Submission to the Department of Communications And Arts

Copyright Modernisation in Australia: A Review of the *Copyright Act 1968* (Cth) with respect to fair use exceptions and software copyright

James Scheibner

PhD Candidate, University of Tasmania

Bachelor of Laws-Bachelor of Computer Science (Honours)

Author Information:

I am currently a final year PhD candidate in Law and Computer Science at the University of Tasmania under the supervision of Professor Dianne Nicol, Dr Jane Nielsen (both of the Centre for Law and Genetics at the University of Tasmania) and Associate Professor Michael Charleston (of the School of Physical Sciences at the University of Tasmania). My doctoral research focuses on the praxis between open source licensing norms and formal patent and copyright protection within academic computational biology research. For my thesis, I rely on a multi-methods approach relying on patent landscaping and comparative doctrinal analysis of US, EU, Australian and New Zealand patent and copyright law. I combined these doctrinal and quantitative analyses with semi-structured interviews of bioinformaticians, molecular biologists and computer scientists who were either listed as patent applicants, or were associated with institutes that had applied for patents but whom had also released open source software. From this multi-method analysis, I was able to provide recommendations as to both private ordering strategies for future open source bioinformatics projects and legislative reform strategies (which in part with respect to copyright and fair use are replicated in this submission).

I have already published parts of my empirical research as conference papers and journal articles. In addition, although I am a junior academic, I plan to conduct significant further research into the scope of patent, copyright and trade secrecy protection for emerging technologies, as well as the suitability of *sui generis* regimes such as database protection and regulatory data protection. In particular, I intend to use empirical methods (landscaping, semi-structured interviewing, surveying and experimental economics) to develop evidence based approaches for understanding the appropriate delineation of intellectual property protection to permit open innovation. Finally, I also intend to use the empirical approaches describe above to conduct research into the regulation of information communications technology services, particularly artificial intelligence and blockchain technology.

I confirm that the opinions presented in this submission are developed from my own research and do not represent the views of my institution or faculty.

1. Introduction

The Productivity Commission of Australia's 2016 report into *Intellectual Property Arrangements* recommended, amongst other things, following the advice of the Australian Law Reform Commission (ALRC) and replacing the current specified fair dealing exemptions in the *Copyright Act 1968* (Cth) with a general fair use test.[1] This reform would shift Australian copyright law towards the four part fair use test found within United States (US) copyright law.[2] Whilst the Productivity Commission focused heavily on the benefits of fair use for non-software related copyrighted works as well as the implications of patent law with respect to software inventions,[3] there remain unanswered questions as to the implications of a shift to fair use for software. These uncertainties have been somewhat exacerbated by the recent decision of the Federal Circuit Court of Appeals in *Oracle America Inc. v Google Inc.*, where the Federal Circuit first held that copyright extended to the functional elements of Application Programming Interfaces (APIs) developed by Oracle, and that Google's reuse of this functionality did not constitute fair use.[4]

This submission posits that the decision of the Federal Circuit has substantially reversed three decades of case law on copyright protection of functional aspects of software as well as the extent of the fair use exceptions for software copyright.[5] In addition, the Federal Circuit's decision, read broadly, has the potential to create an 'anti-commons' effect with respect to object oriented software development that arguably outweighs the reported 'anti-commons' potential associated with software patents.[6] Consequently, although the author supports the ALRC's recommendation and the Productivity Commission's recommendation that the fair dealing exceptions in the *Copyright Act 1968* (Cth), this submission argues that further consideration should be given to the consequences of Australian courts following the decision in *Oracle America Inc. v Google Inc*. In particular, this submission argues that *Oracle America Inc. v Google Inc* should be contained to its facts and that Australian courts, in applying a novel fair use test in the *Copyright Act 1968* (Cth), should avoid construing such a test so narrowly so as to exclude legitimate reuse of non-copyright protected functional concepts.

1 Australian Law Reform Commission, *Copyright and the Digital Economy*, Report No 122 (2013).

2 17 U.S. Code § 106 - *Exclusive rights in copyrighted works*.

3 Productivity Commission, *Intellectual Property Arrangements*, Report No 78 (December 2016), Chapter Six.

4 *Oracle America Inc. v Google* 750 F.3d 1339 (Fed. Cir. 2014); *Oracle America Inc v. Google Inc.* No. 17-1118 (Fed. Cir. 2018).

5 Pamela Samuelson, 'Copyrightability of Java APIs Revisited' (2015) 58(3) *Commun. ACM* 22. See *Apple Computer Inc v. Franklin Computer Corp.* 714 F.2d 1240 (3d Cir. 1983) as an example of previously overruled case law regarding the scope of copyright protection for the functional aspects of software.

6 Michael Noel and Mark Schankerman, 'Strategic Patenting and Software Innovation' (2013) 61(3) *The Journal of Industrial Economics* 481, 485; Wen Wen and Chris Forman, 'Do Patent Commons and Standards-Setting Organizations Help Navigate Patent Thickets?' (2016) 59(5) *Commun. ACM* 42; Clark D Asay, 'Software's Copyright Anticommons' (2016) 66(2) *Emory Law Journal* 265, 267.

To this end, this submission contains four parts. Part 2 provides an overview of the technical aspects of modern software development, as well as an explanation as to why modern software development practices (both in proprietary and open source software development) are dependent on the reuse of components. Part 2 also provides an economic rationalisation for limiting the scope of copyright protection to the literary aspects of software (as opposed to the functional aspects). Part 3 then considers the legislative diffusion of software on both a global and international level, with an overview of how copyright protection became the de facto standard in the US and spread to other jurisdictions over time. Part 4 provides an overview of US case law leading up to the decision in *Oracle America Inc v Google Inc*, with a focus on the interplay between the decisions of the District Court of California and of the Court of Appeal of the Federal Circuit. Part 5 then considers the economic and technical implications of the *Oracle America Inc v Google Inc* decision, before concluding with an explanation why, at the very least, Australian courts should treat the decision in *Oracle America Inc v Google Inc* as being contained to its facts.

2. Technical Background and Economic Justifications for Software Copyright

To understand the scope of copyright protection for software, it is first necessary to understand the different elements of a software program and how these different elements have evolved over time. The first software systems were procedural in nature but were also designed specifically for a particular hardware platform, with software containing references to particular memory addresses and transistors for sending and storing data. Whilst developers can produce highly efficient software with this style of machine level programming (particularly for embedded hardware), this style of programming also makes it difficult to translate this software code to different hardware platforms and identify errors.[7] Accordingly, programmers replaced machine level functions with mnemonic functions and memory addresses with symbolic labels to introduce a degree of abstraction into software development.[8] Even with these ‘assembly level’ languages, programmers still relied upon hardware specific syntax, which increased the barrier for translation of software code to different hardware platforms.

As a result, programmers began to develop high level languages that completed abstracted different computational functions performed by hardware. These languages include FORTRAN (which was developed by John Backus and IBM in 1957), ALGOL (which was developed at ETH Zurich in

7 J Glenn Brookshear, *Computer Science: An Overview* (Pearson/Addison Wesley, 2009) 240.

8 Joey Paquet and Serguei A Mokhov, ‘Comparative Studies of Programming Languages; Course Lecture Notes’ [2010] *arXiv:1007.2123 [cs]* 21–22 <http://arxiv.org/abs/1007.2123>.

1958 by a consortium of EU and US researchers for artificial intelligence research) and LISP (a functional language developed by John McCarthy at MIT in 1960).[9] Software written in these languages could then be translated into machine readable code using an assembler program via a process known as compilation.[10] In addition to being hardware independent, all of these high level languages operated functionally rather than procedurally; that is, rather than performing a sequence of tasks in a sequential order, high level languages could be used to write sub-routines to perform tasks in a non-sequential order.[11] In turn, these sub-routines formed the core of object oriented programming, which refers to a set of software engineering techniques where different subroutines are separated from one another so that they can be reused in different parts of the program.[12] The first true object oriented programming language was the C programming language, which was specifically designed to allow programmers to develop software in a modular, architectural fashion.

These modular approaches to software design influenced Alan Kay, who invented object oriented programming techniques and introduced them into his language Sketchpad in 1970.[13] Object oriented programming refers to a programming style where different objects are represented using properties and functions.[14] Object oriented programming languages are designed to encourage developers to use encapsulation and inheritance. Encapsulation refers to a design technique where, primarily for security reasons, the source code defining certain operations is hidden and only the functions themselves are made available for other software programs to use. Inheritance refers to a programming technique where certain generic methods and objects are first defined, and then subsets of these different objects are then defined as extensions of these datatypes.[15] Programmers adopting object oriented programming approaches have led to specialised programs containing only references to particular functions, otherwise known as libraries and Application Programming Interfaces (APIs), forming an important part of modern software development.[16]

From this technical overview, the following observation can be drawn about different programming languages. Firstly, high level or source code languages, which are human readable and have a high level of abstraction, are compiled into low level assembly or machine code languages, which are

9 Marc Nerlove, ‘Programming Languages: A Short History for Economists’ (2004) 29(1–3) *Journal of Economic & Social Measurement* 189, 191.

10 Paquet and Mokhov, above n 8, 14.

11 Jean E Sammet, *Programming Languages: History and Fundamentals* (Prentice-Hall, Inc., 1969) 14–19.

12 Peter Naur and Brian Randell, `Software Engineering' (Report of a Conference Sponsored by the NATO Science Committee, NATO Scientific Affairs Division, Garmisch, 7-11 October 1968)

13 Daniel Saunders and Paul Thagard, ‘Creativity in Computer Science’ in James C Kaufman and John Baer (eds), *Creativity Across Domains: Faces of the Muse* (Psychology Press, 2005) 158, 170.

14 Justin Joque, ‘The Invention of the Object: Object Orientation and the Philosophical Development of Programming Languages’ (2016) 29(4) *Philosophy & Technology* 335, 338.

15 Stephen R Schach, *Classical and Object-Oriented Software Engineering with UML and C++* (WCB/McGraw-Hill, 1999) 201–2.

16 Jean-François Blanchette, ‘A Material History of Bits’ 62(6) *Journal of the American Society for Information Science and Technology* 1042, 1046.

referred to by the neologism object code and have a low degree of abstraction. This distinction foreshadows the initial distinctions between 'source code' and 'object code' as literary and functional works respectively in determining the scope of copyright law.[17] Secondly, the influences of object oriented programming on software development have meant that programmers now rely heavily on reusing different functions in their software through the use of libraries and APIs.[18] This style of development is particularly important in specialised fields such as scientific software development, where researchers frequently rely on and combine different libraries to perform different analytical tasks.[19]

Object oriented programming techniques have consequently become a key feature of both proprietary and open source development strategies. In this regard, proprietary or 'closed source' development strategies refer to software where only the object code or the executable part of the software is made available to users, whilst the source code is protected by the developer. In contrast, 'open source' development strategies refer to development strategies where both the source and object code are made available to users so that they can modify and reuse that software.[20] Although these two approaches to software development are presented as alternatives, in practice proprietary and open source software are often used to complement each other in the development of a larger software system (where either proprietary or open source licensing permits this combination).[21] However, as Asay notes, this reuse means that numerous copyright interests may be present in any single software package or indeed within a whole software ecosystem. In the absence of cross licensing between parties, or the use of compatible open source licences to allow for reuse of these components, there needs to be a policy lever to permit reuse of non-copyright protected components, [22]

Accordingly, there are a number of competing economic considerations involved with respect to copyright protection for software. On the one hand, the primary purpose of copyright law is utilitarian in nature; that is, to provide creators with an incentive to publish and recoup the sunken costs of producing an expressive work.[23] To this end, copyright protection should (and does by

17 Pamela Samuelson, 'Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement' (2016) 31(2) *Berkeley Technology Law Journal* 1215, 1296.

18 Jesus M Gonzalez-Barahona et al, 'Macro-Level Software Evolution: A Case Study of a Large Software Compilation' (2009) 14(3) *Empirical Software Engineering* 262.

19 Likit Preeyanon, AlexisBlack Pyrkosz and CTitus Brown, 'Reproducible Bioinformatics Research for Biologists' in Victoria Stodden, Roger D Peng and Friedrich Leisch (eds), *Implementing Reproducible Research* (Chapman and Hall/CRC, 2014) 185 <http://www.crcnetbase.com/doi/abs/10.1201/b16868-10>.

20 Kevin Crowston et al, 'Free/Libre Open-Source Software Development: What We Know and What We Do Not Know' (2012) 44(2) *ACM Computing Surveys* 7.

21 Daniel M German, Massimillano Di Penta and Julius Davies, 'Understanding and Auditing the Licensing of Open Source Software Distributions' in *2010 IEEE 18th International Conference on Program Comprehension* (2010) 84.

22 Asay, 'Software's Copyright Anticommons', above n 6, 307.

23 Thorsten Käseberg, *Intellectual Property, Antitrust and Cumulative Innovation in the EU and the US* (Bloomsbury Publishing, 2012) 12.

virtue of international law) vest in computer programs to protect developers from both software piracy (the direct copying of copyrighted works for personal use) and imitation (by other software developers).[24] On the other hand, as discussed previously, object oriented programming has created a software ecosystem which is heavily dependent on the reuse of functional software packages, particularly libraries and APIs. This interconnected design approach holds particularly true for platform based computing services such as mobile communication and cloud computing platforms, where APIs and libraries constitute infrastructure that helps maintain interoperability for such platforms.[25] In addition, interconnected software design (as well as data sharing) has become an increasingly important feature of computationally scientific research, as it allows researchers to share source code and data rather than have to repeat experiments and software development.[26] All of these factors are important economic factors to consider when not only determining the proper scope of copyright protection for software but also the extent of a reformed fair use provision for software reengineering.

3. The Origins and Spread of Copyright Law for Software

As Samuelson observes, arguments about copyright protection for software first began in the 1960s and 1970s, where lawyers and economists debated over whether what form (if any) of legal protection was or should be available for computer programs.[27] In the US, because the American software industry was flourishing without the support of copyright protection (due to most software being custom designed for particular hardware platforms), lawmakers deemed it unnecessary to extend copyright protection to software.[28] In addition, the US Copyright Office was initially strongly opposed to extending copyright protection to software, rejecting early applications for registration, due to longstanding US precedent opposing the grant of copyright protection to functional works.[29] However, over time it became apparent that this decision to exclude copyright

24 Eleonora Rosati, 'The Idea/Expression Dichotomy: Friend or Foe?' in Richard Watt (ed), *Handbook on the Economics of Copyright: A Guide for Students and Teachers* (Edward Elgar Publishing, 2014) 51, 62–8; Stan J Liebowitz, 'The Impacts of Internet Piracy' in Richard Watt (ed), *Handbook on the Economics of Copyright: A Guide for Students and Teachers* (Edward Elgar Publishing, 2014) 225, 261.

25 Jean-François Blanchette, 'Computing As If Infrastructure Mattered' (2012) 55(10) *Commun. ACM* 32, 33; Geoffrey Parker, Marshall Van Alstyne and Xiaoyue Jiang, 'Platform Ecosystems: How Developers Invert the Firm' (2017) 41(1) *MIS Quarterly* 255, 256.

26 Paul F Uhlir and Peter Schröder, 'Open Data for Global Science' (2007) 6 *Data Science Journal* OD36, OD47.

27 Pamela Samuelson, 'A Square Peg in a Round Hole? Copyright Protection for Computer Programs' in Brad Sherman and Wiseman (eds), *Copyright and the Challenge of the New* (2012) 251 <http://works.bepress.com/pamela_samuelson/258>.

28 Stephen Breyer, 'The Uneasy Case for Copyright: A Study of Copyright in Books, Photocopies, and Computer Programs' (1970) 84(2) *Harvard Law Review* 281, 347–8; Pamela Samuelson, 'The Uneasy Case for Software Copyrights Revisited' (2010) 79(6) *George Washington Law Review* 1746, 1746–7.

29 Whilst international copyright treaties do not impose a formalities requirement for copyright protection, the US has adopted a system where copyright holders can chose to register their copyright.

protection for software was not congruent with the evolving economic reality of software development.

For example, because until 1978 International Business Machines (IBM) released all their software for their mainframe computers for free, Fujitsu, a Japanese company, were able to reverse engineer this software and produce mainframe machines that were backwards compatible with IBM's software. Because these machines were cheaper than IBM's machines, Fujitsu were able to undermine IBM's domestic and international sales.[30] Although Fujitsu and IBM eventually reached a settlement, the case underlined the need for copyright support for software in the US.[31] This incentive to guarantee copyright protection was exacerbated by the fact that both the EU and Japan were considering introducing a form of *sui generis* intellectual property right for software that would have carried a significantly shorter term of protection than copyright protection.[32]

Accordingly, the US Congress convened a National Commission on New Uses of Copyrighted Works (CONTU) in 1974, which, in contrast to its EU and Japanese equivalents, ultimately concluded that the US *should* offer copyright protection for software, and that courts should ultimately be left to decide what aspects of software were protected by copyright.[33] CONTU's recommendations resulted in the amendment of the US Copyright Act in 1980 to include a specific definition of a computer program as a 'set of statements or instructions to be used directly or indirectly to bring about a certain result'. The amendment also included specific exemptions for copying, creating backup copies of, and reselling computer programs.[34]

In addition to extending copyright protection to software, an amendment to US copyright law flowing from CONTU was to formalise the boundaries of the fair use doctrine with respect to software copyright. As discussed by Victoria Stodden and Vincent Carey, the fair use doctrine is essentially a 'safety valve that permits use of copyrighted material in certain settings where approval has not been obtained from the copyright holder'.[35] Although the doctrine of fair use was originally imported from the United Kingdom (UK) as a common law doctrine, it was eventually

30 Pamela Samuelson, 'IBM's Pragmatic Embrace of Open Source' (2006) 49(10) *Commun. ACM* 21, 23. See also *IBM v Fujitsu* No. 13T-117-0636-85 (American Arbitration Association Commercial Arbitration Tribunal 1987).

31 Anita Stork, 'The Use of Arbitration in Copyright Disputes: IBM v. Fujitsu' (1988) 3 *High Technology Law Journal* 241, 260–1; Marie Anchordoguy, 'Japan's Software Industry: A Failure of Institutions?' (2000) 29(3) *Research Policy* 391, 394.

32 Samuelson, 'A Square Peg in a Round Hole?', above n 27, 258; Giorgio Fabio Colombo and Matteo Dragoni, 'The Legal Protection of Software in Japan—An Original Model?' in Giuseppe Bellantuono and Fabiano Teodoro Lara (eds), *Law, Development and Innovation* (Springer, Cham, 2016) 67, 76 <https://link.springer.com/chapter/10.1007/978-3-319-13311-9_5>.

33 Gerardo Con Díaz, 'The Text in the Machine: American Copyright Law and the Many Natures of Software, 1974–1978' (2016) 57(4) *Technology and Culture* 753, 756, 777.

34 Computer Software Copyright Act of 1980 Pub. L. No. 96-517, 94 Stat. 3015 1980.

35 Vincent J Carey and Victoria Stodden, 'Reproducible Research Concepts and Tools for Cancer Bioinformatics' in *Biomedical Informatics for Cancer Research* (Springer, Boston, MA, 2010) 149, 168 <https://link.springer.com/chapter/10.1007/978-1-4419-5714-6_8>.

codified in the US via the *Copyright Act* of 1976.[36] This statutory doctrine includes a four factor test for weighing whether alleged infringing use amounts to fair use:

1. the purpose and character of the use, including whether such use is of a commercial nature or for nonprofit educational purposes;

2. the nature of the copyrighted work;

3. the amount and substantiality of the portion used in relation to the copyrighted work as a whole; and,

4. the effect of the use of the potential marker for or value of the copyrighted work.

Initially, the market harm factor was considered the predominate factor in determining whether there had been copyright infringement; in other words, whether there had been a transformative use of the copyrighted material.[37] However, with respect to copyright infringement for software, CONTU was left with a troubling determination as to how fair use should be best moulded to fit the unique nature of software. Although CONTU eventually reached the conclusion that computer programs should be subject to the sale rules of fair use and systematic copying, the commission's pronouncements have been contradicted by courts, which will be discussed in Part 4 below.[38]

After copyright became the formal intellectual property right for software, there was a massive surge in the number of software developers registering copyright in software programs in the US, where the copyright was accepting over five thousand yearly registrations by the mid 1980s.[39] Due to the economic pressure exerted by the US as a both a major trading partner for Japan and the EU and a major supplier of software, the latter two jurisdictions abandoned their *sui generis* proposals and introduced copyright protection for software. This development was then followed by software being formalised as copyrightable subject matter in the TRIPS Agreement.[40] As a result, signatories to the TRIPS Agreement (including Australia) were compelled to introduce copyright protection for software.

The political influence that the US (as the home of many of the world's top software development companies, as well as computer science universities and research facilities) had in driving and

36 Copyright Act 1976 (US) § 107; Matthew Sag, 'The Prehistory of Fair Use' (2010) 76(4) *Brooklyn Law Review* 1371–1412 1411.

37 Pierre N Leval, 'Toward a Fair Use Standard Commentaries' (1989) 103(5) *Harvard Law Review* 1105, 1111; Jane C Ginsburg, 'Conflicts of Copyright Ownership between Authors and Owners of Orignial Artworks: An Essay in Comparative and International Private Law' (1992) 17(4) *Columbia-VLA Journal of Law & the Arts* 395, 401.

38 Arthur R Miller, 'Copyright Protection for Computer Programs, Databases, and Computer-Generated Works: Is Anything New Since CONTU' (1992) 106(5) *Harvard Law Review* 977, 1022.

39 Díaz, above n 33, 770.

40 *Marrakesh Agreement Establishing the World Trade Organization, annex IC, The Agreement on Trade Related Aspects of Intellectual Property Rights* ('TRIPS Agreement'), opened for signature 15th April 1994, 1867 UNTS 3 (entered into force 1st January 1995). article 9(2), article 10(2).

formalising software copyright protection was two fold. Firstly, foreign inventors were driven to seek copyright protection in the US if they wished to release their software to the US market so as to keep pace with domestic developers.[41] Secondly, to prevent the leakage of software engineering know how into overseas markets where copyright protection was not available, the US exerted bilateral pressure on other nations to standardise the regulation of software copyright, leading to the diffusion of copyright protection into other countries.[42] The next section will discuss how US courts have resolved the questions of copyright protection and fair use with respect to software, leading to the decision of the Federal Circuit court in *Oracle America Inc v Google Inc*.

4. Fair Use and Software Copyright Case Law in the US – comparing *Apple* to *Oracle*

4.1 *Apple v Microsoft* and the 'SSO test' for software copyright infringement

Despite the explicit extension of copyright to software in US law, there was still considerable early uncertainty as to the extent of software copyright, particularly with respect to what elements of a computer program were functional and therefore unprotected by copyright law.[43] The first case in the US considering the reformed copyright law, *Data Cash Systems Inc v JS & A,* featured the 7th Circuit Court of Appeals rejecting extending copyright protection to object code on read only memory on the grounds that the 'structure, sequence and operation' of computer programs was purely function.[44] However, in the subsequent case of *Apple v Franklin*, the 3rd Circuit Court of Appeals held that copyright was available for both the source code and the object code of a software application. In addition, the Court of Appeals rejected the defendant's argument that reverse engineer the plaintiff's software to produce compatible hardware amounted to fair use.[45]

The 3rd Circuit Court of Appeals extended this protection further to the 'non-literal elements' of a computer program in *Whelan and Associates v Jaslow Dental Laboratory Inc* ('*Whelan v Jaslow*'), holding that a dental practice management software package that was designed on a competing product amounted to copyright infringement.[46] Samuelson and Lemley acknowledge that the Court of Appeals decision with respect to copyright infringement was correct, insofar as exact copying of object code amounting to copyright infringement was concerned. However, they argue that it also created a troubling precedent, which was subsequently reinforced by *Whelan v Jaslow*, of

41 Kenneth C Shadlen, Andrew Schrank and Marcus J Kurtz, 'The Political Economy of Intellectual Property Protection: The Case of Software' (2005) 49(1) *International Studies Quarterly* 45, 52.

42 Laurence Diver, 'Would the Current Ambiguities within the Legal Protection of Software Be Solved by the Creation of a Sui Generis Property Right for Computer Programs?' (2008) 3(2) *Journal of Intellectual Property Law & Practice* 125, 129; Keith Maskus, 'The New Globalisation of Intellectual Property Rights: What's New This Time?' (2014) 54(3) *Australian Economic History Review* 262, 264.

43 *Baker v. Selden* 101 U.S. 99 (1880);

44 *Data Cash Systems, Inc. v. Js&a Group, Inc.* 480 480 F. Supp. 1063 (1979).

45 *Apple Computer, Inc. v. Franklin Computer Corp.* 714 F. 2d 1240 (3d Cir. 1983).

46 *Whelan Associates, Inc. v. Jaslow Dental Laboratory, Inc.* 797 F.2d 1222 (3d Cir. 1986), 1234, 1240.

broadening copyright protection to include the functional and graphical features of software. In addition, both *Apple v Franklin* and *Whelan v Jaslow* placed the copying of object code for the purposes of developing interoperable software outside the scope of the fair use test.[47] Emboldened by their success in *Apple v Franklin*, Apple then pursued Microsoft, who had modelled the graphical user interface for their new Windows operating system on Apple's own operating system, for copying the 'look and feel' of Apple's graphical user interface.[48]

However, in both of the *Apple v Microsoft* cases, as well as the simultaneous *Computer Associates v Altai* series of cases, the 9th Circuit and 2nd Circuit Appeal Courts respectively rejected the extension of copyright to the 'structure, sequence and organisation of computer programs'. In *Computer Associates v Altai*, the respondent's code was misappropriated by a former employee, who had then incorporated this code into the applicant's software. The respondent argued that despite the purge of the tainted code, there were still substantial similarities between the two code bases (and therefore there was copying of the structure of a computer program.[49] The 2nd Circuit Appeals Court in *Computer Associates v Altai* declined to follow *Whelan* and replaced the test from *Apple v Franklin* with a three part test for determining copyright infringement in software that had been adopted by district courts.[50] The first step of this three part test involves developing a set of abstractions from the originally infringed software; that is, breaking down the program into its novel ideas and expressions as well as the parts that have been adopted from the public domain.[51] The second step of this test involves the court filtering out all elements which are unprotected under copyright law and removing them from the consideration of the court. The final step of this test involves comparing the allegedly infringing code with the original code to determine the infringement.

Applying this test to the facts in *Computer Associates v Altai*, the 2nd Circuit held that infringement had not been established, as the wayward employee had not copied the key part of the protected source code but had only copied organisational flowcharts.[52] With respect to the parts that had been copied, the 2nd Circuit referred to a recent Supreme Court case, *Feist Publications Inc v Rural Telephone Service Inc*, that *de minimis* copying was not indicative of copyright infringement and therefore concluded that there was no infringement on the facts. Likewise, the 9th Circuit held on appeal that the elements of Apple's graphical user interface that were copied by Microsoft were functional and therefore did not amount to copyright infringement.[53]

47 Mark A Lemley, 'Convergence in the Law of Software Copyright' (1995) 10 *High Technology Law Journal* 1, 7; Samuelson, 'A Square Peg in a Round Hole?', above n 27, 260–1.

48 *Apple Computer, Inc. v. Microsoft Corp.* 799 F.Supp. 1006 (N.D. Cal. 1992); *Apple Computer, Inc. v. Microsoft Corp.* 35 F.3d 1435 (9th Cir. 1994).

49 *Computer Associates Intern., Inc. v. Altai, Inc.* 982 F.2d 693, 701 (2d Cir. 1992).

50 *Computer Associates Intern., Inc. v. Altai, Inc.* 982 F.2d 693, 706 (2d Cir. 1992).

51 *Computer Associates Intern., Inc. v. Altai, Inc.* 982 F.2d 693, 707 (2d Cir. 1992).

52 *Computer Associates Intern., Inc. v. Altai, Inc.* 982 F.2d 693, 715-6 (2d Cir. 1992).

53 *Apple Computer, Inc. v. Microsoft Corp.* 35 F.3d 1435, 1446 (9th Cir. 1994).

The 9th Circuit then considered the question of reverse engineering and functionality (particularly within the context of fair use) in the *Sega Enterprises Ltd v Accolade Inc* case, which concerned the Sega Genesis video game console. Accolade, a video game developer, had attempted to reverse engineer a Sega Genesis console so that they could produce games that would run on its operating system without having to pay exorbitant licensing fees to Sega.[54] This reverse engineering process involved Accolade copying and decompiling the object code on a Sega console and then rewriting the source code for their own software so that the resultant code was compatible with the Sega console. Sega sued Accolade for copyright infringement, with Accolade defending their actions on the grounds that their decompilation amounted to fair use.

The 9th Circuit confirmed that in this instance, although Accolade had breached copyright by decompiling the object code, the replication of functionality amounted to fair use under the amended US copyright law.[55] In applying the four fair use factors, the 9th Circuit noted that Accolade's decompilation as an act of copying was limited to the functional aspects of software. Further, the 9th Circuit held that despite the fact that Accolade were copying source code in direct competition with Sega (who were also writing computer games for the Sega Genesis), this copying was sufficiently transformative to amount to fair use.[56]

This application of the fair use defence can be contrasted with the contemporaneous case of *Atari Games Corp v Nintendo America Inc* where Atari requested a copy of Nintendo's source code from the US Copyright Office to prepare for alleged copyright infringement action, but were in fact using this source code in order to assist with the decompilation of source code.[57] Because Atari then used this source code to create a competing platform, the Federal Circuit held that there was no fair use in the circumstances.[58] Nevertheless, both *Sega Enterprises Ltd v Accolade Ltd* and *Atari Games Corp v Nintendo America Inc* established a clear precedent on fair use for software reverse engineering that remained undisturbed by both case law and further copyright reform (such as the *Digital Millennium Copyright Act*)[59] until the *Oracle v Google* series of cases.

54 *Sega Enterprises Ltd. v. Accolade, Inc.* 977 F.2d 1510, 1514-5 (9th Cir. 1992).
55 *Sega Enterprises Ltd. v. Accolade, Inc.* 977 F.2d 1510, 1519-20 (9th Cir. 1992).
56 *Sega Enterprises Ltd. v. Accolade, Inc.* 977 F.2d 1510, 1526 (9th Cir. 1992).
57 *Atari Games Corp v Nintendo of America Inc* 975 F.2d 832, 843 (Fed. Cir. 1992).
58 *Atari Games Corp v Nintendo of America Inc* 975 F.2d 832, 844 (Fed. Cir. 1992).
59 Joseph P Liu, 'The DMCA and the Regulation of Scientific Research' (2003) 18(2) *Berkeley Technology Law Journal* 501, 509. In this article, Professor Liu argues that the *Digital Millennium Copyright Act* could place legitimate boundaries on the reverse engineering of software for academic cryptographic research. Likewise, the DeCSS litigation appears to have placed some limitations on the distribution of DeCSS software. This submission will address the potential consequences

4.2 *Oracle v Google* and a return to the *Apple* test for infringement

The genesis of the *Oracle v Google* dispute can be found within the development of Google's Android mobile operating system platform. Although the kernel for Android is built on GNU/Linux, an open source operating system, Google designed Android so that third party developers could write Android applications using Java, a programming language that at the time was published by Sun Microsystems and was the 'industry standard' for mobile application development.[60] Sun had released a number of Java APIs, some under the GNU General Public Licence agreement (a free and open source software licence requiring relicensing of derivative works), others under proprietary licences which mandated that developers only use Sun's version of Java and not create any derivatives. The purpose of this licensing requirement was to ensure application interoperability and prevent multiple versions of Java APIs from being released (indeed, Sun had already been engaged in lengthy litigation with Microsoft to prevent Microsoft from releasing their own version of Java contrary to their licence agreement).[61] However, unlike the litigation with Microsoft, where Sun and Microsoft had a pre-existing licence agreement, negotiations between Sun and Google to cross licence these proprietary Java APIs had stalled. Accordingly, Google made the decision to reimplement the functionality of 37 of these Java APIs, or approximately 600 classes or 6,000 methods, into their own Android operating system.[62] Specifically, Google copied class and method inheritance, file structure and 'structure, sequence and organisation' but wrote their own implementations each of these features.

Whilst Sun was aware of Google's reimplementation, they chose not to sue Google. However, this situation changed following Sun being bought out by Oracle in 2010, who immediately brought action against Google for copyright and patent infringement. Crucially, the decision to seek patent infringement led to the case being redirected towards the Federal Circuit Court of Appeals, which is a specialised US court established for hearing patent appeal matters (as opposed to copyright matters).[63] Oracle argued that despite the fact that the main part of the copied code was the 'structure, sequence and organisation' of the Java APIs, the APIs in question still required a substantial degree of creativity to develop.[64] In response, Google denied that copyright vested in the

60 Asay, 'Software's Copyright Anticommons', above n 6, 304.

61 *Sun Microsystems Inc. v Microsoft Corporation* (No. C 97-20884RMWPVT, 2000 WL 33223397, May 2000). For a further discussion of this litigation please refer to Victoria Nemiah, 'License and Registration, Please: Using Copyright Conditions to Protect Free/Open Source Software' (2013) 3(2) *New York University Journal of Intellectual Property and Entertainment Law* 358, 368.

62 *Oracle America Inc. v Google* 750 F.3d 1339, 1351 (Fed. Cir. 2014).

63 Rochelle Cooper Dreyfuss, 'The Federal Circuit: A Case Study in Specialized Courts' (1989) 64(1) *New York University Law Review* 1, 3.

64 Pamela Samuelson, 'Oracle V. Google: Are APIs Copyrightable?' (2012) 55(11) *Commun. ACM* 25, 26.

APIs as they were largely functional in nature, and even if they were copyrighted material, Google's reimplementation was sufficiently transformative to amount to fair use.[65]

At first instance, Justice Aslup of the Northern District Court of California dismissed Oracle's copyright claim, noting that 'as long as the specific code is different, anyone is free under the *Copyright Act* to write his or her own method to carry out the exact same function or specification'. In particular, Judge Aslup was influenced by the fact that only three percent of the lines of code in Google's reimplemented Java APIs were shared with the original Sun Java APIs, and that the replicated functionality was analogous to the factual compilations, which were decisively ruled not to constitute copyrightable materials in *Feist Publications v Rural Telephone Service*.[66] However, on appeal, the Federal Circuit upheld Oracle's copyright claim, holding that Judge Aslup erred in finding that the functionality of the APIs was not protected by copyright. Crucially, the Federal Circuit held that because there were multiple ways of implementing each of the functions contained within each API, copyright vested in these methods and in the 'structure, sequence and organisation' of these APIs even though they were functional in nature.[67] Accordingly, the Federal Circuit held that Google's exact replication of these elements amounted to copyright infringement, because there were alternative means of expression.[68]

Google attempted to appeal to the Supreme Court, who refused leave to appeal on the question of copyright infringement but remitted the matter to the District Court to hear the question of whether Google's reimplementation was covered by fair use. Judge Aslup, once again hearing the matter, directed a jury that because Google had only reimplemented the APIs that it needed for mobile application development (and because Sun's APIs were originally developed for Java applications on desktop and laptop computers), Google's reverse engineering was sufficiently transformative to amount to fair use.[69] Oracle again appealed to the Federal Circuit, which again reversed Judge Aslup's decision on the grounds that Google's reimplementation was not protected by fair use. In particular, despite the fact that Google released their own reverse engineered APIs without charge, the Federal Circuit reasoned that these APIs formed part of a broader software ecosystem which allowed directly to compete with Oracle in the market for mobile services.[70] Accordingly, Google's use was not sufficiently transformative to amount to fair use.[71] The decision has been again remanded to the Northern District Court of California to determine damages, although given the

65 *Oracle America, Inc. v. Google Inc.* 872 F.Supp.2d 974, 975-6 (Dist. Court, ND California 2012).
66 *Oracle America, Inc. v. Google Inc.* 872 F.Supp.2d 974, 979, 992 (Dist. Court, ND California 2012).
67 *Oracle America Inc. v Google* 750 F.3d 1339, 1367 (Fed. Cir. 2014).
68 *Oracle America Inc. v Google* 750 F.3d 1339, 1361 (Fed. Cir. 2014).
69 *Oracle America, Inc. v. Google Inc.* (No. C 10-03561 WHA, May 2016), 7-10.
70 *Oracle America, Inc. v. Google Inc.* (No. 17-1118, Fed. Cir. 2018), 34-6.
71 *Oracle America, Inc. v. Google Inc.* (No. 17-1118, Fed. Cir. 2018), 36.

potential impact on Google’s Android ecosystem it is highly likely that Google will again attempt to appeal to the Supreme Court.

The final section of this submission will consider the implications of the *Oracle v Google* decision for software copyright in the US, and will then provide recommendations as to how legislators should draft an amended fair use defence under Australian copyright law.

5. Implications of *Oracle* for an amended fair use defence in the *Copyright Act 1968* (Cth)

5.1 The implications of *Oracle v Google* for software copyright protection in the US

As discussed in the previous section, the decision in *Oracle v Google* has effectively reversed three decades of US law on both the extension of copyright to the functional elements of computer programs and the application of fair use to the reverse engineering of computer software.[72] In part, Menell argues that the decision to send the matter to the Federal Circuit (which is established for the purpose of hearing patent litigation) due to the initial patent infringement claims by Oracle meant that the Federal Circuit arrived at a significantly different interpretation of 9th Circuit copyright precedent than another appellate court.[73] Further, as Menell argues, because the Federal Circuit does not technically have exclusive appellate jurisdiction with respect to copyright claims, district courts are left with a dilemma on whether to follow existing 9th Circuit jurisprudence or the Federal Circuit’s interpretation of this precedent.[74]

Accordingly, the decision of the Federal Circuit to deviate considerably from existing copyright precedent has attracted significant criticism. Asay, in citing Sag’s study that established that transformative use is the most important factor in determining fair use,[75] argues that the precedent established by *Oracle v Google* means that reimplementation of functionality, even with entirely different source code, does not amount to fair use. Asay argues that this decision ignores the fact that the vast majority of software innovation involves at least some degree of replication of functionality to perform pre ordained functions such as assigning values to particular variables.[76] Proponents of the *Oracle v Google* precedent might argue the circumstances are factually unique because of Google’s dominance in both the search engine and mobile development market, which provides Google with a greater capacity to exclude other companies from those markets through

72 Samuelson, ‘Copyrightability of Java APIs Revisited’, above n 5, 24.
73 Peter S Menell, ‘API Copyrightability Bleak House: Unraveling and Repairing the Oracle v. Google Jurisdictional Mess’ (2016) 31(3) *Berkeley Technology Law Journal* 1515, 1518.
74 Ibid 1519.
75 Matthew Sag, ‘Predicting Fair Use’ (2012) 73(1) *Ohio State Law Journal* 47, 55.
76 Clark D Asay, ‘Transformative Use in Software’ (2017) 70(1) *Stanford Law Review Online* 9, 14.

sheer market dominance and first mover advantage than other potential software developers.[77] However, Gratz and Lemley note that the decision from *Oracle v* Google, taken at its highest mark, could impact the development of interoperable software by developers who do not hold a market monopoly.[78]

For example, the GNU/Linux free and open source operating system (which does not hold a market monopoly in the personal computer operating system market) contains the Portable Operating System Interface (POSIX) family of reimplemented APIs, which collectively allow software designed for proprietary versions of UNIX to run on Linux.[79] Although the Institute of Electrical and Electronic Engineers (IEEE) Computer Society supports POSIX as a standard for interoperable software development, the current licence holders of proprietary UNIX, Micro Focus, could potentially bring suit against the POSIX developers for copyright infringement.[80] On the opposite end of the spectrum, in 2016 Microsoft introduced a Linux subsystem into their Windows operating system, engaging in a 'clean room' implementation of Linux APIs to avoid copying any source code into their reimplementation of Linux commands.[81] Whilst the current version of the GNU General Public Licence (version 3) contains an explicit recognition of fair use rights or any other rights held by the licensee,[82] under the precedent from *Oracle v Google* an aggrieved rights holder in the GNU/Linux operating system might pursue Microsoft for copyright infringement. This result is particularly problematic in light of the GPL's derivative licensing requirements, where all derivatives of the original GPL licensed code are subsequently relicensed under the GPL upon their publication.[83]

Asay further argues that the ruling from *Oracle* could even have an impact on statutorily legitimate purposes such as research and teaching purposes. The fair use section of the US Copyright Code, although not exhaustively, suggests that uses for 'teaching', 'scholarship' and 'research' are indicative of fair use purposes.[84] However, as mentioned previously, Sag has noted that US courts have still treated the presence or absence of a transformative element as strongly indicative of fair

77 Nathan Newman, 'Search, Antitrust, and the Economics of the Control of User Data' (2014) 31(2) *Yale Journal on Regulation* 401, 409–10.

78 Joseph Gratz and Mark A Lemley, 'Platforms and Interoperability in Oracle V. Google' (2018) 31(SI) *Harvard Journal of Law & Technology* 603, 604.

79 DR Kuhn, 'IEEE's Posix: Making Progress' (1991) 28(12) *IEEE Spectrum* 36.

80 Klint Finley, *The Oracle-Google Case Will Decide the Future of Software* (23 May 2016) WIRED <https://www.wired.com/2016/05/oracle-google-case-will-decide-future-software/>.

81 Jack Hammons, *Windows Subsystem for Linux Overview – Windows Subsystem for Linux* (22 April 2016) Microsoft | Developer <https://blogs.msdn microsoft.com/wsl/2016/04/22/windows-subsystem-for-linux-overview/>.

82 *The GNU General Public License v3.0 - GNU Project - Free Software Foundation* clause 2 <http://www.gnu.org/copyleft/gpl.html>.

83 Clark D Asay, 'The General Public License Version 3.0: Making or Breaking the Foss Movement' (2007) 14(2) *Michigan Telecommunications and Technology Law Review* 265, 268.

84 17 U.S. Code § 107 – *Limitations on Exclusive Rights: Fair Use.*

use.[85] Accordingly, if rewriting source code to mimic functionality is now no longer considered transformative, a significant number of scientific software packages may now be infringing on third party copyright.[86] The impact of this narrowing of the boundaries of transformative fair use may also adversely affect scientists and researchers who use software to analyse 'big data' drawn from multiple sources. Sag argues that the easiest way to deal with software programs that can be used to conduct such 'mass infringement' would be to include a broad exception for copyright infringement using such technology.[87] However, Reichman, Uhlir and Dedeurwaerdere argue that this approach would directly conflict with the goals of the scientific publishing industry, particularly in biomedical and agricultural research.[88] Whilst Reichman, Uhlir and Dedeurwaerdere suggest relying on a norm based system to encourage scientific publishers to engage with the patent system, research into norm based computational science research suggests that there is still tension over encouraging open access and ensuring the sustainability of scientific research.[89] This tension leads Reichman, Uhlir and Dedeurwaerdere to suggest that transformative use factor in fair use, now threatened by the precedent from *Oracle v Google*, as the last barrier protecting researchers working in data intensive fields from copyright infringement.[90]

Accordingly, although an optimal fair use standard in a copyright system will provide 'spillover' social benefits to artistic and literary creations, the decision in *Oracle v Google* significantly imperils the efficacy of the fair use system.[91] In particular, the lack of a functioning fair use system is what Wendy Gordon describes as a 'market failure', where the use of copyrighted materials under that fair use system is not efficient, leading to a net social welfare loss. In other words, the presence of a narrow fair use defence forces other creators to not use those copyrighted materials when there is no possibility of cross licensing of the copyrighted material due to transactional breakdowns, thereby leading to a decrease in follow on creation.[92] The next section of this chapter will address how various stakeholders in the Australian copyright system should respond to the issues surrounding a reinvigorated fair use system in Australia.

85 Sag, above n 75, 55; Matthew Sag, 'Copyright and Copy-Reliant Technology' (2009) 103(4) *Northwestern University Law Review* 1607, 1675.
86 Asay, 'Transformative Use in Software', above n 76, 15.
87 Sag, above n 85, 1675–82.
88 Jerome H Reichman, Tom Dedeurwaerdere and Paul F Uhlir, *Governing Digitally Integrated Genetic Resources, Data, and Literature: Global Intellectual Property Strategies for a Redesigned Microbial Research Commons* (Cambridge University Press, 2016) 333.
89 Matthew S Mayernik et al, 'Assessing and Tracing the Outcomes and Impact of Research Infrastructures' (2017) 68(6) *Journal of the Association for Information Science and Technology* 1341, 1346.
90 Reichman, Dedeurwaerdere and Uhlir, above n 88, 334.
91 William M Landes and Richard A Posner, 'An Economic Analysis of Copyright Law' (1989) 18(2) *Journal of Legal Studies* 325, 359; Brett M Frischmann and Mark A Lemley, 'Spillovers' (2007) 100(2) *Columbia Law Review* 257, 287–288; Wendy J Gordon, 'The Fair Use Doctrine: Markets, Market Failure and Rights of Use' in Richard Watt (ed), *Handbook on the Economics of Copyright: A Guide for Students and Teachers* (Edward Elgar Publishing, 2014) 84–6.
92 Wendy J Gordon, 'Fair Use as Market Failure: A Structural and Economic Analysis of the Betamax Case and Its Predecessors' (1982) 82(8) *Columbia Law Review* 1600, 1614–5.

5.2 The implications of *Oracle v Google* for an amended fair use defence in the *Copyright Act 1968* (Cth): recommendations for legislators and judges

How should law makers approach the question of fair use reform in Australia? A challenge to any legal transplant such as fair use will be attempting to meld the considerable existing case law under *Copyright Act 1968* (Cth) with US case law. The situation is somewhat complicated by the requirement under international copyright law that signatories to the various treaties comply with the three part test under Article 13 of *The Agreement on Trade Related Aspects of Intellectual Property (TRIPS Agreement)* for determining copyright flexibilities. This three part test stipulates that, to be compliance with international law, national copyright exceptions:

1. must be restricted to 'special cases';

2. must not prevent the normal exploitation of a copyrighted work; and,

3. must not unreasonably prejudice the interests of the rights holder.[93]

A further complicating factor are suggestions by Okediji and others that the US fair use doctrine is incompatible with the third step of this test because a fair use doctrine prevents the lawful exploitation of a copyrighted work.[94] This dilemma is not aided by the US Trade Representative's office rebuffing requests to introduce fair use abroad,[95] as well as a decision of a WTO Dispute Settlement Panel that held that section 110(5) of the US *Copyright Act*, which permitted royalty free performance of copyrighted works in bar, infringed all three steps.[96]

However, the question of the fair use doctrine's compatibility with international law remains an open ended question. In considering the drafting history of the three step test, Geiger, Senftleben and Gervais note that the original three step test included in the Stockholm Revision Conference was defined as a proportionality test that allows for the consideration of policy factors and does not treat the rights of a copyright holder as limitless.[97] Accordingly, it is possible that the WTO decision may not be treated as necessarily invalidating the fair use test (as opposed to the section 110(5) of

93 Berne Convention for the Protection of Literary and Artistic Works, opened for signature 9th September 1886, 1161 UNTS 30 (entered into force 4th December 1887) article 5(2), article 7(1), article 9(2); Marrakesh Agreement Establishing the World Trade Organization, annex IC, The Agreement on Trade Related Aspects of Intellectual Property Rights ('TRIPS Agreement'), opened for signature 15th April 1994, 1867 UNTS 3 (entered into force 1st January 1995) article 12, article 13.

94 Ruth L Okediji, 'Toward an International Fair Use Doctrine' (2000) 39(1) *Columbia Journal of Transnational Law* 75, 126–34.

95 Jonathan Band and Masanobu Katoh, *Interfaces on Trial 2.0* (MIT Press, 2011) 182.

96 Dispute Resolution Panel, *United States – Section 110(5) of the US Copyright Act*, WT/DS160/24/Add.158 (15 January 2001)

97 Christophe Geiger, Daniel Gervais and Martin Senftleben, 'The Three-Step Test Revisited: How to Use the Test's Flexibility in National Copyright Law' (2013) 29 *American University International Law Review* 581, 625–6.

the US *Copyright Act*),[98] raising the question of whether this reform can be transplanted into the existing Australian copyright system.[99]

The limited number of jurisdictions that have implemented a fair use defence into copyright law (such as Israel, Malaysia, Singapore and South Korea) have relied on US precedent.[100] However, as discussed previously, this approach risks importing deleterious precedent such as the *Oracle v Google* decision. The potential 'chilling' effects of the *Google v Oracle* decision can be reflected in Cotter's 'rights accretion' hypothesis, where a legitimate user of copyrighted material could be discouraged from using that material due to concerns about legal sanctions. This situation would create what Cotter calls a 'Pareto inferior' situation, where, in the absence of a comprehensive fair use test and where transaction costs are greater than zero, neither party benefits due to the absence of cross licensing or fair use as a defence outside the narrow boundaries of such defences such as research and study.[101]

To this end, Dnes discusses an alternative approach to the implementation of a copyright fair use defence into UK law (another common law jurisdiction that had discussed the implementation of a fair use defence) by importing common law and equitable doctrines into the interpretation of the UK's fair dealing defence. The *Copyright, Patents and Designs Act 1988* (UK) is similar to the current *Copyright Act 1968* (Cth) in that it contains a set of fair dealing exemptions in sections 29 and 30 of the act, together with a longer list of specific exemptions in the remainder of Chapter III of the UK Act, as opposed to a fair use test. Specifically, sections 29 and 30 indicate three very specific fair dealing purposes (provided that each are accompanied by appropriate acknowledgement):

1. Private research and non-commercial use;

2. Criticism or review of a work; and,

3. Reporting of current affairs and news.

However, courts in the UK have interpreted the fair dealing exemptions (specifically sections 29 and 30) have been interpreted flexibly, with Lord Denning in the case of *Hubbard v Vosper* comparing the application of the fair dealing test to the equitable nature of fair comment under defamation law.[102] Unfortunately (at least from the perspective of a fair use defence), the subsequent

98 Martin Senftleben, 'The International Three-Step Test. A Model Provision for EC Fair Use Legislation' (2010) 1(2) *Journal of Intellectual Property, Information Technology and E-Commerce Law* 76, 79 <http://www.jipitec.eu/issues/jipitec-1-2-2010/2605>.

99 Peter K Yu, 'Customizing Fair Use Transplants' (2018) 7(1) *Laws* 9.

100 Gabriel J Michael, *'To Promote the Progress'? Explaining the Global Diffusion of Intellectual Property Law* (PhD Thesis, The George Washington University, 2014) 229.

101 Thomas F Cotter, 'Fair Use and Copyright Overenforcement' (2007) 93(4) *Iowa Law Review* 1271, 1277–8.

102 *Hubbard v Vosper* [1972] 2 QB 84, 94.

decision of Justice Proudman in *Newspaper Licensing Agency v Meltwater BV & Ors* adopted a narrower interpretation of the meaning of what was fair for the purposes of fair dealing requiring that there be some element of public advantage to justify infringement.[103]

Nevertheless, Dnes argues that it may be possible to combine the fair dealing exceptions with other specific exceptions to create a 'fair use' jurisprudence.[104] In particular, section 31 of the UK Act codifies a *de minimis* exception for copyright infringement and allows for the taking of non insubstantial pieces of copyrighted material. Accordingly, Dnes argues that in concert with the broad interpretation of fair dealing supported by Lord Denning in *Hubbard*, the *de minimis* exception could be used to build the foundations of a fair use jurisprudence in the UK.[105] In addition, the advantage of this style of fair use is that it allows a flexible approach when dealing with new technologies ~~suc~~. These emerging technologies include~~h~~ ~~as~~ data mining and text retrieval (which Handke, Guibault and Vallbé suggest are currently ~~are currently~~more likely to be compromised by EU copyright ~~standards~~laws that require the express permissions of contracting parties).[106]

What relevance does this debate over UK reform have to the copyright reform in Australia? As Pappalardo and Fitzgerald note, UK copyright law has significantly influenced the development of Australian copyright law (such as the references to copyright protection under section 51(xviii) of the Australian Constitution).[107] Indeed, in determining the scope of this provision, the High Court has repeatedly characterised these powers as embodying a social contract model of copyright, where the interests of rights holders must be balanced against the interests of the public.[108] Accordingly, it is the contention of this submission that this flexibility would provide sufficient scope to introduce an operational fair use defence into the *Copyright Act 1968* (Cth).

In addition, adopting this flexible policy driven approach to fair use will have the benefit of avoiding the deleterious consequences that the *Oracle v Google* precedent would have on software development in Australia. In 2013, exports account for 19.9 percent of Australia's GDP, but only 1.2

103 *Newspaper Licensing Agency v Meltwater BV & Ors* (2010) EWHC 3099 (Ch), paragraph 115. In particular, Justice Proudman was heavily influenced by the Information Society Directive (OJ Law 167, 22/06/2001 P. 0010–0019), Articles 5.3(c) and (d) of which mandate the protection of newspaper headlines and quotes unless attributions can be made to the original author.

104 Antony W Dnes, 'Should the UK Move to a Fair-Use Copyright Exception?' (2013) 44(4) *IIC - International Review of Intellectual Property and Competition Law* 418, 426, 430.

105 Ibid 444.

106 Christian Handke, Lucie Guibault and Joan-Josep Vallbé, 'Is Europe Falling behind in Data Mining? Copyright's Impact on Data Mining in Academic Research' in *New Avenues for Electronic Publishing in the Age of Infinite Collections and Citizen Science: Scale, Openness and Trust: proceedings of the 19th International Conference on Electronic Publishing* (2015) 120, 128–9.

107 Kylie Pappalardo and Brian Fitzgerald, 'Copyright, Fair Use and the Australian Constitution' in Brian Fitzgerald and John Gilchrist (eds), *Copyright Perspectives: Past, Present and Prospect* (Springer, 2015) 125, 129–30.

108 *Grain Pool of Western Australia v Commonwealth of Australia* (2000) 202 CLR 479, 496; *Ice TV v Nine Network Australia Pty Ltd* (2009) 239 CLR 458, 472 (French CJ, Kiefel and Crennan JJ; 'the information/expression dichotomy, in copyright, is rooted in considerations of social utility').

percent of Australia's total exports are ICT related exports.[109] However in 2013, Australian businesses spent 41 percent of their total research and development expenditure on ICT research (7.7 billion Australian dollars). In the same year, Australian governments and non-profits spent 40 percent of their total research and development expenditure on ICT research (407 million Australian dollars).[110] In addition, a 2015 report by the Chief Scientist revealed that advanced physics and mathematics research accounts for 22.5 percent of Australian economic activity, or approximately 292 billion Australian dollars.[111]

Accordingly, there is a strong incentive to ensure that this research and development investment in the nascent ICT field in Australia is appropriately supported. In other jurisdictions, a significant amount of software development and computationally driven research is supported by reducing the transactional costs associated with either the replication of functionality from copyrighted software or the use of data mining for copyrighted data collections.[112] However, a shift towards the restrictive interpretation of the fair use test for software seen in *Oracle v Google* would have a significant detrimental impact on this research. To this end, in implementing a modified fair use test into the *Copyright Act 1968* (Cth), the government and courts must avoid adopting a narrow conception of fair use as that relied upon by the Federal Circuit in *Oracle v Google*.

6. Conclusion

This submission supports the adoption of a fair use test for copyright infringement in the *Copyright Act 1968* (Cth). However, this submission also encourages legislators and courts to remain mindful of the purpose of fair use in lowering the transaction costs associated with legitimate, non-infringing uses of copyrighted software. These legitimate uses can include the replication and reuse of software functionality, which is a crucial part of modern object oriented software development. This process is supported (at least in the US) by both a restriction of copyright protection to only the literary aspects of software and a broad fair use exception that lowers the transaction costs associated with legitimate use of copyrighted material. However, the Federal Circuit Court of Appeals decisions in the *Oracle v Google* cases to both extend the scope of copyright protection to

109 Michael, above n 100, 263.Michael, above n 99, 263.

110 *Business Expenditure on R&D (BERD) - 2013 to 2014* (4 September 2015) Australian Bureau of Statistics <http://www.abs.gov.au/AUSSTATS/abs@ nsf/Previousproducts/8104.0Main%20Features22013-14?opendocument&tabname=Summary&prodno=8104.0&issue=2013-14&num=&view=>; *Government Resources Devoted To Research and Experimental Development (R&D)* (6 July 2016) Australian Bureau of Statistics <http://www.abs.gov.au/ausstats/abs@.nsf/mf/8109.0>.

111 Centre for International Economics and Office of the Chief Scientist, 'The importance of advanced physical and mathematical sciences to the Australian economy' (Economic Report, *Australian Academy of Science*, March 2015), 50 <https://www.science.org.au/supporting-science/science-sector-analysis/reports-and-publications/importance-advanced-physical-and>.

112 TA Alspaugh, HU Asuncion and W Scacchi, 'Intellectual Property Rights Requirements for Heterogeneously-Licensed Systems' in *2009 17th IEEE International Requirements Engineering Conference* (2009) 24; Handke, Guibault and Vallbé, above n 106, 129.

Application Programming Interfaces (APIs) and to limit the scope of fair use for the reverse engineering of software functionality has the potential to compromise object oriented development. Accordingly, the amended fair use test that will be introduced into the *Copyright Act 1968* (Cth) must be drafted so as to give effect to the fundamental social policy principles that have pervaded Australian intellectual property law, as well as the technical realities of software development and computationally driven science.