Q. Now, the reason for allocating work to Winsopia was because to do this Db2 work it was essential to have a mainframe, wasn't it?
>
> A. Well, certainly David thought so, and he persuaded me that it would have been enormously helpful. There may have been other options, but we didn't consider them.

> Q. Well, Mr Moores, it was absolutely essential, wasn't it? You couldn't have got anywhere with Db2 unless you'd had access to a mainframe?
>
> A. I think access to a mainframe would have been crucial."

871. Sixthly, in practice, Winsopia acted at all times at the behest of LzLabs. Mr Cresswell agreed in cross-examination that, at least from 2015 when he joined, Winsopia did not act independently of LzLabs:

> "Q. But would you agree that Winsopia had no relevant role that was independent of LzLabs?
>
> A. Winsopia's only customer was LzLabs, so, yeah, it didn't have a role independent of LzLabs."

872. This was echoed by Mr Moores, who agreed in cross-examination that Winsopia acted under the direction of LzLabs:

> "Q. … throughout the development of the SDM, Winsopia acted at all times under the direction of LzLabs and those controlling LzLabs, didn't it?
>
> A. Yes.
>
> Q. I suggest that it was always the intention that Winsopia would be used for the sole purpose of assisting LzLabs in the development of the SDM.
>
> A. I think that's probably right, at least from the time of the acquisition."

873. It follows that, to the extent that Winsopia carried out activities that amounted to a breach of the ICA, it did so at the direction and under the control of LzLabs. The breaches that have been established by IBM could not be described as isolated errors; their nature and extent, and the duration over which they occurred, are indicative of deliberate and systematic disregard of the terms of the ICA. Most of those activities were directed by LzLabs through the DR process; the requests for test results and information, with which Winsopia was required to comply, were a direct cause of the responses, which entailed reverse engineering and other breaches of the ICA. But even where specific requests were not recorded in DRs, the activities by Winsopia fell within the scope of the discovery and testing work instructed by LzLabs under the Services Agreement. That amounted to active procurement or inducement by LzLabs to Winsopia to carry out acts in breach of the ICA.

874. The defendants seek to rely on the clean room procedures, namely, the Codes of Conduct, the DR system and scrubbing of IBM proprietary materials, as evidence that there was no systematic breach of the ICA. However, those who controlled LzLabs and Winsopia did not take any meaningful steps to ensure that the clean room procedures were effective in practice. The safeguards set

out in the early versions of the Codes of Conduct were eroded over time until there was no effective separation between LzLabs and Winsopia.

875. In particular, the Codes of Conduct were amended to allow LzLabs experts, including LzLabs developers, to be seconded to Winsopia, where they were given access to the Winsopia mainframe and direct communications were permitted between LzLabs developers and Winsopia employees. Mr Wilkinson of LzLabs visited Winsopia in May and June 2016, when he made commits to the SDM Git repository. Following a further visit to Winsopia at the end of March 2019, he sent an email explaining the work he did there to compare SDM and mainframe traces using debugging tools, such as GTF and XDC. Mr Reynard of LzLabs committed code to the Git repository whilst on secondment to Winsopia in April 2017. On 7 April 2017 Mr Janicek at LzLabs sent an email, identifying an urgent need for Mr Rider or Mr Pavlyuk to visit Winsopia, to meet a deadline of the end of April to get the DB2 Connect code working. Shortly afterwards, Alexey Pavlyuk, a software developer at On Target, visited Winsopia between 5 and 7 June 2017, during which period he made commits to the Git repository. In April 2018 Mr Wilkinson was authorised by Mr Rockmann to visit Winsopia and shortly thereafter he sent an email detailing the results of his investigations into the DL/2 CSECTs he found. Mr Bendich of LzLabs visited Winsopia in March 2019, where he was able to find bugs in the SDM code and use this information for development of the SDM. The dismantling of the barriers intended to separate SDM development from the mainframe software was exacerbated by the movement of employees between LzLabs and Winsopia, including via LzLabs UK.

876. This inter-mingling of LzLabs developers with Winsopia gave direct access to the mainframe software that augmented the information obtained through the DR system. The DR system did not prove to be successful in supervising or regulating the requests raised and responses provided so as to enforce compliance with the terms of the ICA, as evidenced by the findings of breach. These findings include deficiencies in the scrubbing process introduced by the CPX tool, that did not achieve removal of all IBM CSECTs and other proprietary materials.

877. In conclusion on this issue, I am satisfied that LzLabs procured or induced Winsopia to commit acts that amounted to breaches of contract.

878. The next question is whether LzLabs knew, or was reckless as to, the essential facts which made the acts unlawful for the purpose of the required element of knowledge.

879. The primary rule of attribution is that a company must necessarily have attributed to it the state of mind of its directing organ under its constitution, i.e. the board of directors acting as such or for some purposes the general body of shareholders: *Meridian Global Funds Management Asia Ltd v Securities Commission* [1995] 2 AC 500 per Lord Hoffmann at p.506; *Bilta (UK) Limited v Nazir (No.2)* [2015] UKSC 23 per Lord Sumption at [67].

880. Mr Rockmann was the CEO or Chief Operating Officer of LzLabs from its inception and Chairman of the Board. He was aware that the terms of the ICA

prohibited the acts of reverse engineering and misuse of ICA Programs giving rise to the established breaches of contract on the part of Winsopia as follows.

881. Firstly, he signed the ICA, having reviewed the terms with the legal team, and initialled each page.

882. Secondly, although Mr Rockmann signed the ICA on behalf of Winsopia, thereby binding Winsopia but not LzLabs to the same, he acquired knowledge of its terms in his capacity as an officer of both Winsopia and LzLabs because Winsopia entered into the ICA under the direction and control of LzLabs.

883. Thirdly, in cross-examination, Mr Rockmann accepted that he was aware of the provisions of the ICA, including the restrictions contained in clause 4 of the agreement.

884. Fourthly, the Software Directive arguments, rejected by this court, would not be sufficient to displace knowledge of the wrongfulness of Winsopia's activities. As explained by Lord Leggatt in *Lifestyle Equities*, liability does not depend on whether the defendant knew that the act done by the primary wrongdoer was against the law; all that is required is knowledge of the essential facts which make the act unlawful. In this case, the essential facts are knowledge that the terms of the ICA prohibited the acts of reverse engineering and other misuse of ICA Programs that LzLabs directed Winsopia to carry out.

885. Fifthly, Mr Rockmann is not entitled to rely on any belief that the Software Directive would provide a defence to the ICA breaches because he conceded that he did not have an understanding of the Software Directive. Further, no legal advice pursuant to which he might have acted has been produced in these proceedings.

886. Although that would be sufficient to establish the requisite knowledge attributable to LzLabs, I find that Mr Cresswell also had knowledge of the essential facts which made the acts unlawful. He was CEO of LzLabs between 18 May 2015 and November 2020. Thereafter, his role was Executive Chairman, followed by Non-Executive Chairman. He was also a director at Winsopia between 2 December 2015 and 28 February 2022. As an officer of both LzLabs and Winsopia, he must have known the terms of the ICA, including the prohibitions on reverse engineering and other misuse of the ICA Programs. Indeed, in cross-examination he accepted this:

> "Q. So, at least at a high level, you appreciated that the ICA prohibited reverse-engineering?
>
> A. Yes, I think I mentioned that in my witness statement.
>
> Q. And you understood that it prohibited individuals outside of Winsopia being given access to ICA Programs or using them?
>
> A. Yes. There was -- absolutely, yes."

887. In any event, as stated by Lord Denning MR in *Emerald Construction Co Ltd v Lowthian* [1966] 1 WLR 691 at pp.700-701:

> "Even if they did not know the actual terms of the contract, but had the means of knowledge - which they deliberately disregarded - that would be enough. Like the man who turns a blind eye… For it is unlawful for a third person to procure a breach of contract knowingly, or recklessly, indifferent whether it is a breach or not."

888. The final required element is intention. In this case, that element is straightforward. LzLabs' intention was to develop and market the SDM as quickly as possible. That could not be done without access to the Winsopia mainframe and the licensed operating system. Access to such software put Winsopia in breach of the terms of its licence. Thus, the means to achieving LzLabs' aim was breach of the ICA.

LzLabs UK

889. The pleaded case is that LzLabs UK assisted LzLabs in procuring breaches of the ICA by Winsopia. It is recognised that assisting in a breach of contract can amount to inducement where the defendant joins in with the conduct of the primary contract-breaker so as to make him an accessory to the breaking of the contract: *Kawasaki* per Popplewell LJ at [23]. The difficulty in this case is that there is very little evidence as to the activity of LzLabs UK in connection with the established technical breaches.

890. LzLabs UK was incorporated in September 2015 as a wholly owned subsidiary of LzLabs to provide a UK-based LzLabs QA team. Mr Cresswell and Mr Rockmann were appointed as directors. Mr Cresswell explained that there were three teams at LzLabs UK: the sales team, headed up by Mr Gray; the marketing team headed by Dale Vecchio; and the QA team, headed up by Graeme Hollinger. Mr Hollinger reported to Mr Durand, who was based in France, and Mr Durand reported to Mr Cresswell.

891. The role of the QA team at LzLabs UK was to identify the aspects of the SDM that needed to be tested, write the appropriate tests, supervise their execution and track progress of the same. Initially, LzLabs UK was housed in the same office as Winsopia. Subsequently, LzLabs UK employees were permitted to visit Winsopia's premises. Although there was a separate office at Winsopia, consisting of meeting rooms which were set aside for LzLabs UK employees, accessed via a separate entrance, there was a common toilet facility that was shared by both companies, as was the common room and coffee station.

892. There is evidence that strict separation between LzLabs UK and Winsopia was not maintained. On 31 January 2017, in the course of a discussion regarding a response to DR2199 and Mr Bray's explanation that he did not know how to test the program in question, Mr Viebrock, then a QA engineer at LzLabs UK, stated:

> "May have one of the guys slide over and just do a cursory look (over someone's shoulder, of course)."

893. It is clear from this conversation that Mr Viebrock was aware that it was not permitted for the LzLabs UK employee to work on the Winsopia side but intended to circumvent the rules by looking over the Winsopia employee's shoulder.

894. This approach, whereby the distinction between QA staff and developers was not adhered to in practice, was referred to by Mr Bleach of LzLabs UK in his email dated 4 December 2019:

> "My understanding of the important - vital - divide between Development and QA is that we in QA never write code which becomes part of SDM. This is what enables me to visit Winsopia every week, enables me to talk them through any problematic DRs (no Legal representative present!), to look over their shoulders as the use z/OS."

895. This was echoed by Christian Wehrli of LzLabs, who stated in an e-mail dated 29 March 2019:

> "Unfortunately we can't have access to the Winsopia mainframe. And even accessing customer mainframes is a bit tricky. So far the only solution is on site education at customers (that's also how I learnt it BTW), sitting next to the customer and looking over the shoulder.
>
> For educational purposes it's possible to visit Winsopia and have somebody walk through it, you can't just use the mainframe."

896. The above exchanges undermine the defendants' case that there was compliance with the clean room procedures. However, it does not provide support for IBM's allegation that LzLabs UK induced any breach of contract by Winsopia. Direct communications between LzLabs UK and Winsopia, or looking over the shoulder of Winsopia's employees whilst they were working, do not without more amount to a breach of the ICA. There are references to LzLabs UK employees, such as Mr Althen and Mr Vitale, in the email exchanges concerning identified scrubbing failures but their role appears to have been concerned to stop the practice, rather than facilitating or intending any breach. I note that a number of the original allegations relating to third party access to the mainframe (including by LzLabs UK QA engineers) were abandoned by IBM or not supported by the factual/expert evidence.

897. As a result, there is insufficient evidence on which to make a finding that there was persuasion, encouragement or assistance by LzLabs UK which had any causative effect on Winsopia breaching the terms of the ICA. Thus, IBM has failed to establish the required elements of inducement or intention.

Claims against the directors

898. The wrongful procurement case against Mr Cresswell and Mr Rockmann is limited to their role as directors of Winsopia.

899. The basis of the case against the two directors is pleaded as follows:

> "33. … each of the Fourth and Fifth Defendants:
>
> 33.1. In their capacity as directors and executive officers of the Second Defendant, directed, instructed and/or requested the activities of the Second Defendant that amounted to or gave rise to breaches of the ICA or, alternatively, in their capacity as directors and executive officers of the Second Defendant, approved and/or ratified such directions, instructions and/or requests;
>
> 33.2. Knew that the activities of the Second Defendant being so directed, instructed and/or requested amounted or would give rise to breaches of the ICA or, alternatively, they were reckless in that regard …
>
> 34. In light of the aforesaid, it is inconceivable that the Second Defendant carried out the aforesaid activities without both the Fourth and Fifth Defendant intending the breaches of the ICA to which those activities amounted or gave rise (or, alternatively, being reckless as to whether those activities amounted to or gave rise to such breaches).
>
> 34A. As the Fourth and Fifth Defendants must have appreciated, likely consequences of the breaches of the ICA as identified in paragraphs 24 to 26 above, and as further particularised in paragraph 27 above, included that upon discovery of such activities by the Claimant (i) the ICA would be terminated, (ii) the Second Defendant would be severely hampered in its ability to carry on its business activities and to make any use of the IBM Mainframe that it purchased, and (iii) the reputation of the Second Defendant and its only shareholder would be harmed to such a degree that they would no longer be able to market or sell their core product, the SDM. In such circumstances, by engaging in the conduct identified in paragraphs 33 and 34 above, the Fourth and Fifth Defendants were in breach of their duties to the Second Defendant under section 172 of the Companies Act 2006.
>
> 34B. Accordingly, the Fourth and Fifth Defendants are each liable to the Claimant for such damage as was caused by the Second Defendant's breaches of the ICA."

900. It is first necessary to consider whether Mr Cresswell and/or Mr Rockmann caused Winsopia to act in breach of the ICA before considering the *Said v Butt* defence relied on by them.

901. As set out above, to the extent that Winsopia carried out activities that amounted to a breach of the ICA, it did so at the direction and under the control of LzLabs. Mr Cresswell and Mr Rockmann were directors at Winsopia. In those roles, they directed, instructed and encouraged the Winsopia employees to carry out the discovery work requested by LzLabs. Mr Rockmann controlled the work requested by LzLabs and carried out by Winsopia as explained above, in particular pursuant to the Services Agreement and through the DR process. Mr Cresswell was brought in to the project by Mr Moores to get a minimum viable product to market as quickly as possible. His protestation that he did not give any technical instructions to Winsopia personnel does not exonerate him from responsibility because he knew what work they were undertaking as part of his delivery plan for the minimum viable product. Examples of Mr Cresswell's immersion in the technical detail can be seen in his email dated 1 May 2015, stating that he understood the technical issues under discussion, his email discussions with LzLabs and Texas Wormhole dated 29 December 2015, and his receipt of the Lynch reports in November 2015 and April 2016. On that basis, I am satisfied that, in their positions as directors of Winsopia, they induced the acts that amounted to breaches of the ICA.

902. For the reasons set out above in relation to LzLabs, both Mr Rockmann and Mr Cresswell had the requisite knowledge of the essential facts, namely knowledge that the terms of the ICA prohibited the acts of reverse engineering and other misuse of ICA Programs that Winsopia carried out at LzLabs' direction. Further, Mr Rockmann and Mr Cresswell knowingly caused Winsopia to act in breach of the ICA with the intention of developing the SDM and getting it to market as quickly as possible.

903. It follows that, in their role as directors of Winsopia, Mr Cresswell and Mr Rockmann caused Winsopia to act in breach of contract.

904. That then makes it necessary to consider the defence based on the principle in *Said v Butt* [1920] 3 KB 497 per McCardie J at p.506:

> "if a servant acting bona fide within the scope of his authority procures or causes the breach of a contract between his employer and a third person, he does not thereby become liable to an action of tort at the suit of the person whose contract has thereby been broken."

905. The rationale behind this principle was examined by Lord Leggatt in *Lifestyle Equities*:

> "[54] I think that the rule stated in *Said v Butt* is sound … When parties make a contract, unless the contract is personal in nature, the general rule is that a party may employ agents to carry out its obligations. When the contracting party is a company, that is of course the only possible means of performance. If a company

> breaks a contract, that must be because one or more agents of the company have caused the breach. When an agent, acting as such, makes a contract, the normal understanding is that the agent assumes no liability towards the other contracting party. Only the principal does. Similarly, the normal understanding is that, if the agent causes the principal to break the contract, only the principal will incur liability to the other contracting party, and not the agent. This is, I think, a general norm or social understanding which the law should and does reflect.
>
> [55] It would be inconsistent with that understanding for the law of tort to make an agent who, acting within the scope of their authority, causes or procures a breach of contract by the principal liable to compensate the other contracting party for loss resulting from the breach. By the same token, to allow the injured party to recover damages from the agent would give them a free ride. That is because the same norm or understanding that, unless otherwise specifically agreed, only the contracting parties themselves will be liable in the event of a breach of the contract entails that, if a party wants a right of recourse against an agent of the other party, they must bargain for it."

906. The effect of the rule in *Said v Butt* is that a director of a company who caused his company to act in breach of contract cannot be found to have committed the tort of inducing a breach of a contract to which the company is party, provided that the director acted bona fide in the course of his duties as a director.

907. The concept of good faith was considered by David Richards J (as he then was), in *Lictor Anstalt v Mir Steel UK Ltd* [2011] EWHC 3310 (Ch) at [54]:

> "In this context good faith connotes the proper conduct of his duties and functions as a director".

908. Section 172(1) of the Companies Act 2006 requires the director of a company to act in the way that they consider, acting in good faith, will be most likely to promote the success of the company for the benefit of its members as a whole having regard to matters including: (a) the likely consequences of any decision in the long term, (b) the interests of the company's employees, (c) the need to foster the company's business relationships with suppliers, customers and others, (d) the impact of the company's operations on the community and the environment, (e) the desirability of the company maintaining a reputation for high standards of business conduct, and (f) the need to act fairly as between members of the company. The matters to which a director must have regard under section 172 do not impose absolute duties; they can arise in many different factual circumstances and in most cases, when the relevant factors are considered and weighed in the balance, they will produce a range of reasonable options open to the director.

909. Not every breach of contract will amount to bad faith or breach of a director's duties. In *Ridgeway Maritime Inc v Beulah Wings Ltd ("The Leon")* [1991] 2 Ll.Rep. 611, a director who wrote a letter on behalf of his company which he

knew would place the company in repudiatory breach was entitled to rely on *Said v Butt* by way of defence.

910. In *Antuzis v DJ Houghton Catching Services Ltd* [2019] EWHC 843 (QB) at [114] Lane J observed that it was the director's conduct and intention in relation to his duties towards the company, not towards third parties, that provided the focus of the enquiry to be undertaken pursuant to the rule in *Said v Butt*. On the facts of that case, the officers of the company were found to have acted in breach of their duties under section 172 because their exploitation of employees was manifestly not in the employees' interests and ruined the company's reputation in the eyes of the community. They had not acted bona fide because they knew that what the company was doing was unlawful. Thus they were not protected by the rule in *Said v Butt*.

911. As noted by Eyre J in an earlier judgment in this case, *IBM v Lzlabs & Others* [2022] EWHC 884 at [36], in considering whether a director was properly acting in accordance with the duties imposed by section 172, it is necessary to consider the circumstances of the particular case as a whole:

> "Not every instance of causing a company to breach a contract or a legal obligation will involve a director in breach of the section 172 duties nor will every such instance cause him or her to be characterised as acting in bad faith for the purposes of the rule in Said v Butt. The key will be whether the director was properly acting to promote the success of the company taking account of the matters to which he or she is required by section 172 to have regard. Those will include the motivation of the director and the nature of the duties said to be broken but in addition the nature of the obligations being broken by the company and the consequences of the company's breach can be relevant to the question of whether the director can properly have been said to have been acting in the interests of the company.

912. In *Northamber plc v Genee World Limited and others* [2024] EWCA Civ 428, in upholding a finding that directors were liable for inducing the company to act in breach of an injunction, exposing the company to fines for contempt of court, sequestration of its assets and reputational damage, Arnold LJ stated at [92]:

> "A director who causes his company not merely to breach its contract, but also to act in breach of an injunctions is plainly in breach of his duty under section 172. Even if not every breach of section 172 would suffice to deprive the director of protection, this is a serious breach of duty for the reasons given by the judge."

913. In *Northamber* Arnold LJ at [81] stressed the importance of putting to a director in cross-examination any allegation of bad faith or breach of duty as a director. It was put expressly to each of Mr Cresswell and Mr Rockmann in cross-examination that they were in breach of their duties as directors and were acting in bad faith so as to engage this issue. They both denied any such breach.

914. In my judgment, Mr Cresswell and Mr Rockmann did not act in bad faith or outside the scope of their authority as directors of Winsopia. Firstly, their motive was assistance in development of the SDM. To that intent, they simply ensured that Winsopia complied with the requests of LzLabs. Secondly, although the means by which such assistance was provided was through breach of the ICA, there was no dishonesty on the part of the directors vis-à-vis the company. Thirdly, despite the findings of systematic and widespread breach of the ICA, they did not involve egregious conduct on the part of Winsopia, such as exploitation of employees or contempt of court, so as to attract public opprobrium. Fourthly, the likely (indeed, inevitable) consequence of their actions was termination of the licence by IBM but that in itself would not be sufficient to establish bad faith (see *Ridgeway*).

915. A distinction can be drawn between the impact of the findings of breach on Winsopia and the impact of the findings of procurement of such breaches on LzLabs. LzLabs exploited the fruits of reverse engineering and misuse of IBM software for its own gain. The consequences for LzLabs are likely to include restrictions on its ability to carry out its business activities, in particular, marketing and sale of the SDM, and severe damage to its reputation. But the claim against Mr Cresswell and Mr Rockmann is confined to their role as directors of Winsopia. This was the result of a consent order dated 17 January 2023 to resolve an appeal in respect of a strike out application, as a result of which, references to the claim against Mr Cresswell and Mr Rockmann in their capacity as directors of LzLabs were deleted from the pleaded case. Given the limited ambit of Winsopia's business, and the purpose for which it was set up, namely, to provide discovery and testing support services to LzLabs, the directors' duties did not extend to promoting success of the development and marketing of the SDM.

916. For those reasons, Mr Cresswell and Mr Rockmann fall within the scope of rule in *Said v Butt* and are not liable for unlawfully procuring breach of contract on the part of Winsopia.

Mr Moores

917. I turn to consider the position of Mr Moores, who is the indirect beneficial owner of LzLabs, Winsopia and LzLabs UK and, along with his affiliated entities, the principal investor in the LzLabs Group. Because he was not an officer or employee in any of those companies, he does not benefit from the defence afforded to Messrs Cresswell and Rockmann under the principle in *Said v Butt.*

918. The pleaded case against Mr Moores is that he was ultimately in control of the corporate defendants, who followed his instructions in relation to the development of the SDM. It is alleged that Mr Moores directed and coordinated the development of the SDM, participating in the detail of its development, testing and marketing. In so doing, Mr Moores gave instructions and took steps that knowingly and intentionally induced and/or facilitated breaches of the ICA. The key allegations are:

i) Mr Moores instructed and/or authorised relaxations to the Codes of Conduct, permitting direct communication between employees of LzLabs and Winsopia, and access by LzLabs to the mainframe, thereby circumventing rules put in place to maintain operational separation between the defendants;

ii) Mr Moores instructed and/or authorised the regular movement of individuals between LzLabs, Winsopia, LzLabs UK and Texas Wormhole, thereby negating any real operational separation between the companies;

iii) Mr Moores instructed and/or authorised the transfer of ICA Program code from individuals working for Winsopia to individuals working for LzLabs and to others outside Winsopia's Enterprise; and

iv) Mr Moores directed and/or approved that Winsopia use machine readable portions of the IBM mainframe software or parts thereof on devices which were not Designated Machines as defined in the ICA, such as the Pizzabox and Brad's NUC.

919. The starting point is to recognise the power and control that Mr Moores exerted over the LzLabs Group, including Winsopia. He did not have a formal role or title within the LzLabs Group but, as the ultimate owner, he took an acute interest in what was happening with the SDM project, exerted pressure on the management, and obtained regular reports on development progress. This was understandable; he had invested substantial sums of money in LzLabs and wanted to see the results of such funding. From 2011 to 2018, he rented a flat in Zurich so that he could spend time at LzLabs. He estimated that he spent about 10 weeks of the year (across multiple trips) in Switzerland up until around 2018. Although Mr Moores was not a member of the LzLabs Board of Directors, or an officer of the company, he was appointed as a "board observer" of LzLabs. In that capacity, he attended almost all of LzLabs management board meetings, in which he played an active role and received reports on progress. Mr Cresswell, who was brought in to LzLabs and Winsopia by Mr Moores in 2015 to get a minimum viable product completed and delivered to market, reported directly to Mr Moores. He considered that he was ultimately accountable to Mr Moores and stated that, although he would not necessarily follow all directions from Mr Moores, probably he would not have done anything to which Mr Moores strenuously objected.

920. Mr Moores acknowledged that he could understand what the SDM did conceptually in quite a high level of detail. He visited Winsopia on a number of occasions and took a keen interest in the technical issues which arose, as illustrated by his email exchanges with Mr Cresswell on 1 May 2015 in which he referred to his involvement with the SDM development issues at a granular level.

921. Mr Moores used his power and control over LzLabs and Winsopia to direct and/or assist in the breaches of the ICA that have been established, primarily by dismantling the protective barriers put in place by the clean room processes. By March 2014 it was clear that he was unhappy with the rate of progress and

considered that the Codes of Conduct were hindering development, as set out in his email dated 25 March 2014:

> "Some things are failing & failing badly here. In general, the Chinese Wall is proving to be quite unworkable in its present manifestation. To wit:
>
> 1). We have not been able to connect the mainframe to our appliance for NJE (Network Job Entry) or for CICS Inter System Coupling. This has been going on for months now and, even worse, will recur from time to time, even after the current problems are resolved.
>
> While there are problems with the availability of consultants in the UK, the fatal flaw in establishing & maintaining connectivity is that this stuff is horribly arcane & fragile. There is no reasonable alternative to person-to-person communications.
>
> In general, the Chinese Wall is proving to be quite unworkable in its present manifestation. To wit:
>
> 2). The process of redaction & printing of listings to avoid a claim of copyright infringement is proving to be a complete bar to our discovering how APIs & communication protocols actually work.
>
> This just isn't working. To be clear, it isn't working poorly; rather it isn't working at all.
>
> So, as my long-time attorney from Houston has stated: Things can only be as good as they can be. The status quo is unworkable.
>
> So I suggest a modification to our Code that provides for direct communication between Winsopia & LzLabs for establishing and maintaining communication links. We also need to get unredacted listings, including assembly listings, of customer programs and traces, etc. Both these issues are narrow in scope and can be thoroughly documented, by 3rd parties and by maintaining a copy of transmittals.
>
> I propose that we send several people ASAP to the UK to fix the communication issues, including things that may or may not be working with the LzLabs appliances & hardware. Otherwise we need to shut the biz down and go home.
>
> While this isn't working, it can be fixed. I am deeply frustrated."

922. Mr Moores' level of concern was indicated by his veiled threat to shut down the project if the position did not improve. He suggested modifications to the Codes of Conduct, to permit direct communication between Winsopia and LzLabs, and

to get unredacted listings, including assembly listings, of customer programs and traces, to fix what he perceived to be the problem.

923. His impatience with progress was further reflected in his memorandum of 26 April 2014, in which he referred to the "Goat Rope 2014", a meeting attended by Winsopia, LzLabs and other software developers to share information about various projects, including the SDM, as disappointing. He was frank in cross-examination that he wanted developers at LzLabs to have direct access to a mainframe.

924. In April 2014 the Codes of Conduct were revised in accordance with Mr Moores' wishes, with the approval of Mr Rockmann. This was a significant departure from the principle of separation that formed the basis of the existing Codes of Conduct. Although Mr Moores demurred at the suggestion that he dictated the changes, the only plausible explanation for the timing and content of them is that they were in response to his concerns.

925. Mr Moores became concerned about progress again at the end of April/beginning of May 2015, when he asked for an update on progress and questioned whether the deadline of 1 June 2015 would be met. By email dated 28 May 2015, in response to delay caused by a DR response requiring legal review, Mr Moores called for further changes:

> "Lordy. Yet another clear example why we need Code of Conduct v2.0".

926. He clearly expected his call for changes to be implemented because the next day, he stated:

> "I hope that the new Code of Conduct is completed Monday because development progress is being substantially impacted. Continuing the current system won't work.
>
> I mentioned to a couple of folks that they should plan on going to the UK to work at Winsopia as early as this coming Friday."

927. Mr Moores confirmed that he was referring to LzLabs developers and that he anticipated that a change would be made to the existing version of the Codes of Conduct to permit them direct mainframe access. As anticipated, the revised Codes of Conduct were issued shortly afterwards, on 10 June 2015, pursuant to which LzLabs developers were permitted to have direct access to Winsopia's mainframe. Immediate advantage was taken of the relaxed procedures, as demonstrated by Mr Moores' email dated 19 June 2015, in which he referred to a planned visit to Winsopia with Mr Jaeger to discuss the new "Load Module compiler proposal" and suggested that Mr Spencer at Texas Wormhole could participate in the analysis.

928. In the light of that evidence, I accept IBM's submission that during 2014 and 2015: (i) Mr Moores concluded that the development of the SDM was impossible without direct access to the Winsopia mainframe and changes to the clean room procedures; (ii) Mr Moores applied pressure to Mr Rockmann and

(from May 2015) to Mr Cresswell to make significant changes to the Codes of Conduct; and (iii) Mr Rockmann and Mr Cresswell procured and/or implemented changes to the Codes of Conduct in response to the pressure which Mr Moores had applied.

929. It is disputed by the defendants that the relaxation of the clean room processes had any direct link with the breaches of the ICA. But it can be seen that it did have such a direct causal effect, when the ICA breaches are examined against the changes directed and approved by Mr Moores.

i) Following the revisions to the Codes of Conduct in 2014 to allow Winsopia to provide LzLabs with assembler listings, compiler listings and other information using debugging tools, such assembly and compiler listings were used to reverse engineer ICA Program components (Items 6 to 12, 19 to 30 and 46); and traces and other debugging tools were used to reverse engineer ICA Program components (Items 5 and 13 to 18).

ii) Following the revisions to the Codes of Conduct in 2014, permitting communications between Winsopia and LzLabs outside the DR process, as illustrated by Mr Taylor's email of 22 June 2014, compiler listings were used to expand the CICS Control Blocks Document (Items 3 and 4).

iii) Following the revisions to the Codes of Conduct in 2015 to allow LzLabs to have access to the Winsopia mainframe and direct communications with Winsopia employees, Mr Moores visited Winsopia with Mr Jaeger to discuss that new LMD/LMC project which was developed, directed by LzLabs (Item 2); and Mr Taylor of LzLabs was afforded direct access to the Winsopia mainframe (Item 49).

iv) Mr Moores actively participated in forwarding the disassembled code carried out by Mr Lynch in 2014 to LzLabs (Item 1).

930. In conclusion on this issue, I am satisfied that Mr Moores persuaded, encouraged and assisted Winsopia to commit acts that amounted to breaches of contract.

931. Turning to the question of knowledge, this has to be considered against the background of the Neon litigation. Mr Moores acknowledged that reverse engineering was a complaint made by IBM Corp in the Neon litigation as amounting to breach of the Neon ICA:

> "Q. Well, why don't you tell me now, Mr Moores. Are you -- is your -- you were aware that there was disassembly in the Neon litigation?
>
> A. At some point I was.
>
> Q. And you were aware that IBM was making a complaint that was in breach of the ICA?

A. At some point I was.

Q. And that was before Winsopia entered into its ICA, wasn't it?

A. Yes."

932. Although Mr Moores stated firmly that he did not read the ICA and did not know the precise terms and conditions of the same, he agreed that the ICA was a standard term agreement used by IBM and that he was aware that the ICA prohibited decompilation. In any event, as set forth in the discussion on knowledge in relation to Mr Rockmann and Mr Cresswell, to the extent that Mr Moores elected to disregard the ICA terms and conditions, which were available to him, that would be sufficient to constitute knowledge of the same.

933. It was suggested by Mr Moores that he thought that the Software Directive allowed Winsopia to carry out disassembly and decompilation. However, when asked about his understanding of the Software Directive, he was unable to provide any details and no legal advice has been produced which might support his declared understanding.

934. As set out above in relation to LzLabs, the required intention is satisfied. Mr Moores' intention was to develop and market the SDM as quickly as possible. That could not be done without access to the Winsopia mainframe and the licensed operating system:

> "Q. And you knew that it was impossible to develop the SDM without breaching Winsopia's ICA?
>
> A. It certainly would have been harder without -- first of all, let me back up. You suggest that I -- you're asking do I know that IBM -- or that Lz breached the ICA? I don't know that. And your question suggests that -- makes me assume that I knew a breach had occurred. That's not correct, and I'll be glad to answer a shorter question, but I'm not going to answer one that assumes a breach of the ICA.
>
> Q. Well, you say it certainly would have been harder without. Can you finish your answer?
>
> A. It would have been harder without some access to -- to a mainframe without -- even without heroic efforts. It was -- it was not something that -- that I ever wanted to do, and I always, as I said earlier, thought that the world's longest running, most expensive, most exhaustive Clean Room process ever needed to take place to achieve interoperability with customer programs."

935. Although he disputed that access to the mainframe software put Winsopia in breach of the terms of its licence, I have found on the facts of this case that it did so do. Thus, the means to achieving Mr Moores' aim of developing the SDM was breach of the ICA.

Summary on unlawful procurement

936. For the above reasons I find that:

i) LzLabs and Mr Moores unlawfully procured breaches of the ICA by Winsopia;

ii) the claim against LzLabs UK fails;

iii) Mr Rockmann and Mr Cresswell are entitled to rely on the principle in *Said v Butt* by way of defence and therefore are not liable for the tort of unlawfully procuring breaches of the ICA by Winsopia.

Section VII - Unlawful means conspiracy

937. IBM's case is that the defendants combined with each other to achieve the common end of developing the SDM using unlawful means, namely, breaches of the ICA by Winsopia and procurement of such breaches by the other defendants. It is said that the combination was entered into with the intention to injure the claimant, by developing a competitor product which would damage the claimant's mainframe business, the defendants undertook concerted action consequent upon the combination, knowing that the breaches and procurement of such breaches were unlawful, and the claimant suffered damage as a result of the conspiracy.

938. The defendants' position is that none of the necessary elements of the tort have been established and IBM's case fails on both the evidence and the law. The case is entirely dependent on the breach and procurement allegations but fails with those allegations. There was no combination or agreement to develop the SDM using material and information extracted from the IBM mainframe software in breach of the ICA. IBM has failed to establish the requisite mental element to found its conspiracy claim based on the procurement case. IBM has failed to establish any causal link between the alleged unlawful acts and the loss alleged to have been caused. There was no intention to injure IBM.

939. Given my findings on breach of the ICA and unlawful procurement above, this additional claim is of limited relevance to the outcome of the case. However, as it has been addressed by the parties, I consider it briefly below.

940. The pleaded case is as follows:

> 44A. Further, the Defendants entered into an unlawful means conspiracy to commit breaches of the ICA and/or procure breaches of the ICA. In particular:
>
> 44A.1 The Defendants (and each of them) combined with the other Defendants (and each of them) to achieve the common end of developing the SDM using unlawful means. In summary (and as further particularised above):
>
> 44A.1.1 The First Defendant developed and marketed the SDM, using information and material that had been passed to

it by the Second Defendant in breach of the ICA, and procured breaches of the ICA.

44A.1.2. The Second Defendant breached the ICA in order to assist the First Defendant in the development of the SDM.

44A.1.3 The Third Defendant developed and marketed, and/or assisted the First Defendant in the development and marketing of, the SDM to third parties, knowing it had been and/or would be developed as a result of the Second Defendant's breaches of the ICA, and procured breaches of the ICA.

44A.1.4 The Fourth and Fifth Defendants were responsible for the operations of each of the corporate Defendants as set out above, and procured the breaches of the ICA.

44A.1.5 As the (ultimate) beneficial owner and controller of each of the Defendants, it is inferred that the Sixth Defendant took the initial decision to develop the SDM via an unlawful scheme requiring breaches of the ICA by the Second Defendant, and was principally responsible for the development of the scheme to achieve these ends. In any event, he was principally responsible for the execution of this scheme, as the ultimate decision-maker throughout the process of developing the SDM. The Sixth Defendant also procured the breaches of the ICA as set out above.

44A.2 The said unlawful means therefore comprised:

44A.2A Breaches of clauses 4.1.2(b), 4.1.3(a) and 4.13(b) of the ICA by the Second Defendant, as set out at paragraphs 23-28 above; and

44A.2B Procurement of the said breaches by the First and Third to Sixth Defendants, which was itself tortious, as set out at paragraphs 29-34F above. Without the said breaches the SDM could not have been developed, for the reasons set out above at paragraph 27.

44A.3 The said combination was entered into with the intent to injure the Claimant, by developing a competitor product which would damage the Claimant's mainframe business.

44A.4. The Defendants undertook concerted action consequent upon the combination, as set out above.

44A.5 The Claimant suffered damage as a result of the conspiracy, as further set out below at paragraph 46.2.

> 44B. To the extent relevant, it is averred that the Defendants knew that the breaches and/or procurement of breaches set out above were unlawful.
>
> 44C. In the premises the Defendants (and each of them) is liable for the tort of unlawful means conspiracy for such damage as was caused by the Second Defendant's breaches of the ICA."

Applicable legal principles

941. The tort of unlawful means conspiracy requires: (i) a combination between the defendant and others; (ii) an intention to injure the claimant; (iii) unlawful acts carried out pursuant to the combination as a means of injury; and (iv) causation of loss to the claimant: *Kuwait Oil Tanker v Al Bader* [2000] 2 All ER (Comm) 271 (CA) at [108] & [132]; *JSC BTA Bank v Khrapunov* [2018] UKSC 19 at [8]; *The Racing Partnership Ltd v Sports Information Services* [2020] EWCA Civ 1300 per Arnold LJ at [104].

942. The first necessary element is a combination, arrangement or understanding between two or more people. It is not necessary to show an express agreement, whether formal or informal. It is sufficient if two or more persons combine with a common intention, or, in other words, that they deliberately combine, albeit tacitly, to achieve a common end. It is not necessary for the conspirators all to join the conspiracy at the same time, but the parties must be sufficiently aware of the surrounding circumstances and share the same object for it properly to be said that they were acting in concert at the time of the relevant acts: *Kuwait Oil Tanker* at [111]. A conspirator's participation in the combination may be active or passive: *Lakatamia Shipping v Su* [2021] EWHC 1907 (Comm) per Bryan J at [95]-[98].

943. For the reasons set out above, I find that this element is satisfied in respect of LzLabs, Winsopia and Mr Moores. LzLabs developed and marketed the SDM using information gleaned from the ICA Programs and IBM proprietary material that had been requested from, and passed to it by, Winsopia. Winsopia acted in breach of the ICA to assist LzLabs as directed in the development of the SDM. Mr Moores used his funding, power and control over LzLabs and Winsopia in furtherance of these activities to develop the SDM.

944. The second necessary element is intention to injure: *OBG v Allan* (above) per Lord Hoffmann at [62] and Lord Nicholls at [164]-[166]. In that regard, it is necessary to establish an intention to injure the claimant but not a predominant intention or purpose to do so: *Kuwait Oil Tanker* at [118]. It is sufficient that the defendant intends to advance their or the company's economic interests at the expense of the claimant's, where the gain and corresponding loss are inseparably linked: see *FM Capital Partners v Marino* [2018] EWHC 1768 (Comm) per Cockerill J at [94].

945. I find that this element is satisfied in respect of LzLabs, Winsopia and Mr Moores. For the reasons set out above, LzLabs, Winsopia and Mr Moores intended to develop the SDM as a competitor product at the expense of IBM's mainframe business through breach of the ICA.

946. The third necessary element is unlawful acts carried out pursuant to the combination as a means of injury. Although there may be debate as to the scope of "unlawful means" required, it is common ground that it covers civil wrongs such as torts and breaches of contract: *OBG v Allan* (above) Lord Nicholls at [150]-[151].

947. I have already found that Winsopia was in breach of contract, and that LzLabs and Mr Moores unlawfully procured such breaches, thereby satisfying this required element.

Knowledge of unlawfulness

948. There is a dispute between the parties as to whether it is necessary for a claimant to establish that the defendants had knowledge of the unlawfulness of the means to be liable for unlawful means conspiracy.

949. This thorny issue was considered by the Court of Appeal in *Racing Partnership* (above). Having carefully analysed the conflicting authorities in detail at [106]-[138], Arnold LJ stated at [139]:

> "Accordingly, the conclusion I draw from the authorities is that, having regard both to the general statements of the ingredients of the tort which do not include any requirement of knowledge of unlawfulness, and to the persuasive force, even if not binding status, of *Churchill v Walton* and *Belmont v Williams*, knowledge of the unlawfulness of the means employed is not required for unlawful means conspiracy."

950. It was acknowledged by Arnold LJ that there were good arguments in support of the contrary view that knowledge of unlawfulness should be required but he was not persuaded by them. In response to the argument that the tort of conspiracy to injure by unlawful means would otherwise be a tort of very broad reach, in particular because a predominant intention to injure the claimant is not required, and it is sufficient that the defendant intends to advance his own economic interests at the expense of those of the claimant, the Learned Judge indicated that he was sympathetic to the proposition that that the tort should be kept within bounds but considered that it did not necessarily follow that knowledge of unlawfulness was the means of imposing the limit. He rejected the argument that the tort of inducing breach of contract would become redundant, on the basis that almost all cases of inducing breach of contract could be reframed as conspiracy to injure by unlawful means, because *OBG v Allan* had established that these were separate torts with distinct elements. Finally, he considered the argument that knowledge of unlawfulness should be required if the unlawful means consists of an infringement of private law rights, even if it is not required where it consists of a crime or contravention of a regulatory provision imposed for public benefit:

> "[143] This receives a degree of support from the observations of Lord Sumption and Lord Lloyd-Jones JJSC in Ablyazov at para 15 to the effect that breaches of private law rights raise different considerations to crimes. I am unable to see why this

> should make a difference so far as the present issue is concerned, however. As noted above, both private law and the criminal law make some acts unlawful without proof of any mental element whatever. To my mind, it would be more logical to say that knowledge of the unlawfulness should be required where the unlawfulness of the means requires knowledge."

951. Arnold LJ's conclusion at [144] was clear:

> "For the reasons given above, I would respectfully conclude that the judge was wrong to conclude that knowledge that the means is unlawful is required in order for the tort of conspiring to injure by unlawful means to be established where the means is an infringement of a private right."

952. Phillips LJ agreed with Arnold LJ's conclusion that knowledge of the unlawfulness was unnecessary. Lewison LJ came to a different conclusion on this difficult issue at [265] but that was in a dissenting judgment.

953. A degree of support for further consideration (in a suitable case) of the argument that knowledge of the unlawfulness might be required where the unlawfulness of the means requires such knowledge can be found in the judgment of Phillips LJ in *Racing Partnership* at [171]:

> "I agree with Arnold LJ's conclusion at para 139, based on his analysis of the authorities, that knowledge of the unlawfulness of the means employed is not required for unlawful means conspiracy. The point was directly in issue and so decided by this court in *Belmont Finance Corpn v Williams Furniture Ltd (No 2)* [1980] 1 All ER 393, a decision that was not referred to by Toulson LJ in his obiter dictum in *Meretz Investments NV v ACP Ltd* [2008] Ch 244. The interplay between unlawful means conspiracy and inducing breach of contract (where knowledge of an unlawful breach of contract is an essential element) may merit further examination in a suitable case, but I am not convinced that many cases in which a defendant induces a breach of contract, but without knowing that he is doing so, would be capable of being reformulated as an unlawful means conspiracy."

954. Such an argument would not make any difference to the result in this case. Even if knowledge of unlawfulness were required in respect of breach of the ICA by Winsopia and/or unlawful procurement of breach by LzLabs and Mr Moores, I have found that Mr Moores, and Mr Cresswell and Mr Rockmann (officers of LzLabs and Winsopia) knew, or deliberately turned a blind eye to, the essential facts which made the acts unlawful.

955. As regards the case against Mr Cresswell and Mr Rockmann, IBM correctly submits that a company and its directors may be co-conspirators; the rule in *Said v Butt* does not apply in the conspiracy context. In *Lifestyle Equities* (above) at [63] Lord Leggatt stated:

> "The rule in *Said v Butt* does not apply to civil wrongs which do not depend on any contract or voluntary arrangement between the parties and where liability arises even if they are complete strangers to one another. "

956. However, the pleaded case by IBM is that the unlawful means relied on are: (a) the pleaded breaches of the ICA by Winsopia; and (b) the pleaded unlawful procurement of such breaches by the other defendants. The pleaded unlawful procurement case against Mr Cresswell and Mr Rockmann is limited to their respective roles as directors and officers of Winsopia. For the reasons set out above, that claim against Mr Cresswell and Mr Rockmann for unlawful procurement fails. Therefore, there is no established unlawful means that could give rise to liability for the purpose of the conspiracy claim.

957. This was recognised by David Richards J (as he then was) in *Lictor v MIR Steel* (above) at [68]:

> "The essence of the rule is that agents are not to be liable for procuring their principal to act in breach of contract, provided they acted in good faith in the course of their agency, and it should make no difference whether the claim is made for inducing a breach of contract or for an unlawful means conspiracy. The High Court of Australia so held in O'Brien v Dawson (1942) 66 CLR 18 and, in my judgment, the position is the same in English law."

958. Likewise, for different reasons as set out above, the claim for unlawful procurement against LzLabs UK fails. As a result, there is no established unlawful means that could give rise to liability for the purpose of the conspiracy claim against LzLabs UK.

959. The fourth necessary element is causation of loss to the claimant. I am satisfied that at least some loss must have been caused to IBM, even if limited to additional management time or lost marketing opportunities. IBM has pleaded loss and damage to its mainframe business but this part of the trial is limited to issues of liability and therefore quantum has not been fully pleaded or investigated.

Summary on unlawful means conspiracy

960. For the above reasons I find that:

i) LzLabs, Winsopia and Mr Moores are liable for the tort of unlawful means conspiracy;

ii) the claims against LzLabs UK, Mr Cresswell and Mr Rockmann fail.

Section VIII – Audit and Termination

961. IBM's case is that by notice dated 24 February 2021, it terminated the ICA and associated Agreements on grounds that: (a) Winsopia refused to comply with

its audit obligations in response to IBM's request under the ICA; and (b) Winsopia was in breach of clause 4.1 of the ICA. Further or alternatively, the breaches of the ICA were repudiatory and/or by means of such breaches Winsopia evinced an intention not to be bound by the Agreements and thereby renunciated the same. If the notice dated 24 February 2021 were ineffective to terminate the Agreements, IBM would rely on contractual or common law termination by service of its pleaded case.

962. The defendants' case is that the letter dated 24 February 2021 was an invalid letter of termination on the grounds that: (a) the audit request was not made on reasonable notice and failed to offer a reasonable period for Winsopia to respond; (b) the audit request was not made in good faith but was intended to elicit information about LzLabs for purposes extraneous to the ICA and/or for the collateral purpose of supplying such information to IBM Corp; and (c) no confidentiality undertakings were offered by IBM. Accordingly, the letter amounted to repudiation of the ICA and other Agreements, which Winsopia accepted by its letters dated 29 July 2024.

Validity of audit request

963. During 2020, Mr Anzani of IBM Corp held initial discussions with Mr Cresswell about a future partnership between IBM Corp and LzLabs. The question of a potential intellectual property infringement on the part of Winsopia was first raised by Mr Anzani in a letter dated 24 November 2020.

964. Shortly thereafter, by letter dated 3 December 2020, Mr Anzani of IBM Corp wrote to Winsopia in the following terms:

> "… I write to inform Winsopia that IBM will be conducting an audit of Winsopia's compliance with the terms of the IBM-Winsopia Agreements.
>
> Before commencing the in-person portion of the audit, IBM therefore seeks the following preliminary information from Winsopia:
>
> - A list of all IBM ICA Programs and other IBM Programs (collectively, the "IBM Software) used by Winsopia for any purpose in the past 36 months;
>
> - A list of all Designated Machines pursuant to the IBM-Winsopia Agreements;
>
> - A list of all machines on which any copy of any IBM Software, is or has been stored or executed in the past 36 months, including their location, platform (e.g ., z/OS, z/VM, z/Linux, z/VSE, Windows, or Linux), purpose(s) for which they are or were used, and a list of all the third party software installed on, running on, or accessible through such machines in the past 36 months;

- A system diagram showing how all the machines identified in the prior two bullets are interconnected, and have been connected in the past 36 months;

- For any z/OS machines identified above, provide the results of the following z/OS DISPLAY commands: …

- A listing of all persons and entities that have used, analyzed, or otherwise accessed the IBM Software in the past 36 months, as well as their locations and employer during that period;

- Confirmation that no employee or agent of LzLabs has used the IBM Software;

- Confirmation that no employee or agent of LzLabs or any of its affiliates has been provided with any information discerned from operating or testing the IBM Software;

- Confirmation that the IBM Software was not used in connection with the development of LzLabs' Software Defined Mainframe offering;

- A list of all uses of the IBM HLASM Toolkit Disassembler, the persons using it, and the programs on which it was used, in the past 36 months;

- An identification of any attempt to disassemble or reverse engineer any IBM Software, in the past 36 months. The response should include an identification of each use of the IBM Software licensed to Winsopia to attempt to develop software to replace IBM Software, including to attempt to develop software-defined mainframe software to replace IBM Software. The response should also include identification of any IBM Software that was executed or run in an LzLabs SDM or other x86 environment, e.g., through the use of an emulator or other facility to translate the software into x86 instructions;

- An identification of any affiliates of Winsopia, at present or in the past 36 months;

- All SMF data records in Winsopia's possession; Certification Requirements …

Please provide the above information no later than thirty days after receipt of this letter. Once a satisfactory response is received, we will contact you to discuss scheduling and procedures for the in-person phase of the audit."

965. Winsopia replied by letter dated 23 December 2020, refusing to supply the information requested and making the following points:

i) There was no prior notice of the audit and the letter asked for a considerable amount of information to be provided within 30 days during the COVID global pandemic and over a holiday period.

ii) Winsopia's relationship was with IBM and there was no obligation to provide any information to IBM Corp.

iii) The letter did not identify any contractual entitlement to a number of the categories of information sought, which included confidential information and personal data disclosure.

iv) Mr Anzani's earlier letter had suggested possible infringement of IBM's intellectual property rights but provided no basis for such allegations.

966. By letter dated 12 January 2021 Mr Wallin of IBM wrote to Winsopia, disputing the suggestion that IBM Corp could not request the information but repeating the request in his capacity as an officer of IBM. He gave notice that Winsopia was in material breach of its obligations under the ICA and associated agreements and invited Winsopia to cure that breach by providing the requested information in response to the audit demand within 30 days. IBM gave notice of its intent to terminate the agreements in the absence of such compliance.

967. Mr Rockmann of Winsopia replied by letter dated 8 February 2021, making the following points:

i) Winsopia was not in breach of any obligation; its contractual obligations were to IBM and not to IBM Corp.

ii) As Winsopia was not in breach, Mr Wallin's letter did not constitute valid notice under the ICA, and IBM had no right to terminate Winsopia's licences.

iii) The ICA required IBM to allow Winsopia a reasonable opportunity to comply before alleging breach, and to attempt in good faith to resolve all disputes, disagreements or claims but this had not been done.

iv) Before any confidential information would be exchanged, the ICA required a separate signed confidentiality agreement between Winsopia and IBM.

v) The audit request did not give sufficient details of the provisions in the ICA and associated agreements in respect of which IBM sought information and materials to provide auditable verification of Winsopia's compliance.

vi) The 30-day time frame for compliance was not reasonable and would cause disruption to Winsopia's business, especially in the context of the pandemic conditions.

vii) Winsopia expressed concern that the compliance verification provisions of the ICA were being invoked by IBM in a wide-ranging way, not merely to verify Winsopia's compliance, but also for purposes of IBM's commercial advantage, which included anti-competitively trying to gain indirectly non-public and confidential information about LzLabs, its intellectual property and its SDM product.

viii) Winsopia was not refusing to provide information or materials to which IBM was contractually entitled but required further details as to the purpose for which the information was sought and the contractual provisions to which the questions related.

968. By letter dated 24 February 2021, IBM notified Winsopia that it was terminating the ICA and associated agreements, on the grounds that: (a) Winsopia's refusal to provide the requested information in response to IBM's audit demand; and (b) use of the licensed IBM software outside the Winsopia Enterprise, were breaches of the ICA and associated agreements, which breaches Winsopia had failed to cure.

969. By letter dated 1 March 2021, Winsopia disputed the validity of the purported termination and sought to affirm the ICA and associated agreements.

970. By letters dated 29 July 2024, Winsopia purported to terminate the ICA and other licence agreements as from 31 August 2024.

971. The relevant contractual audit provisions were:

> Clause 4.4.1
>
> "IBM's right to verify the Customer's usage data and other information affecting the calculation of charges also includes the right to verify the Customer's compliance with other terms of this Agreement (including applicable Attachments and Transaction Documents) relating to the Customer's use of ICA Programs at all sites and for all environments in which the Customer installs or uses ICA Programs for any purpose. IBM may use an independent auditor to assist with such verification, provided IBM has a written confidentiality agreement in place with such auditor."
>
> Clause 4.4.2
>
> "The Customer agrees to create, retain, and provide to IBM and its auditors written records, system tools outputs, and other system information sufficient to provide auditable verification that the Customer's installation and use of ICA Programs complies with the Agreement terms, including IBM's applicable licensing and pricing terms. IBM will notify the Customer in writing if any such verification indicates that the Customer is not in compliance with Agreement terms. The rights and obligations in this section remain in effect during the period any ICA

> Programs are licenced to the Customer and for two years thereafter."

972. Clause 1.11.4 provided that each party would allow the other a reasonable opportunity to comply before claiming that the other had not met its obligations under the ICA; further, that the parties would attempt in good faith to resolve all disputes, disagreements, or claims between the parties relating to the ICA.

973. Clause 1.11.5 provided for the exchange of any confidential information to be made under a separate, signed confidentiality agreement.

974. The starting point is that clauses 4.4.1 and 4.4.2 gave IBM a contractual right to verify Winsopia's compliance with the ICA and associated agreements.

975. The audit request set out in IBM's letter dated 12 January 2021 was a valid request pursuant to clause 4.4.1 of the ICA. It was sent on IBM headed notepaper by Mr Wallin of IBM, who had at least ostensible authority to send it, and who reiterated the requests for information in connection with the audit as set out in Mr Anzani's earlier letter of 3 December 2020. This did not simply repeat the request by IBM Corp but made the same requests on behalf of IBM.

976. The requests for information and the period allowed for compliance were reasonable. Although the audit request sought a substantial amount of documentation, much, if not all, of the information would have been available readily in electronic form by straightforward searches, which could have been carried out remotely, as Mr Rockmann accepted in cross-examination, and were therefore unaffected by the pandemic. Despite that, Winsopia failed to provide any such information, or explain why any particular categories of information might be difficult to provide. Even if Winsopia had legitimate reasons for its inability to provide all requested information within 30 days, it failed to give any indication as to when it would start to comply with the request.

977. It was not incumbent on IBM to explain or justify the purpose for which it required the information. It was sufficient that it identified the contractual provisions pursuant to which it was entitled to carry out the audit and to receive the information requested. The ICA did not contain any express or implied obligations of good faith in respect of the contractual audit entitlement. Clause 1.11.4 of the ICA contained an obligation of good faith on both parties to attempt to resolve all disputes, disagreements or claims between the parties but this did not override the express provisions entitling IBM to carry out an audit.

978. Clause 1.11.5 provided for any exchange of confidential information to be subject to a separate, signed confidentiality agreement but that was subject to the proviso that confidential information exchanged in connection with any Product or Service under the ICA was covered by the applicable confidentiality agreement incorporated in and subject to the terms of the ICA. For the purposes of the audit request, a separate confidentiality agreement was only required in circumstances where an independent, third party carried out the audit. In any event, much of the information sought by IBM was not confidential. In particular, details of the software and hardware in use were not confidential to Winsopia, nor was the request for Winsopia to confirm that LzLabs had not used

the IBM software, or information derived from testing the IBM software, in development of the SDM.

979. It follows from the above, that the audit request by IBM was valid and Winsopia's refusal to comply with it, or supply any of the information requested, amounted to a breach of the ICA.

Validity of termination

980. Clause 1.12.2 provided that either party might terminate the ICA if the other did not comply with any of its terms, provided the one who was not complying was given written notice and reasonable time to comply.

981. As set out above, Winsopia failed to comply with IBM's valid audit request and therefore was in breach of the ICA.

982. Further, clause 4.5.3 provided that IBM might terminate Winsopia's licence if Winsopia failed to comply with the licence terms.

983. Notwithstanding any arguments as to the validity of the audit request, Winsopia was in breach of the ICA licence terms as set out in my findings on the technical breaches summarised in paragraphs [838] to [844] above.

984. Winsopia failed to respond to the requests for it to provide confirmation that: (i) no employee or agent of LzLabs had used the IBM Software; (ii) no employee or agent of LzLabs or any of its affiliates has been provided with any information discerned from operating or testing the IBM Software; and (iii) the IBM Software was not used in connection with the development of the SDM. In those circumstances, IBM had no obligation to allow Winsopia any further period to cure its breaches or resolve any dispute; the technical breaches were not capable of rectification.

985. In those circumstances, IBM was entitled to rely on Winsopia's breach of clause 4.1 of the ICA as justification for termination of the same pursuant to clause 1.12.2 and/or termination of the Winsopia licence pursuant to clause 4.5.3.

986. Alternatively, IBM was entitled to treat the technical breaches as repudiatory and terminate the ICA and associated agreements at common law.

987. For the above reasons, I find that IBM validly terminated the ICA and/or Winsopia's licence for the ICA Programs.

Section IX - Limitation

988. The material dates are as follows:

i) Winsopia and IBM entered into the ICA on 15 August 2013.

ii) The SDM was marketed from March 2016.

iii) The technical breaches span the period August 2013 through to September 2021.

iv) IBM terminated the ICA and associated agreements by letter dated 24 February 2021.

v) These proceedings were started on 21 September 2021 against LzLabs, Winsopia, LzLabs UK, Mr Cresswell and Mr Rockmann in respect of the ICA breaches and unlawful procurement of breach claims ("the Original Claims").

vi) The unlawful means conspiracy claim against all defendants and the unlawful procurement of breach claim against Mr Moores ("the New Claims") were made on 29 November 2023. Those amendments were made on the basis that IBM would not seek to rely on the 'relation-back' rule and limited its claims to those that were not time-barred as at 29 November 2023.

989. The defendants' pleaded case raises the following issues of limitation:

i) Clause 1.11.4 of the ICA provides for a two-year contractual limitation period, said to be enforceable by Winsopia to bar all (or most) claims against it and the other defendants.

ii) The Original Claims in both contract and tort are said to relate to causes of action accruing prior to 21 September 2015 and, therefore, are statute-barred by reason of sections 5 and 2 of the Limitation Act 1980.

iii) The New Claims in tort are said to relate to causes of action accruing prior to 29 November 2017 and, therefore, are statute-barred by reason of section 2 of the Limitation Act 1980.

990. IBM disputes that the claims are time-barred and raises a plea of deliberate or dishonest concealment:

i) It is said that on a proper construction of clause 1.11.4 of the ICA, it applies only to the parties who entered into it, namely, IBM and Winsopia; and/or it does not apply in respect of claims where the relevant cause of action involves dishonest or deliberate concealment.

ii) It is alleged that acts by the defendants constituted deliberate concealment within the meaning of section 32 of the Limitation Act 1980.

iii) IBM's case is that it did not discover, and could not with reasonable diligence have discovered, the concealment prior to 25 August 2020, less than two years prior to the issue of proceedings in respect of the Original Claims and less than six years prior to the New Claims.

Contractual limitation

991. The material part of clause 1.11.4 provides:

> "Unless otherwise required by applicable law without the possibility of contractual waiver or limitation, i) neither party

> will bring a legal action, regardless of form, arising out of or related to this Agreement or any transaction under it more than two years after the cause of action arose; and ii) after such time limit, any legal action arising out of this Agreement or any transaction under it and all respective rights related to any such action lapse."

992. The defendants submit that clause 1.11.4 bars any cause of action which arose more than two years prior to the issue of proceedings (21 September 2019 in respect of the Original Claims and 29 November 2021 in respect of the New Claims). The express wording of the clause is wide enough to cover all claims by IBM against Winsopia for breach of contract and unlawful means conspiracy. Further, on a proper construction of the provision, it applies to all legal action by IBM arising out of or related to the ICA, against Winsopia and the other defendants.

993. IBM submits that, save for Winsopia, the defendants are not entitled to rely on clause 1.11.4 of the ICA because they were not parties to it; there is no privity of contract. Further, on a proper construction of clause 1.11.4, it does not apply where the wrongdoing has been dishonestly or deliberately concealed by the party in breach.

994. The starting point is that the general principles of contractual interpretation, set out above in the ICA section of the Judgment apply, save that, where there is material ambiguity in a contractual provision which purports to restrict or exclude the parties' common law rights, the court will require clear words before finding that a party has given up its valuable rights: *Triple Point Technology Inc v PTT Public Co Ltd* [2021] UKSC 29 per Lord Leggatt at [110]-[111].

995. I accept the defendants' submission that, as against Winsopia, the express words of clause 1.11.4 are clear in precluding IBM from bringing a legal action more than two years after the cause of action arose. The use of the words *"arising out of or related to this Agreement"* are sufficiently wide to include the claims for breach of contract and the claim for unlawful means conspiracy (which relies on breaches of contract as the unlawful means). I reject IBM's argument that, absent deliberate concealment, the two year period starts only from the date of knowledge of the facts constituting the breach because the clause expressly provides for the start date to run from the date on which *"the cause of action arose"*.

996. As to the application of the clause to the other defendants, IBM relies on *Credit Suisse First Boston v MLC* [1999] 1 Lloyd's Rep 767, a case concerning the scope of a jurisdictional clause. The relevant clause (clause 5.2) stated: *"The courts of England are to have jurisdiction to settle any disputes which may arise out of or in connection with this Agreement and accordingly any legal action or proceedings arising out of or in connection with this Agreement ("Proceedings") may be brought in such courts."* The court rejected the claimant's argument that the reference to *"disputes"* in that clause applied to disputes with non-parties to the agreement per Rix J at p.777:

> "it seems to me to be far-fetched to regard "any disputes" as covering disputes between MLC and anyone other than MLC's contract partner under the Purchase Agreements, namely CS Europe. Clause 5.2 is part of a bilateral agreement between a seller and a buyer, and the disputes to which such an agreement may give rise are prima facie bilateral disputes. Indeed, it is I would have thought axiomatic that, at any rate in the absence of plain language to the contrary, a contract seeks neither to benefit nor to prejudice non-parties: even where such plain language is used, it is black-letter law that the non-party can himself neither take the benefit nor suffer the burden of the contract".

997. I recognise that the case involved consideration of a jurisdictional clause rather than a limitation clause but I consider it apposite here. The ICA was a bilateral agreement between IBM and Winsopia. Any legal action arising out of the ICA would be in respect of bilateral disputes. There is nothing in the language of clause 1.11.4 to suggest that it is intended to have an ambit beyond the parties to the ICA, or to affect either party's rights to bring claims against third parties. For the same reason, clause 1.11.4 cannot be read as containing a promise by IBM not so sue third parties more than two years after the cause of action arose. A similar approach was taken to the interpretation of a limitation provision by Fraser J (as he then was) in *Bloomberg v Sandberg* [2015] EWHC 2858 (TCC) at [21] and [25].

998. Although the defendants correctly point out that the tort of procuring a breach of contract is a form of secondary liability, it does not depend on a continuing right of legal action in respect of the underlying breach. In any event, this argument would not affect the claims of unlawful means conspiracy, which is a tort of primary liability.

999. Accordingly, in my judgment, clause 1.11.4 and the other terms of the ICA do not apply to parties other than IBM and Winsopia. Therefore, clause 1.11.4 does not preclude IBM's claims against the other defendants.

1000. IBM contends that, on its proper construction, clause 1.11.4 does not apply where the cause of action involves dishonest conduct or the defendant dishonestly sought to conceal relevant matters from the claimant; or where a breach was deliberately concealed by the defendant.

1001. Reliance is placed on *Granville Oil v Davis Turner* [2003] EWCA 570, a case concerning the enforceability of a contractual limitation period of nine months which barred a claim brought outside that period. The Court of Appeal held that in the absence of express words, the clause did not operate to bar claims where there was fraud or fraudulent concealment per Tuckey LJ at [15]:

> "I think it is an inescapable conclusion from what he said that the judge did think that the clause applied to a claim for fraud and to a claim which had been fraudulently concealed by the conduct of the freight forwarder. The judge was not asked to construe the clause so widely and I do not think such a construction was justified. The clause is obviously designed to

> meet ordinary contractual claims such as those made in this case which a freight forwarder would expect to have to face in the ordinary of course of his business."

1002. Reliance is also placed on *4Eng Ltd v Harper* [2007] EWHC 1568 (Ch), a case in which the court held that a contractual time bar did not apply for a claim for fraudulent warranties per Briggs J at [66]:

> "… it does not follow, in my judgment, that the parties, in giving and receiving the warranties in the sale agreement and imposing a two-year period for giving notice of warranty claims, must be taken to have contemplated not merely that there might be a fraud but that the fraud would be deliberately concealed by the warrantors. For example, the relevant fraud could be one committed by an employee unknown to and therefore inevitably not disclosed by directors acting honestly and in good faith. It is, in my judgment, well arguable by parity of reasoning with the passage in the Granville Oil case, to which I have referred, that clause 5.1 is not as a matter of construction applicable to warranties fraudulently given in circumstances where the truth and thereby the breach of warranty is deliberately concealed. I would go further and say that, in my judgment, and this is a pure question of construction, if it were necessary to decide the question now I would indeed hold that clause 5.1 does not so extend."

1003. Although the above cases turned on their specific facts, the common theme is the established rule of construction set out in *Alghussein Establishment v Eton College* [1988] 1 WLR 587 (HL) at p.595, that clear express provisions are required to contradict the presumption that it was not the intention of the parties that either should be entitled to take advantage of his own wrong to obtain a benefit as against the other party.

1004. As a matter of construction of clause 1.11.4, there are no words, let alone clear express words, indicating that the parties intended it to bar a claim which was dishonestly or deliberately concealed by the defendants. I accept IBM's submission that, if the position were otherwise, it would allow a party to escape liability by taking advantage of its own dishonest or deliberate concealment of wrongdoing, which would have the perverse effect of encouraging a party purposefully to conceal its wrongdoing and thus depriving the innocent party of any effective remedy. In those circumstances, the presumption that the parties intended the provision to have no effect, where Winsopia dishonestly or deliberately concealed its own wrongdoing, until IBM discovered the concealment or with reasonable diligence could have done so, must prevail.

1005. I reject IBM's case that the two-year limitation period should be disapplied entirely where there has been dishonest or deliberate concealment. Such concealment might operate for a very short period of time before disclosure to or discovery by IBM. In those circumstances, there would be no sound reason for disapplying the agreed limitation period following IBM's notice of the breach. On that basis, the clause should be construed in line with section 32 of

the Limitation Act 1980, namely, that time is postponed where a fact relevant to a party's right of action has been deliberately concealed or there has been a deliberate commission of a breach of duty in circumstances in which it is unlikely to be discovered for some time but time starts to run when a party has discovered the concealment or with reasonable diligence could have done so.

1006. I reject as having no merit IBM's alternative argument based on estoppel by implied representation. Winsopia's use of the mainframe software and payments made under the ICA could not amount to an implied representation that there was no wrongdoing.

1007. In conclusion on this issue, clause 1.11.4 is an effective time bar provision as between IBM and Winsopia in respect of the claims against Winsopia but does not apply to the claims against the other defendants and does not apply to wrongdoing where there has been deliberate concealment until IBM discovered the concealment or with reasonable diligence could have done so.

Statutory Limitation

1008. Section 5 of the Limitation Act 1980 provides that an action founded on simple contract shall not be brought after the expiration of six years from the date on which the cause of action accrued.

1009. IBM's position is that most of the contractual breaches occurred after 21 September 2015, within the limitation period of six years. It concedes that insofar as the technical breaches against Winsopia pre-date 21 September 2015, that part of its claim is *prima facie* statute-barred under section 5 of the Limitation Act 1980.

1010. Section 2 of the Limitation Act 1980 provides that an action founded on tort shall not be brought after the expiration of six years from the date on which the cause of action accrued.

1011. Damage is a necessary element of both tortious claims, unlawful procurement of breach of contract and unlawful means conspiracy. It is submitted by IBM that it did not suffer actionable damage before January 2021, when Mr Wallin of IBM became aware that Winsopia was likely to be in breach of the ICA. Only after that date did IBM start to incur substantial management time and costs in exercising its audit rights and bringing proceedings.

1012. The defendants' position is that for the purpose of the alleged causes of action in unlawful procurement of breach of contract and unlawful means conspiracy, any damage, if any, was suffered no later than when the SDM began to be marketed in March 2016. The Original Claims in tort were brought within six years of this date but not the New Claims.

Deliberate concealment

1013. IBM's case is that, in respect of those parts of the claims that are found to be *prima facie* statute-barred in contract and/or tort, the defendants are deprived of

any limitation defence by reason of section 32(1)(b) and/or 32(2) of the Limitation Act 1980.

1014. The defendants' case is that they did not deliberately conceal the link between LzLabs and Winsopia (section 32(1)(b)), or deliberately commit any breach in circumstances in which it was unlikely be discovered (section 32(2)). Further, IBM has known about the link between LzLabs and Winsopia since 2013 and could with reasonable diligence have discovered the link from that date onwards.

1015. The relevant provisions in Section 32 of the Limitation Act 1980 are:

> "(1) … where in the case of any action for which a period of limitation is prescribed by this Act, either—
>
> …
>
> (b) any fact relevant to the plaintiff's right of action has been deliberately concealed from him by the defendant;
>
> …
>
> the period of limitation shall not begin to run until the plaintiff has discovered the … concealment … or could with reasonable diligence have discovered it. References in this subsection to the defendant include references to the defendant's agent and to any person through whom the defendant claims and his agent.
>
> (2) For the purposes of subsection (1) above, deliberate commission of a breach of duty in circumstances in which it is unlikely to be discovered for some time amounts to deliberate concealment of the facts involved in that breach of duty."

1016. Thus, there are two alternative legal bases from which IBM argues that the defendants are deprived of reliance on their limitation defences:

i) under section 32(1)(b), the limitation period is postponed where any fact relevant to a claimant's right of action has been deliberately concealed by the defendant;

ii) under section 32(2), the limitation period is postponed where the defendant deliberately commits a breach of duty in circumstances where it is unlikely to be discovered for some time.

1017. The section 32(1)(b) test was considered in *Canada Square v Potter* [2023] UKSC 41:

> "[96] What section 32(1)(b) requires is that the defendant has "deliberately concealed" "a fact relevant to the plaintiff's right of action". The words "the plaintiff's right of action" must refer to the right of action asserted by the plaintiff in the proceedings before the court. That follows from the terms of section 32(1), so

> far as material: "… where in the case of any action for which a period of limitation is prescribed by this Act … any fact relevant to the plaintiff's right of action has been deliberately concealed from him by the defendant …" The right of action asserted by the plaintiff may or may not be well-founded: that is a matter which will only need to be determined if the plea of limitation is rejected. As to the words "a fact relevant to the plaintiff's right of action", that phrase has been interpreted as referring to a fact without which the cause of action is incomplete: see, for example, *Arcadia Group Brands Ltd v Visa Inc* [2015] EWCA Civ 883; [2015] Bus LR 1362. That interpretation is not in issue in this appeal, but it makes sense: if the claimant can plead a claim without needing to know the fact in question, there would appear to be no good reason why the limitation period should not run.
>
> …
>
> [98] … the word "conceal" means to keep something secret, either by taking active steps to hide it, or by failing to disclose it. A person who hides something can properly be described as concealing it, whether there is an obligation to disclose it or not.
>
> …
>
> [109] The elaborate and confusing analyses of section 32(1)(b) put forward in *Williams, The Kriti Palm* and the present case represent a wrong turning in the law. It should return to the clarity and simplicity of Lord Scott's authoritative explanation in *Cave* (para 60):
>
> > "A claimant who proposes to invoke section 32(1)(b) in order to defeat a Limitation Act defence must prove the facts necessary to bring the case within the paragraph. He can do so if he can show that some fact relevant to his right of action has been concealed from him either by a positive act of concealment or by a withholding of relevant information, but, in either case, with the intention of concealing the fact or facts in question."
>
> What is required is (1) a fact relevant to the claimant's right of action, (2) the concealment of that fact from her by the defendant, either by a positive act of concealment or by a withholding of the relevant information, and (3) an intention on the part of the defendant to conceal the fact or facts in question."

1018. The section 32(2) test was summarised in *Canada Square* at [153]:

> ""Deliberate", in section 32(2), does not include "reckless". Nor does it include awareness that the defendant is exposed to a claim. As Lord Scott said in *Cave* at para 58, the words

> "deliberate commission of a breach of duty" are clear words of English. They mean, as he added at para 61, that the defendant "knows he is committing a breach of duty.""

1019. In this case the relevant fact that was concealed is said to be the connection between Winsopia and LzLabs; in particular, that Winsopia was a wholly-owned subsidiary of LzLabs, whose sole or primary purpose was to assist LzLabs to develop the SDM, using access to the mainframe software and breaches of the ICA, as set out in Paragraph 11 of the Further Particulars of Concealment dated 21 October 2022:

> "11.1 It was decided to use the Second Defendant as a "shell" for the purposes of entering into the ICA and acquiring access to IBM proprietary materials licensed thereunder, including the IBM Mainframe Software…
>
> 11.2 From the moment of its incorporation onwards, the sole or primary purpose of the Second Defendant was to assist the First Defendant and its associates (including at least Texas Wormhole and OnTarget Group) and later the Third Defendant in the development of the SDM, including by committing multiple breaches of the ICA.
>
> 11.3. At the same time, the Defendants went to great lengths, and took repeated steps, to hide the fact and nature of the connection between the First and Second Defendants. Those steps, which are further particularised below, were taken so as to give to any external observers, and in particular the Claimant in the event of audit or litigation, the false impression that the SDM had been developed lawfully by the First Defendant, without any involvement of the Second Defendant and thus without any breaches of the ICA being committed or procured. In the premises, each and every one of those steps was taken in order to conceal (i) the multiple breaches of the ICA committed by the Second Defendant during the development of the SDM; and (ii) the First and Third to Fifth Defendants' procurement of those breaches."

1020. IBM's case is that it did not discover, and could not with reasonable diligence have discovered, the concealment prior to 25 August 2020. Accordingly, the six-year limitation period runs from 25 August 2020, with the result that all the Original Claims and New Claims were brought within time.

1021. The defendants' case is that there was no deliberate concealment and that IBM had sufficient knowledge to trigger time running for the purposes of section 32, more than six years before the date of issue of proceedings, either because it had actual knowledge of what it alleges was concealed or because it could with reasonable diligence have discovered it.

1022. The allegations of deliberate concealment are that the defendants:

i) used an independent company, namely, Winsopia, for the purpose of acquiring a mainframe and entering into the ICA, so as to conceal the connection with LzLabs;

ii) instituted a policy requiring Winsopia employees to identify themselves as LzLabs employees when in contact with LzLabs customers and adopted a dual email scheme whereby employees of LzLabs, Winsopia and LzLabs UK were required to use a Winsopia email address when communicating with IBM;

iii) removed references to Winsopia from public facing materials; and

iv) refrained from using IBM technical support.

1023. In respect of the first allegation, IBM's case is that Winsopia was set up as an independent corporate entity to create the appearance of separation between the ICA licensee and LzLabs.

1024. On 13 March 2013, following a meeting and conference call with LzLabs and Texas Wormhole, Mr Rastall sent an email to Mr Rockmann, stating:

> "We have a common initial understanding of the long term goal and how we may proceed in the short term. Much will depend on if and when we have mainframe access. In summary we have a green light from everybody. The next step is creating a company structure for us to operate with here".

1025. In cross-examination, Mr Rastall agreed that the long term goal was the use of a UK company which would operate a mainframe for the purposes of assisting in the development of the SDM. He also confirmed that, from the outset of his involvement in the project, he was concerned that if IBM learnt that Winsopia was using its mainframe to assist in the development of a competitor product, IBM could take steps to thwart the development of the SDM by, for example, terminating the software licence:

> "Q. The fact that Winsopia was assisting in the development of the SDM was not a fact which you would have wanted IBM to know; is that fair?
>
> A. Yes.
>
> Q. And similarly, the fact that Winsopia was owned by LzLabs was not a fact which you would have wanted IBM to know?
>
> A. I thought it was inevitable that they would know.
>
> …
>
> Q. So your evidence is that you would not have wanted IBM to know that fact; is that right?
>
> A. Yes."

1026. On 15 March 2013 Winsopia was incorporated.

1027. On 18 June 2013, David Janicek of Texas Wormhole emailed Mr Moores and Mr Rockmann with an outline scope of work for Winsopia:

> "This document will serve as a starting point as to the workload I will be deferring to Keith's company due to my inability to access IBM copyrighted materials as well as the fact that I'm an admittedly poor QA. :)…
>
> A)QA 1) I won't begin to try to list every single test that QA will need to attempt. Let's just say there will literally be thousands…
>
> B) Research - this is the research I have to throw over the "Chinese wall" …
>
> C) Development - tasks that need to be built to execute on the z/OS systems…"

1028. Mr Moores made the following observation regarding recruitment of employees for Winsopia:

> "It would be preferable, as Thilo implies, for these folks to be in the UK. I.e., no business or personal relationship with anyone here, etc. I don't think this is gonna be much of a problem for Keith. "

1029. Mr Moores agreed that it was important that there was no outward appearance of a relationship between LzLabs and Winsopia, stating that would be "a real plus" but disagreed that there was any intention to conceal this from IBM.

1030. On 5 July 2013 Winsopia was acquired by LzLabs as a wholly-owned subsidiary.

1031. On 9 July 2013, Mr Rastall emailed Mr Wilson at RSM, asking RSM to obtain pricing for an IBM software licence. Mr Moores and Mr Rockmann discussed whether they should buy a new mainframe from IBM or a used mainframe from a third-party reseller. Mr Moores' view was that it would be preferable to buy a new mainframe. However, Mr Rastall explained to Mr Moores and Mr Rockmann in his email dated 27 July 2013 that the advantages of buying a used machine went beyond reduced costs:

> "IBM would not be involved AT ALL with supply of hardware, install, de-installs, upgrades, capacity planning, monitoring usage, visiting Winsopia premises. IBM's only involvement with Winsopia once User Agreement signed would be to deliver the software to RSM partners on our behalf".

1032. On 29 July 2013 Mr Rastall forwarded to Mr Rockmann a quotation for the purchase of a used mainframe from GMT360, stating:

> "Only GMT360 would be involved in supply, install, de-install, maintenance, including spare parts".

1033. Notwithstanding Mr Rockmann's protestation that this discussion was solely about cost-savings, on the face of the emails the advantages identified by Mr Rastall extended to the absence of usage monitoring by IBM or visits by IBM to Winsopia. In cross examination, Mr Rastall agreed that if IBM had been involved in the supply, relocation and maintenance of the mainframe, there would be an obvious risk that IBM would learn that Winsopia was using the mainframe to enable the development of the SDM. By purchasing a used mainframe, in addition to the cost saving, it would avoid IBM learning that Winsopia would be using it to assist in SDM development:

> "Q. So, if IBM was involved in the supply, relocation and maintenance of the mainframe, there would be an obvious risk that IBM would learn that Winsopia was using the mainframe to enable the development of the SDM; is that fair?
>
> A. Yes.
>
> Q. And buying a used mainframe, as you were suggesting in the email we just looked at, was a suggestion you were putting forward to cut IBM out of the loop in order to avoid it learning that Winsopia would be using the mainframe to assist in SDM development?
>
> A. The main focus of buying a used mainframe was cost. The cost differential was absolutely huge. This was --this would be a byproduct of that.
>
> Q. Well, when you say "a byproduct", I think what you mean is an additional reason?
>
> A. Yes."

1034. On 1 August 2013 Winsopia purchased the used mainframe from GMT360, which was initially installed at RSM's offices and later moved to Winsopia's office in Farnborough.

1035. RSM negotiated the terms of the ICA for Winsopia, avoiding any direct contact between Winsopia and IBM, although the ICA was signed by Mr Rockmann and Mr Rastall for and on behalf of Winsopia. Mr Rastall's evidence was that he had a close working relationship with Mr Wilson at RSM and expected him to keep Mr Rastall's exchanges with RSM confidential.

1036. During an exchange of messages between Mr Cresswell and Mr Rockmann on 25 March 2020, Mr Cresswell referred to Winsopia as "a shell" and suggested that his proposal to shut down Winsopia would create "even greater separation between LzLabs [and] the ICA".

1037. Contrary to the defendants' submissions, LzLabs did need, or more accurately chose, to use a mainframe in breach of the ICA licence to develop the SDM. Winsopia was incorporated by Mr Rastall but was almost immediately taken over by LzLabs and, thereafter, operated under LzLabs' direction and control for no other purpose than to assist and facilitate development of the SDM. Although there may well have been other factors involved, such as cost and convenience, a material factor in the decision to use RSM to acquire a used mainframe was the desire to avoid scrutiny by IBM.

1038. From the above evidence, I find that steps were taken by LzLabs, Winsopia, Mr Rastall, Mr Rockmann and Mr Moores to ensure that Winsopia was seen as an independent entity with no obvious links to LzLabs, concealing from IBM the fact that Winsopia was a wholly-owned subsidiary of LzLabs and whose sole purpose was to assist LzLabs in development of the SDM.

1039. In respect of the second allegation, the SDM project launch took place in March 2016. On 21 September 2016, Mr Rastall of Winsopia contacted Mr Wehrli of LzLabs, proposing that Winsopia should be directly involved in discussions with potential customers to gain a rapid understanding of their needs. Mr Wehrli responded, saying that he could see the benefit of such direct communications but that he would need to discuss with this with Mr Rockmann, Mr Cresswell and the sales team. Mr Wehrli forwarded this email exchange to Lukas Do of LzLabs, asking for his feedback and stating:

> "… we also don't want to reveal the Winsopia name (as in the trade register directly as a subsidiary and IBM might therefore terminate Winsopia's license for "Z")."

1040. Mr Do responded (as translated by Mr Wehrli during his evidence):

> "The situation you described below is tricky. On the one hand it would be good to have an intermediate who has the required mainframe skills to communicate both with external clients and with Winsopia staff. On the other hand I see a potential large problem that we would provide IBM with large attack surface and we would lean ourselves out of the window quite heavily."

1041. Mr Wehrli sent a further email to Mr Rastall, stating:

> "After discussions, we feel that involving Winsopia with the customer is beneficial in certain cases, and we should do that.
>
> However we have to be aware of the following things:
>
> - We do not want Winsopia's name advocated out there, this is due to the IBM license agreement. We would not like to advocate the Winsopia name associated with LzLabs, because IBM could cancel the agreement for z/OS. Meaning, Winsopia Employees when talking to prospects or customers have to identify as LzLabs employees …"

1042. Mr Rastall issued this instruction to all Winsopia employees. It is clear from the above exchanges that Winsopia employees were told to pretend that they were LzLabs employees when talking to existing or prospective customers so that IBM would not discover Winsopia's involvement in the SDM.

1043. In about May 2020, certain Winsopia and LzLabs employees were provided with two email addresses, @lzlabs.com and @winsopia.co.uk. On 28 May 2020 Markus Liebenberg of LzLabs sent the following instructions regarding the use of these LzLabs and Winsopia email addresses:

> "Hi, here is the procedure we have to follow with regards LzLabs and Winsopia emails
>
> 1. A block will be implemented stopping Winsopia emails from reaching LzLabs and vise versa
>
> 2. All internal communications (e.g. Dev or Delivery) must be done using LzLabs emails. We must remain extremely careful with what we share in these internal emails
>
> 3. All external communications (IBM and customers sharing IBM mainframe details/resources) must be done with Winsopia email
>
> One idea I have to help minimize the switching between emails is to use a second laptop.. .there may be a few extra laptops available at the moment."

1044. Mr Playford responded:

> "… it does put more strain on replies to emails, meaning we have to be very cognisant of to whom we are replying, and is also likely to lead to (human) lapses from time to time - which could be costly or disastrous?!"

1045. Mr Playford was asked about this email in cross-examination:

> "Q. At this point in time, is this correct, you had a suspicion that IBM UK or IBM Corp may have known about the connection between Winsopia and LzLabs, but it's not something you knew for certain?
>
> A. That's correct.
>
> Q. And your understanding is that it would be a disaster if IBM were to learn of that connection; is that right?
>
> A. Possibly.
>
> …

> Q. Now, in this email, you don't spell out the reason why you think it would be a disaster, or possibly a disaster, and presumably there was no need to spell that out because you understood it would have been equally obvious to everybody else on this email chain; is that fair?
>
> A. I suppose that's fair, yes."

1046. In a subsequent email dated 12 June 2020, Mr Playford stated:

> "Does Duncan Love have a Winsopia email account?
>
> The only reason I ask is that Mark Cresswell said that Duncan Love may be the person to take over IBM administrative rights (for granting access to problems, software ordering, SCRT reporting etc). And that wouldn't be possible if he only has an LzLabs email account?!
>
> Goes again[st] the Code of Conduct and rules surrounding how we contact external agencies etc.
>
> And IBM would certainly smell a rat! :-)"

1047. Mr Rockmann stated in cross-examination that he approved this policy of dual email usage to separate communications with IBM about mainframe matters between Winsopia, the ICA licensee, and LzLabs, who was not an ICA licensee:

> "Q. … You do not want communications with IBM taking place on LzLabs email, do you? That is the point of this, Mr Rockmann.
>
> A. It is the point, because obviously whatever IBM sends to Winsopia is supposed to stay at Winsopia.
>
> Q. Mr Rockmann, it is also to ensure that no communications were happening with IBM other than on a Winsopia email?
>
> A. It is, as I described, for that purpose."

1048. Mr Rockmann's suggestion that this arrangement was simply to accommodate the transfer of LzLabs UK employees to Winsopia might explain the decision to issue dual email addresses but it does not explain the policy decision to ensure that communications with IBM should only be through the Winsopia email address.

1049. The emails are clear. From 2016, when the defendants started to market the SDM, they took steps to conceal Winsopia's links to LzLabs and its role in developing the SDM. I acknowledge that the defendants have identified a number of customers or potential customers with whom Winsopia had direct communication without disguising its identity. However, they did not include IBM and the policy of using false identities was not revoked. The defendants submit that there was no need for, or obligation on, LzLabs to advertise the role

played by Winsopia to potential customers and it would have made no commercial sense to do so. It is said that there was nothing unusual, controversial or dishonest about that; it is standard commercial behaviour. That may well be the case but it does not detract from the fact that the defendants did conceal, or not disclose, to IBM, Winsopia's role in developing and marketing the SDM.

1050. In respect of the third allegation, in May 2016, LzLabs and Winsopia were planning to release a public video demonstration of the DMA (subsequently re-named the CPX). The purpose of the video was to show a test application running on a mainframe, followed by migration of the application from the mainframe to the SDM. Part of the video involved a segment recorded at Winsopia showing the test application running on the Winsopia mainframe.

1051. In an email dated 10 May 2016, Mr Tyneski of LzLabs explained that there would be no mention of the customer site (i.e. Winsopia):

> "… - Mark and Thilo will approve the final script and seek legal approval of content
>
> - The video will be shot in Farnborough by Recipe / post production will be by BLC in Zug
>
> - No mention of the customer site - and perhaps no mention of the narrator…"

1052. As part of this dialogue, on 11 May 2016, Mr Palmer expressed his concern that the source of the application to be used should be concealed. I note that this appears to be concerned with customer confidentiality rather than Winsopia:

> "The obvious preference would be to use an existing application that has been tested on the LzAppliance and contains both CICS and DB2 objects, so is there something we can just grab? We may need to change data set names etc. to conceal the source of the application and then re-test and we will also need to confirm this with legal and LzLabs (Thilo & Mark)."

1053. On 25 May 2016 Mr Tyneski sent an email following a conference regarding the demonstration:

> "- The new application is not yet ready for testing; Winsopia will try to have this done by early next week (asap).
>
> - Once the app is ready, Chris will run another test video to include showing the app actually running on the M/F.
>
> - The updated video will allow us to check for time & also to ensure that the app can be unpacked on our side & actually works.

> - In addition to the updated QT video, it would be useful to have an updated script that can be run past Legal for a quick review the first one went OK.
>
> - Everyone was reminded that it is vital not to show the actual location of the demo & also to avoid the use of registered IBM Trademarks wherever possible… ”

1054. On 26 May 2016 Adrian Hudson of Winsopia sent an email stating:

> “OK, so there is nothing on the IC07/IC08 screens to indicate they came from Winsopia. The DMA ftp strips my UserlD from the front of transmitted datasets. So, are we saying that documentation should not show and names starting "WlNxx"?”

1055. Mr Palmer responded:

> “Yes because the DMA demo will show those names in the interface when we perform the migration. Please check with Thilo/legal is you have questions I am just sharing my experiences and trying to prevent last minute problems.”

1056. Mr Hudson replied:

> “OK, I will change the form and remain cognizant of this requirement.
>
> BTW, the Winsopia company records, available on the Web, show us as a subsidiary of LzLabs.”

1057. Mr Rastall confirmed in cross-examination that he gave the instruction to strip out from the public facing demonstration all references which could identify the involvement of Winsopia, the purpose of which was to conceal the link between LzLabs and Winsopia from the public eye. Mr Cresswell was dismissive of the notion that the “WINxxx” pre-fix was sensitive but the fact remains that those concerned agreed that it should be obscured.

1058. The final version of the video was approved by Mr Cresswell and Mr Rockmann. It showed the LzLabs logo but did not contain any references to Winsopia.

1059. The defendants accept that these exchanges show that some individuals at Winsopia and LzLabs were concerned about advertising the link between LzLabs and Winsopia to IBM but submit that they do not establish a policy to conceal the link or amount to concealment. Taken in isolation, the above exchanges could be interpreted as examples of over-zealous concern to preserve confidentiality of third parties and focus attention on LzLabs. However, when read in conjunction with the other concealment evidence, it is apparent that they formed part of a concerted strategy to disguise Winsopia’s involvement with the SDM from IBM.

1060. In respect of the fourth allegation, consistent with the desire to hide from IBM Winsopia's connection with, and work for, LzLabs, during the project, Winsopia avoided contacting IBM for technical support through problem management records ("PMRs").

1061. For example, in his emails dated 6 and 7 September 2018 to Mr Taylor and others, in response to a suggestion by Douglas ("Dougie") Lawson of Winsopia that an IMS DB function failure should be the subject of a PMR, Mr Rastall stated:

> "…Raising a PMR with IBM (if and when we are ready) will require discussion and agreement with Thilo and Legal it is not something we have done previously.
>
> … if we consider the function is not working as documented by IBM I need to decide if you recommend raising a PMR with IBM, if you do I will need further discussion with legal and Thilo about opening pandoras box to IBM who could request information we may not want to disclose."

1062. Likewise, in an email dated 16 April 2020, Alan Playford stated that he wanted to exhaust all possible avenues before reporting a possible bug to IBM. On 17 April 2020 Mr Bleach suggested opening an APAR to which Mr Playford replied:

> "Will certainly do if absolutely necessary?
>
> But that's why I'm almost begging Dougie to run our tests on a different Customer machine that he might have access to, just to see where the actual problem is?
>
> Once I know we can't possibly attribute it to ourselves, then I'll certainly raise a PMR!"

Finding - section 32(1)(b)

1063. The above evidence shows that Mr Moores, Mr Rockmann and Mr Cresswell were participants in the acts of concealment. As explained in relation to the procurement breaches, by reason of their roles as directors and officers of the corporate defendants, the acts and knowledge of Mr Rockmann and Mr Cresswell are attributable to LzLabs, Winsopia and LzLabs UK.

1064. I acknowledge that Mr Rockmann's explanation was that when LzLabs was in the early stages of winning customer engagements, it did not want to introduce Winsopia into the picture because it made dealings more complicated:

> "The more you offer information to potential customers, the more questions they will have, and the connection with Winsopia was largely irrelevant to customers in the early stages."

1065. This view was shared by Mr Cresswell who stated:

> "Any reduction in complexity or confusion or increase in clarity that you can – that you can put into a marketing message is – is a good thing."

1066. This was echoed also by Mr Moores:

> "In Marketing 101, there's no sense in creating brand awareness of something that is not going to help you in dealing with customers. The customers were concerned about a relationship with LzLabs. Introducing Winsopia into a discussion wouldn't have been particularly helpful. Likewise, most software companies do not publicise their rather extensive third party relations with many other software companies…it just doesn't make any business sense."

1067. Nonetheless, what the defendants embarked on was a series of steps and policies to conceal from IBM that Winsopia was a wholly-owned subsidiary of LzLabs, set up to acquire a mainframe and software licence, using what I have found to be breaches of the ICA, with the sole purpose of assisting and facilitating development of the SDM.

1068. Those facts were relevant to IBM's right of action; the concealment of those facts occurred either by positive acts of concealment or by withholding the relevant information; and the defendants intended to conceal those facts. This amounted to deliberate concealment for the purpose of section 32(1)(b) of the Limitation Act 1980.

Finding - Section 32(2)

1069. Given my findings above, it is not necessary to determine the alternative case under section 32(2) of the Limitation Act 1980 but I deal with it briefly.

1070. As set out above, under section 32(2), the limitation period is postponed where the defendant deliberately commits a breach of duty in circumstances where it is unlikely to be discovered for some time.

1071. For the reasons set out above, I have found that Winsopia was in breach of the ICA; LzLabs and Mr Moores unlawfully procured those breaches of contract; and Winsopia, LzLabs and Mr Moores combined with each other to develop the SDM using unlawful means. The breaches of duty by each of Winsopia, LzLabs and Mr Moores were deliberate. The breaches were carried out behind closed doors, at Winsopia and LzLabs, in circumstances where they were unlikely to be discovered for some time.

1072. Therefore, the alternative case of concealment based on section 32(2) is established.

Actual or constructive knowledge – legal principles

1073. The relevant dates of knowledge for limitation purposes are:

i) pre-21 September 2015 (statutory limitation cut-off for the Original Claims);

ii) pre-29 November 2017 (statutory limitation cut-off for the New Claims);

iii) pre-21 September 2019 (contractual limitation cut-off for the Original Claims); and

iv) pre-29 November 2021 (contractual limitation cut-off for the New Claims).

1074. The relevant test to be applied to section 32(1) cases was set out in respect of mistakes of law by the Supreme Court in *Test Claimants in the FII Group Litigation v Revenue and Customs Commissioners* [2020] UKSC 47 at [213]:

> "(1) Limitation periods set a time limit for issuing a claim, which normally begins to run when the cause of action accrues. They apply whether the substance of the claim is disputed or not. They apply to claims regardless of whether there is in truth a well-founded cause of action.
>
> (2) Section 32(1) of the 1980 Act postpones the running of time beyond the date when the cause of action accrues, in cases where the claimant cannot reasonably be expected to know at that time the circumstances giving rise to the cause of action, by reason of fraud, concealment or mistake. Its effect is that the limitation period commences not on the date when the cause of action accrues, but on the date when the claimant discovers, or could with reasonable diligence discover, the fraud, concealment or mistake.
>
> (3) Consistently with (1) above, section 32(1) cannot be intended to postpone the commencement of the limitation period until the claimant discovers, or could discover, that his claim is certain to succeed.
>
> (4) Consistently with (1) above, section 32(1) cannot be intended to postpone the commencement of the limitation period until the proceedings have been completed.
>
> …
>
> (13) The purpose of the postponement effected by section 32(1) is to ensure that the claimant is not disadvantaged, so far as limitation is concerned, by reason of being unaware of the circumstances giving rise to his cause of action as a result of fraud, concealment or mistake. That purpose is achieved, where the ingredients of the cause of action include his having made a mistake of law, if time runs from the point in time when he knows, or could with reasonable diligence know, that he made such a mistake "with sufficient confidence to justify embarking

> on the preliminaries to the issue of a writ, such as submitting a claim to the proposed defendant, taking advice and collecting evidence"; or, as Lord Brown put it in Deutsche Morgan Grenfell, he discovers or could with reasonable diligence discover his mistake in the sense of recognising that a worthwhile claim arises (paras 193 and 209).
>
> …
>
> (16) Authorities concerned with the meaning of "reasonable diligence" in section 32(1) also indicate that it is concerned with the steps which a person could reasonably be expected to take before issuing a claim (para 203 above). The standard of reasonable diligence is how a person carrying on a business of the relevant kind would act, on the assumption that he desired to know whether or not he had made a mistake, if he had adequate but not unlimited staff and resources and was motivated by a reasonable but not excessive sense of urgency. The question is not whether the claimant should have discovered the mistake sooner, but whether he could with reasonable diligence have done so. The burden of proof is on the claimant. He must establish on the balance of probabilities that he could not have discovered the mistake without exceptional measures which he could not reasonably have been expected to take (para 209).
>
> (17) Authorities concerned with the pre-1939 equitable rule on which section 32(1) is based also support the view that the limitation period runs from the time when the claimant discovers the facts essential to his cause of action, and not from the date of a judicial decision supportive of his claim (paras 204—208 above)."

1075. The Court of Appeal held that this test applied to other cases concerning section 32(1) in *Gemalto Holding BV v Infineon Technologies AG* [2022] EWCA Civ 782 per Sir Geoffrey Vos MR:

> "[45] In my judgment, the parties were right to submit that, after FII, limitation begins to run in a deliberate concealment case when the claimant recognises that it has a worthwhile claim, and that a worthwhile claim arises when a reasonable person could have a reasonable belief that (in a case of this kind) there had been a cartel. Gemalto's four propositions overcomplicate the position. The FII test must be applied with common sense. As the judge held, there is unlikely in most cases, as in this case, to be a real difference between the application of the statement of claim test and the FII test. Indeed, the statement of claim test is, perhaps, little more than a gloss on the FII test. It is also worth noting that competition cases are not to be treated differently from other cases under section 32 (see Arcadia [2015] Bus LR 1362 at para 51).

[46] First, the FII test makes clear that the claimant is not entitled to delay the start of the limitation period until it has any certainty about its claim succeeding. So, whilst in a fraud case, if there were an essential fact about the fraud that the claimant had not discovered, without which there would have been no fraud, it would make sense to say that the claimant had not discovered the fraud. But in concealment, what needs to have been discovered is just that, the concealment. Once the claimant knows objectively that a cartel has been concealed, it does not need to have certainty about its existence or about the details of that cartel. That is why the Supreme Court made clear that the claimant needs only sufficient confidence to justify embarking on the preliminaries to the issue of a writ, such as submitting a claim to the proposed defendant, taking advice and collecting evidence. The term "worthwhile claim" is also not to be construed as a deed. It requires a common-sense application. A claim in respect of a concealed event would not be a worthwhile one if it were pure speculation, but it would be if, as in this case, an authoritative regulator had thought it sufficiently serious, having investigated all the evidence available, to lay charges or issue a Statement of Objections.

[47] Secondly, the test adumbrated by the Supreme Court must be intended to operate in all situations in which there has been mistake, fraud or concealment, and to be consistent with the Limitation Act more generally. It would make no sense for the limitation period for a road traffic accident to start running when it happens (at which point the victim may know nothing about the circumstances of the accident that, for example, rendered them unconscious), but for section 32 to allow a claimant a lengthy period of investigation before it is said to have "discovered" that the facts relating to its claim have been concealed. The person who is run down knows that they have a worthwhile claim, even if they may eventually be shown to have been responsible for the accident by running in front of the vehicle. The claimant cannot postpone the start of the limitation period until it has had the time to investigate the details of the claim and the possible defences and to evaluate its prospects, any more than the road traffic victim is able to do so. That is what the six-year limitation period is for. The question of whether a claim is worthwhile is not a complex balance of the chance of success as Mr Turner suggested. The limitation period is not postponed until the claimant can show that it is more likely than not to succeed. Of course, if the putative claim would be struck out as not disclosing a cause of action, it would be right to say that the claimant had not discovered that it had a worthwhile claim (see the comparisons with Earl Beatty, Paragon, Sephton and Molloy above at para 37). That is why I say that I am far from sure that there is a real difference between the statement of

> claim test and the FII test so far as concealment cases are concerned.
>
> …
>
> [53] To summarise, therefore, the position after FII is that the proviso to section 32(1) has to be construed consistently as between mistake and deliberate concealment cases. Time begins to run in a deliberate concealment case when the claimant recognises that it has a worthwhile claim. In a case of this kind, a worthwhile claim arises when a reasonable person could have a reasonable belief that there had been a cartel. The claimant can embark on the preliminaries to the issue of a writ (and therefore the limitation has begun) once it knows that there may have been a cartel and the identity of the participants, without knowing chapter and verse about the details. It would not, however, know that it had a worthwhile claim if a claim pleaded on the basis of the details it knew would be struck out."

1076. With that test in mind, I turn to consider the relevant facts on date of actual or constructive knowledge.

Date of knowledge issues

1077. IBM's case is that it did not discover, and could not with reasonable diligence have discovered, the concealment prior to 25 August 2020.

1078. Emma Wright is a qualified solicitor who, in 2020 was the claimant's Litigation Counsel, UK & Ireland. In her statement, Ms Wright confirmed that prior to 25 August 2020, she was not aware of any link between Winsopia and LzLabs, the existence of the SDM, or any of the matters constituting breach of any software licence agreement between Winsopia and IBM.

1079. Steve Wallin, executive product development leader at IBM stated in evidence that he became aware of LzLabs, but not Winsopia, in late 2018. His recollection is that LzLabs was making announcements about a potentially new product and that it was offering a cloud based mainframe platform. When Mr Wallin heard about LzLabs, he was not particularly concerned and it did not raise any red flags. In October 2020 Mr Wallin discovered that IBM Corp was conducting an investigation into LzLabs, led by Mr Anzani. Mr Wallin's evidence is that he became aware of the relationship between LzLabs and Winsopia in January 2021, when he sent Winsopia the audit letter as set out above.

1080. Ian Mitchell, Strategic Modernisation Leader at IBM, stated that he first became aware of LzLabs in 2016, when he read on the Internet about its development of the SDM. From around the middle of 2016, Mr Mitchell assisted Mr Anzani of IBM Corp in Project Eiger. Mr Mitchell's evidence was that he did not become aware of Winsopia until these proceedings were commenced in September 2021.

1081. The defendants' case is that IBM knew, or should with reasonable diligence have known, about the link between Winsopia and LzLabs from about: (i) August 2013, when the ICA was entered into; (ii) 2017 when Mr Knight became aware of LzLabs as Winsopia's parent company; (iii) 2018 when Mr Anzani was informed of the link between LzLabs and Winsopia by Robert Soprano, which knowledge should be attributed to IBM.

(i) ICA 2013

1082. The ICA and associated agreements were signed by Mr Rockmann and Mr Rastall of Winsopia in August 2013. Nicola Bushnell of IBM returned signed copies of the agreements directly to Mr Rastall at Winsopia. In response to a request by Ms Bushnell, Mr Rastall stated that the type of business undertaken by Winsopia was information technology.

1083. On 21 August 2013 Nick Dowling of IBM emailed Mr Rastall about creating a customer number for Winsopia:

> "I'm trying to create a customer number for Winsopia on behalf of Nicola - but the request has been rejected due to an invalid VAT number. Nicola gave me the number VAT ID: CHE-164.625.611 MWST but the number should be in the following format …
>
> I've tried just using the numerals in CHE-164.625.611 MWST but that is not valid. Can you please check and send me the correct number?"

1084. Mr Rastall responded by return with the correct VAT number for Winsopia, apologising for the confusion.

1085. Ian Lyon, the First Line Sales Manager of Systems business at IBM confirmed in cross-examination that IBM kept customer records in a database at the time:

> "Q. … Now, having seen this description of the customer master record database, does that jog your memory at all in relation to the existence of such a database?
>
> A. I recognise a number of the terms in here which – that we would hold against customer records, yes.
>
> Q. And they're all sensible things that you'd need to know. You'd need to know who the client was you were with, what the legal structure of the client was, what industry they were in, who was dealing with them and so forth, yes?
>
> A. That's pretty much accurate, yes.
>
> Q. And without that, you'd be flying blind if you tried to do any sales, wouldn't you?

> A. So in terms of the sales process, it's actually unusual for sellers to use that data. But it's clearly helpful data."

1086. Mr Lyon's evidence was that in the case of a new customer, he would not expect a seller at IBM to undertake due diligence about a customer's corporate structure or shareholding. Mr Lyon stated that if a customer wanted to purchase equipment using credit, the Global Finance team at IBM, a separate part of the IBM Group, might ask a seller to obtain certain documents or provide information but this would not extend to any analysis by the sales member. As a result, he would not expect members of the sales team actively to look for red flags when engaging with new or existing customers.

1087. The defendants submit that the above exchange shows that IBM must have been aware of LzLabs because the VAT number originally obtained by Ms Bushell was the Swiss VAT number for LzLabs. It is said that the only reasonable inference for the court to draw is that as part of the registration process IBM became aware that LzLabs was linked to Winsopia (most likely as a result of checking Companies House which recorded LzLabs as the parent of Winsopia or carrying out searches in relation to Mr Rockmann) and had carried out checks in relation to LzLabs itself (hence IBM had LzLabs' VAT and Commercial Registration Numbers).

1088. The inference which the defendants invite the court to draw is a stretch given the circumstances of the exchange. The source of the LzLabs' VAT reference obtained by IBM is not obvious; it could have been provided erroneously by RSM or Winsopia; hence Mr Rastall's apology for the confusion. There is no evidence that IBM carried out any search at Companies House that would have disclosed Winsopia's parent company as LzLabs. For the reasons explained by Mr Lyon, such due diligence would only be conducted where the issue of Winsopia's credit standing was an issue, for example, if it wished to lease the equipment or purchase using credit finance. As Winsopia decided not to purchase equipment from IBM using such terms, it did not prove necessary for IBM Global Finance to carry out any checks.

(ii) Mr Knight - 2017

1089. Paul Knight was the sales manager responsible for the Winsopia account. His main point of contact at Winsopia was Mr Rastall. On 17 May 2017 Mr Knight contacted Mr Rastall by email regarding the potential purchase of a mainframe by Winsopia. Mr Knight explained in the email that if the equipment were to be leased from IBM, IBM Global Finance would require Winsopia to provide financial information:

> "For us to provide you with a firm lease price we would require full year 2016 annual accounts for WINSOPIA LTD and full year 2016 consolidated figures for top parent LzLabs GmbH to be able to make a credit decision. "

1090. This email shows that Mr Knight must have been aware that LzLabs was Winsopia's parent company; but it does not follow that he must have been aware of Winsopia's role in development of the SDM. Mr Knight explained in cross-

examination that, when he wanted to offer a lease arrangement to a client, he would first go to IBM Global Finance, who would identify what information was required in respect of the corporate entities. He stated that but for the request from IBM Global Finance, he would not have known that LzLabs was the parent of Winsopia and he otherwise knew nothing about LzLabs.

1091. In cross-examination, Mr Knight stated that he was unaware of all categories of information entered in the IBM database in respect of customers and did not consider it reliable:

> "Q. And you would have looked at the records kept by IBM in order to see who the customers were, what equipment they'd got and so on and so forth; correct?
>
> A. Yes.
>
> Q. And there would have been a database in existence at that time, at the beginning of 2013, which held that information; correct?
>
> A. Yes. Yes.
>
> Q. And can you recollect that on that database there were fields which allowed you to see what sector the company was operating in?
>
> A. Perhaps, yeah.
>
> Q. What the legal structure of the company was?
>
> A. Not so sure about that.
>
> Q. Who the parent was?
>
> A. Not sure about that.
>
> …
>
> Q. You haven't looked, have you, at the company database?
>
> A. If I'm brutally honest about IBM's database at the time, it was absolutely hopeless."

1092. On 18 May 2017, Mr Rastall replied to Mr Knight, informing him that Winsopia wanted to purchase the equipment outright, without finance. As a result, Mr Knight did not make any further inquiries into LzLabs or make any other inquiries regarding Winsopia's credit worthiness. Such limited reference to LzLabs was not sufficient to put Mr Knight, let alone IBM, on notice that Winsopia might have involvement in LzLabs' development of the SDM.

1093. The defendants submit that the link between Winsopia and LzLabs was known by IBM and was in the public domain; the link was discoverable on the

Companies House website from July 2013 and Mr Rockmann and Mr Cresswell were publicly listed as directors of both companies. However, Mr Knight's evidence was that prior to the request from IBM Global Finance, he had never previously heard of LzLabs and did not know anything about its business or products. Further, he did not know about the SDM or that LzLabs was developing any product that might be considered competitive to IBM. Because Mr Rastall promptly indicated that he did not wish to pursue any leasing arrangement, no further inquiries were made as to Winsopia's corporate structure or financial standing. Therefore, there was nothing to put IBM on notice that it should carry out a search on the Companies House website or otherwise investigate the legal structure of Winsopia and its parent company.

(iii) Mr Anzani - 2018

1094. Mark Anzani is Special Projects Executive at IBM Corp. His responsibilities include monitoring potential competitors or potential partners to IBM's mainframe technology. On his own description, his mode of operation is a bit "cloak and dagger". He freely admitted that typically he does not write a great number of things down and purposefully does not keep too many records of some of his actions; others within the IBM Group are informed only on a "need to know basis".

1095. Mr Anzani's evidence was that he first heard about LzLabs in about February 2016, when an industry contact informed him that LzLabs was expected to announce the SDM product at the forthcoming CeBIT 2016 computer conference.

1096. At the CeBIT conference on 14 March 2016 LzLabs publicly launched the SDM. Mr Anzani stated in cross-examination that he became very interested and immediately carried out initial investigations into LzLabs:

> "Q. And almost immediately you carried out some investigations into LzLabs yourself, didn't you?
>
> A. That, yes, is when I really took interest in -- in LzLabs.
>
> Q. And you became aware, extremely early on, that the ultimate owner of LzLabs was Mr John Jay Moores?
>
> A. Yes.
>
> Q. And you knew, because of your involvement in the litigation involving Neon, who he was?
>
> A. Yes, that is correct.
>
> Q. You also carried out investigation of who else was involved in relation to the matters and you discovered the existence of Texas Wormhole?

> A. Yes, I know -- I know, during the enquiries, Texas Wormhole's name came up. I don't recall exactly when in -- in the process.
>
> Q. Very early on. Very early on.
>
> A. I -- as I recall, it was -- it was relatively early, yes. Exactly when, I don't know, but it was early, yeah, I'll give you that.
>
> Q. And you also came to understand that there had been certain individuals who had transferred to Zurich in relation to the LzLabs development; correct?
>
> A. Yes, that is correct."

1097. From LzLabs' publicly available marketing materials, Mr Anzani learned that LzLabs claimed that the SDM would be able to run IBM mainframe applications in their object-level form without changes or compromise to performance. On about 25 April 2016, Mr Anzani set up a formal investigation into LzLabs and the SDM through a special project named "Project Eiger".

1098. Mr Anzani described the project as an investigation to understand the technical capabilities of the SDM that was kept separate from any marketing strategy by IBM Corp's subsidiaries. In cross-examination he accepted that there were enquiries from clients of IBM, IBM France, IBM Germany and IBM Switzerland, to which he responded. This included, on occasions, discouraging those clients from moving their applications to LzLabs. When it was put to him that from summer of 2016 he decided upon a strategy which he maintained until 2019, namely, that he would try and put customers off, but would not institute any legal proceedings, Mr Anzani stated:

> " No, I would say that that's not correct, or at least I have to be more clear and put some context around it. Before instituting any legal proceedings, you need to have facts and technical information about the solution and sufficient depth of understanding to know that, you know, if legal proceedings are going to be decided to be taken by -- by IBM, that you have a basis upon which to do it. So there was the period of time, several years, where very little information is available and we continued to work to try to reveal and understand what we could. In parallel with all of that work, I was working to do my best to ensure that customers had a set of questions that I felt were relevant, with the idea hopefully that they would continue to stay with IBM and not choose to go with the SDM."

1099. In September 2016, a two-day conference was held in Toronto for the engineers to discuss theories as to how the SDM might work. Mr Anzani stated that during this period, technical reports were produced within Project Eiger but they were not shared with others in the IBM Group.

1100. Mr Anzani's evidence is that he did not know about Winsopia in 2017. He said that he suspected that LzLabs must have access to a mainframe but he had no information as to if, or how, they had such access. He became aware of a link between LzLabs and Winsopia in July 2018, when IBM Corp's legal team were contacted by Robert Soprano, a former employee of LzLabs and former employee of IBM Corp, who told them that he thought that LzLabs was stealing from IBM and identified the involvement of a subsidiary company called Winsopia. This information was passed on to Mr Anzani.

1101. On 16 July 2018, Mr Anzani requested Marcy Nechemias, an IBM z data analyst, to carry out a search for Winsopia in the IBM Corp global database. Although an earlier search for LzLabs in October 2017 had produced no hits, the 2018 search by Ms Nechemias indicated that LzLabs was listed as Winsopia's parent company in the database.

1102. Subsequently, Mr Anzani arranged a call with Mr Soprano in September 2018, during which Mr Soprano disclosed that LzLabs was working in conjunction with Winsopia to reverse engineer IBM's code in order to steal IBM's intellectual property.

1103. Thus, by July 2018, there was evidence that amounted to a trigger, putting Mr Anzani on a course of enquiry which led to the discovery of the breaches.

1104. Mr Anzani's evidence is that throughout his initial discussions with Mr Soprano, he did not discuss or disclose the link between Winsopia and LzLabs to any employee at IBM. To the best of his recollection, the first time he discussed Winsopia with IBM was in late 2020 at the time of the Winsopia audit requests.

1105. In cross-examination, he stated that it was not until 2020 that the Project Eiger team gained an understanding of how the SDM worked, including details of the data structures that are set up as a part of the compilation process, which are not publicly available but would be necessary for a link-edited application in its binary form to operate on a different technical platform. It was this insight that led the team to believe that this must have been achieved through use of IBM's intellectual property.

1106. The defendants' position is that the knowledge of Mr Anzani (and, if necessary, members of Project Eiger, including Mr Mitchell) should be attributed to IBM, applying the general principles of agency as set out in *Meridian Global Funds Management Asia Ltd v Securities Commission* (above) per Lord Hoffmann at pp.506-507:

> "The company's primary rules of attribution will generally be found in its constitution, typically the articles of association … There are also primary rules of attribution which are not expressly stated in the articles but implied by company law …
>
> …The company … builds upon the primary rules of attribution by using general rules of attribution which are equally available to natural persons, namely, the principles of agency. It will

> appoint servants and agents whose acts, by a combination of the general principles of agency and the company's primary rules of attribution, count as the acts of the company. And having done so, it will also make itself subject to the general rules by which liability for the acts of others can be attributed to natural persons, such as estoppel or ostensible authority in contract and vicarious liability in tort."

1107. In *Bilta v Nazir* (above) Lord Mance stated at [41]:

> "As Lord Hoffmann made clear in Meridian Global, the key to any question of attribution is ultimately always to be found in considerations of context and purpose. The question is: whose act or knowledge or state of mind is for the purpose of the relevant rule to count as the act, knowledge or state of mind of the company?"

1108. This issue was considered, albeit in a different factual context, by the Privy Council in *Julien v Evolving Tecknologies and Enterprise Development Co Ltd* [2018] UKPC 2. In that case, the issue was whether knowledge of a sole shareholder should be attributed to the company for the purpose of ascertaining the date of knowledge under the relevant limitation legislation, in circumstances where claims were made against former directors for negligence that was said to have been deliberately concealed. Having accepted the well-established principle of company law and rule of attribution, namely that the unanimous decision of all the shareholders in a solvent company about anything which the company under its memorandum of association has power to do should be the decision of the company, Lord Briggs considered the difficulties inherent in extending that principle, so as to attribute to the company a sole shareholder's knowledge of, or the ability to discover, a breach of duty owed to the company by its directors:

> "[52] Section 14 of the Limitation Act is concerned not with the knowledge of claimants at a particular moment in time when taking some positive step, but rather with their knowledge, or their means of discovering the relevant facts, exercising due diligence, within some period after the occurrence of the breach giving rise to a cause of action.
>
> [53] This is not a concept easily applicable to a body of shareholders, who have no reason to be unanimous about matters concerning the company otherwise than when making some relevant decision which, by virtue of their unanimity, is treated by the law as an act of the company. It is one thing to say that shareholders making a unanimous decision may have the type of knowledge, constituting an intent that the thing resolved upon should be done, sufficient to be attributable to the company as mens rea. It is quite another thing to say that shareholders, who may or may not be unanimous when asked to make a decision about the company, should have attributed to the company the knowledge of some of them, or even all of them, about the facts

> relevant to a wrong done to the company by its directors, or the ability, with reasonable diligence, to discover them during some period of time.
>
> [54] Some, but not all, of these difficulties may be surmountable where the company has a sole shareholder, rather than a body of shareholders who may, or even frequently do, act unanimously. The problem that they may have different views from time to time plainly disappears, at least where the single shareholder is an individual or a corporation sole. But a number of other problems remain. First, the ordinary basis upon which the knowledge of directors or agents of a company is attributed to the company is that they owe a duty to the company to report relevant knowledge about its affairs. In sharp contrast, it is a cardinal principle of company law that shareholders do not owe such duties to their company. Shareholders are, in principle, entitled to leave their company to make its own inquiries about its affairs and, in particular, owe no duty of reasonable diligence to inform themselves about facts which might give rise to a claim by the company against wrongdoers, even against its directors.
>
> [55] The absence of any such duty was a powerful factor leading the courts below to reject the more generally based submissions about the attribution of shareholder knowledge. Mr Knox submitted that, in the limitation context, that absence of such a duty on even a sole shareholder made no difference, because the policy behind the Limitation Act was concerned with the prosecution of stale claims by those with an interest in them, rather than by those with a duty to sue. But this submission misses the point, because it fails to respect the separate identities of the company and its sole shareholder. True it is that the company need have no more than an interest in bringing proceedings for s.14 to be applicable to it. But s.14 applies to the company as potential claimant rather than to its shareholder. The attribution question is nonetheless powerfully affected, in this context negatively, by the absence of any duty of the shareholder to report its, or his, knowledge to the company.
>
> [56] For these and other reasons it is difficult to treat the attribution to a company of the knowledge of its own shareholders about facts relating to a claim by the company against its directors as a general rule of attribution. The general rule is that it is the knowledge of the company's directors that is attributed to it and, in appropriate cases, the knowledge of its agents."

1109. IBM's position is that it is only where a person acts on behalf of a company (whether as director, employee or some other form of agent) that their knowledge is capable of being attributed to the company. It is said that, as a matter of statutory construction, the knowledge of an agent is not attributable to the company for the purposes of section 32.

1110. Reliance is placed on the decision in: *Peco Arts Inc v Hazlitt Gallery* [1983] 1 WLR 1315 at 1326 E-G, in which Webster J said that the acts or omissions of an agent of a claimant are not to be attributed to the claimant for the purpose of ascertaining any lack of reasonable diligence, having regard to the express words in section 32(1):

> "References in this subsection to the defendant include references to the defendant's agent and to any person through whom the defendant claims and his agent.".

1111. The issue in that case was whether the claimant, acting through its art expert agent, had used reasonable diligence in not having had a painting authenticated and therefore, the judge in that case did not consider directly the issue whether the agent's knowledge should be attributed to the company. However, in *Allison v Horner* [2014] EWCA Civ 117, a broader application was expressly accepted as correct by Aikens LJ at [15]:

> "… in Peco Arts Inc v Hazlitt Gallery Ltd Webster J held that the acts or omissions of an agent of the claimant were not to be attributed to the claimant for the purposes of section 32(1). Thus knowledge of the deceit alleged on the part of a claimant's agent will be insufficient to start the limitation period running under section 32(1). Similarly, the fact that the claimant's agent could with reasonable diligence have discovered the alleged deceit does not start the limitation period running. I would accept this construction of section 32(1) for the reasons that Webster J gives at page 202G-H of the report."

1112. Further, approval of this approach was given in *Riyait v Dawett* [2018] EWCA Civ 593 at [34]-[37] per Sir Geoffrey Vos C (where the knowledge of the claimants' solicitors was not attributable for section 32 purposes):

> "[34] … In Horner this court held that only knowledge of the defendant's agents may be attributed under section 32, so that "knowledge of the deceit alleged on the part of a claimant's agent will be insufficient to start the limitation period running under section 32(1)", and "the fact that the claimant's agent could with reasonable diligence have discovered the alleged deceit does not start the limitation period running", and "the knowledge of agents of Mr Horner concerning the fraudulent representations is not to be attributed to him".
>
> [35] … It is clear that the qualifying words in section 32(1) apply to the whole of the subsection, so that the attribution of the knowledge of the defendant's agents applies in all parts of the subsection, just as the non-attribution of the knowledge of the claimants' agents applies in all parts of the subsection."

1113. The defendants submit that *Riyait*, *Horner* and *Peco Arts* are authority for the proposition that the knowledge of external (and arm's length) entities/individuals such as solicitors, tax consultants and valuation houses

cannot be attributable to a claimant for the purposes of section 32. They are not authority for the proposition that for the purposes of section 32 the knowledge of an individual (or indeed other group entities or group task forces) through whom the company is operating (whether as employee/director or in a role analogous to that) is not to be attributed to the company.

1114. The defendants' submission is correct. As the above authorities make clear, the court must ascertain the date by which IBM, and not its external agents, discovered, or could with reasonable diligence have discovered, the concealment for the purpose of section 32(1). However, IBM is a corporate entity, not a natural person. When considering the state of IBM's knowledge, the question that arises is whether the knowledge of any employee or agent should be attributed to it under the established rules of attribution set out in *Meridian* and *Bilta*. As the authorities make clear, the answer to the question whether an individual's knowledge should be attributed to a company depends on context and is fact sensitive.

1115. In this case I find that Mr Anzani was not a servant or agent of IBM when carrying out his investigations as part of Project Eiger and his knowledge should not be attributed to IBM. Firstly, Mr Anzani was employed by IBM Corp and has never been an officer or employee of IBM. Secondly, Mr Anzani had no involvement in the ICA between IBM and Winsopia, prior to the audit request in December 2020. Thirdly, Mr Anzani was not appointed by IBM to investigate LzLabs and the SDM; his work for Project Eiger was undertaken in his role as employee of IBM Corp. Fourthly, Mr Anzani had no obligation to disclose his investigations or report to IBM. His evidence was clear that it was his practice to keep secrets from everyone else in the IBM Group and that individuals were given information on a "need-to-know" basis. Fifthly, Mr Anzani's evidence, which I accept, was that he did not tell anyone at IBM of his knowledge about the link between LzLabs and Winsopia prior to 2020.

1116. Mr Anzani admitted in cross-examination that he acted on behalf of the whole IBM Group, including IBM, when orchestrating the message to be disseminated to existing or potential customers approached by LzLabs or interested in the SDM:

> "Q. You are the executive, aren't you, who is on point and has been on point since February of 2016, trying to protect IBM's interests against the competitive threat of LzLabs?
>
> A. Yes, I agree with that.
>
> Q. And you've done that on behalf of IBM UK and on behalf of IBM Corporation?
>
> A. Yeah, I have been on point to protect our interests across all of the IBM companies."

1117. A distinction must be drawn between the acts carried out by Mr Anzani on behalf of the wider IBM Group, to protect the Group's market share, and knowledge acquired by Mr Anzani during his investigation into LzLabs and the

SDM, through Project Eiger, as an employee of IBM Corp. That latter investigation was not carried out as employee or agent of IBM.

1118. For the above reasons:

i) Mr Anzani's knowledge is not to be attributed to IBM.

ii) Mr Anzani discovered the connection between LzLabs and Winsopia in July 2018 but did not impart that information to IBM until late 2020.

iii) No-one at IBM was aware of the connection between LzLabs and Winsopia prior to August 2020.

iv) There was nothing to put IBM on notice that there was such a connection between LzLabs and Winsopia so as to trigger an investigation into potential breaches of the ICA. Therefore, IBM, acting with reasonable diligence, could not have discovered the concealment prior to August 2020.

1119. There is a dispute between the parties as to the date on which any damage occurred so as to give rise to a cause of action in the tort of unlawful means conspiracy. The competing dates are 2016 (the defendants) and 2021 (IBM). The evidence before the court on the date of damage is sparse, not least because issues of quantum were not included in this trial. I do not consider that it is necessary to resolve this matter because on the facts it does not affect any limitation defence or the key findings on liability. The only claim that might be affected would be IBM's claim against Winsopia for unlawful means conspiracy, which might be caught by the contractual limitation period of two years, subject to the issue of dishonest or deliberate concealment. However, based on my Judgment dated 29 November 2023, when giving permission to IBM to amend to plead the unlawful conspiracy claim, I am satisfied that, as a result of late disclosure, IBM could not with reasonable diligence have discovered the concealment regarding the unlawful conspiracy claim prior to June 2023, when that claim was formulated.

1120. It follows that (subject to the disputed date of damage which, on the basis of my other findings, does not affect the outcome), the claims are not contractually time-barred or statute-barred for limitation.

Section X - Conclusions

1121. For the reasons set out above, I find as follows:

i) Winsopia was in breach of the ICA in respect of Items 1 to 30, 34, 35, 37, 38, 40, 41, 43, 45, 46, 49 and 50 as summarised in paragraphs [838] to [844] above.

ii) The above breaches of the ICA did not fall within the rights conferred by the Software Directive/the CDPA.

iii) LzLabs and Mr Moores unlawfully procured the above breaches of the ICA by Winsopia.

iv) The procurement claim against LzLabs UK fails.

v) Mr Rockmann and Mr Cresswell are entitled to rely on the principle in *Said v Butt* by way of defence and therefore are not liable for the tort of unlawfully procuring breaches of the ICA by Winsopia.

vi) LzLabs, Winsopia and Mr Moores are liable for the tort of unlawful means conspiracy.

vii) The unlawful means conspiracy claims against LzLabs UK, Mr Cresswell and Mr Rockmann fail.

viii) The audit request made by IBM was valid and Winsopia was in breach of the ICA in failing to comply with it.

ix) IBM validly terminated the ICA and associated agreements under clause 1.12.2 and/or 4.5.3; alternatively, IBM was entitled to treat the breaches as repudiatory and terminate at common law, as set out in paragraphs [980] to [987] above.

x) Clause 1.11.4 is an effective time bar provision as between IBM and Winsopia in respect of the claims against Winsopia but does not apply to the claims against the other defendants and does not apply to wrongdoing where there has been deliberate concealment until IBM discovered the concealment or with reasonable diligence could have done so.

xi) The defendants deliberately concealed the connection between Winsopia and LzLabs and/or Winsopia's breaches of the ICA for the purpose of section 32 of the Limitation Act 1980 and IBM could not with reasonable diligence have discovered the concealment prior to August 2020. IBM could not with reasonable diligence have discovered the concealment regarding the unlawful conspiracy claim prior to June 2023.

xii) The claims are not contractually time-barred or statute-barred for limitation.

1122. Following hand down of this judgment, the hearing will be adjourned to a date to be fixed for the purpose of any consequential matters, including the form of order, any declaratory, injunctive or other relief, future quantum determination or other disposal, any applications for costs or permission to appeal. Therefore, any relevant time limits are extended until the adjourned hearing or further order.