Reverse Engineering: a European IPR perspective

Paolo Ciancarini
University of Bologna & CINI
via Mura Anteo Zamboni, 7
Bologna, Italy
paolo.ciancarini@unibo.it

Daniel Russo[*]
University of Bologna & CINI
via Mura Anteo Zamboni, 7
Bologna, Italy
daniel.russo@unibo.it

Alberto Sillitti
Center for Applied Software
Engineering & CINI
Italy
alberto@case-research.it

Giancarlo Succi
Innopolis University
52 Peterburgskaya st.
Kazan, Russia
giancarlo.succi@innopolis.ru

ABSTRACT

Even if reverse engineering is a well known and deeply investigated activity in software engineering, little research has been performed from an Intellectual Propriety Rights (IPRs) perspective. In this paper, we analyze some cases of reverse engineering in a IPR perspective; in particular, we survey the behavior of the major European court about reverse engineering. Apparently, legal *fora* have little attention for copyright infringements regarding reverse engineering. The major contribution of this work is an analysis case by case of the European Court of Justice (ECJ). The ECJ looks very permissive regarding reverse engineering, since "principles," or "ideas," are not copyrightable by themselves. In particular we discuss the impact of a recent and relevant ECJ ruling.

CCS Concepts

•**Social and professional topics** → **Software reverse engineering;** *Copyrights;*

Keywords

Software Engineering; Reverse Engineering; Copyright.

1. INTRODUCTION

Reverse engineering is a common practice in software engineering in order to understand how a piece of software works and let it to interoperate with other systems. The size and complexity of computer programs are one of the greatest concern of the software industry because of the huge maintenance costs. Not surprisingly, the estimated cost of software maintenance accounts for 50% to 75% of the overall cost of a software system [10, 4, 3, 6]. Since many software companies do not provide detailed information about how their systems work (e.g., to prevent undesired alteration of the behavior of the system, to keep a business advantage, etc.) experts perform reverse engineering activities to disclose such important knowledge and use it for different purposes (e.g., integration with other systems, improvements, risk analysis, etc.). Two are the most relevant activities for reverse engineering [2]:

- *Redocumentation*, is the analysis of a system to reproduce documentation in various forms e.g.users manuals. The resulting forms are artifacts which are understandable by humans i.e., data flows. Apparently it is the oldest form of reverse engineering and it is considered to be particularly unintrusive [2]. There are specific tools for reducumentation to provide an easier way to visualize the relationships among program components;
- *Design recovery*, is when external information is added to the observations of the system to identify higher levels of abstractions beyond those obtained directly by examining the system itself. It recreates the design abstractions from the combination of the available knowledge about the problem and the application domain. The aim of design recovery is to collect all available information required to fully understand all program's functions. Therefore, it deals with much more information than just representations of the code.

Furthermore, [2] proposes also other activities, strictly related to reverse engineering, namely:

- *Restructuring*, or the transformation from one representation form to another at the same abstraction level

[*] Corresponding author.

SAC 2016,April 04-08, 2016, Pisa, Italy
©2016 ACM. ISBN 978-1-4503-3739-7/16/04. . . $15.00
DOI: http://dx.doi.org/10.1145/2851613.2851790

Table 1: Reverse engineering scenarios

Type	Main characteristics
Achieving Interoperability with other Software	Development of applications that interoperate with operating systems libraries or applications
Legacy Software Maintenance	Design legacy recovery when the source code of modules is not available for maintenance
Software Quality Evaluation	Preventive software quality assurance
Implementation Testing	Verification of the forward development produced code matches the design

while maintaining unchanged the functionality and the semantics of the program;

- *Reengineering*, is both the examination and the modification of a system, to transform it in another form, as its implementation.

There are, at least, four scenarios where reverse engineering applies [7]; Table 1 shows them clearly.

A primary goal of reverse engineering is to produce software views by abstracting facts stored in a knowledge base to cope with new and emerging software development scenarios and new system architecture [1]; however, little attention was devoted by the research community in studying its legal implications.

Software is an artifact protected in the EU by copyright laws[1]. When dealing with any legally protected artifact, any software engineer should be aware of the legal consequences of reverse engineering. This paper explores the position of the European Court of Justice (the highest court in Europe) regarding this issue. Here, regardless of the personal opinion of scholars, we analyze the position of the ECJ, in order to understand and possibly predict future leanings of lower courts in Europe.

The structure of this paper is as follows. In Section 2 a short legal overview is given, to understand the main reference points of courts. In Section 3 we explain our research methodology regarding the case law analysis. A manual analysis of the case law was carried out. In Section 4 we discuss a recent ruling of the European Court of Justice. In Section 5 we discuss the main implication of this paper. Finally, in Section 6 we summarize our main findings and discuss some future work.

2. IPR IN THE EU

The ECJ has to apply the EU law. A particularity of the ECJ is that it acts as a Supreme Court of several States with (slightly) different laws. Each ECJ judgment is binding for any court of each member state. The EU law system is a mixture of the civil law and the common law systems. Each court has to interpret the national law but the ECJ rulings are binding[2]. The most common EU laws regarding copyright are represented in Table 2.

Table 2: EU Directives regarding Copyright

Directive	Subject
Council Directive 93/83/EEC	Copyright and related rights: satellite broadcasting and cable retransmission
Directive 98/84/EC	Protecting electronic pay services against piracy
Directive 96/9/EC	Legal protection: databases
Directive 2001/29/EC	Copyright and related rights in the information society
Directive 2001/84/EC	Resale right for the benefit of the author of an original work of art
Directive 2006/116/EC	Copyright and related rights: term of protection
Directive 2006/115/EC	Rental, lending and certain other rights related to copyright in the field of intellectual property
Directive 2009/24/EC	Legal protection: computer programs
Directive 2012/28/EU	Wider access to copyright material - Orphan works

Furthermore, there are also EU Commission's recommendations, shown in Table 3. Even if these recommendations are not formally binding, they are part of the EU legislation, since they elaborate a formalized position of the EU legislative bodies.

Table 3: Commissions's Reccomendations

Reccomendation	Subject
COM (2004) 261	Management of copyright and related rights in the internal market
Commission Recommendation 2005/737/EC	Management of online rights in musical works
COM(2009) 532 final	Copyright in the Knowledge Economy

Reverse engineering in software, as such, is considered by the EU Law an issue that underlies to copyright. Since software is protected by copyright, all actions related to software are part of the legal domanin of the copyright law.

3. CASE LAW PROTOCOL

A systematic review of law cases has some strong limitations since there is no official database that collects all law cases. Rulings are written by judges, after that employees of commercial databases e.g., LexisNexis put them on line. So, it is possible that some rulings are not into any database, this happens usually with lower courts judgments considered by the database editor of low importance or impact.

Moreover, a literature review in such context is very different compared to a Systematic Literature Review (SLR) in the Computer Science domain. Law databases contain technical keywords to help in the identification of the cases but such keywords belong to the law domain and not to the Computer Science domain. Therefore, the identification of the relevant law cases is quite difficult and includes a high number of manual inspections also related to the fact that

[1] Protection of computer programs: Directive 2009/24/EC

[2] Please consider that this is a very rough approximation, useful for the purposes of this article.

interpretation is a key aspect of law.

The arbitrary dimension in the analysis of case law is ineluctable. Nonetheless, our main reference for the design of our review protocol was [9], to stick as much as possible to the empirical software analysis tradition. To systematize as much as possible this study, we set up the following review protocol:

1. identification of the country/legal system where to carry out the analysis;
2. definition of an appropriate database for the case law;
3. definition of an appropriate query, according to the legal system;
4. manual identification of the relevant cases:
   (a) exclusion criteria: not relevance to the topic;
   (b) inclusion criteria: relevance to the topic;
5. manual analysis of the relevant cases.

According to our protocol, we choose to analyze the EU legal system's on reverse engineering.

We used Eur-Lex[3]. Eur-Lex has the advantage of being the official Law database of the European Court of Justice. So, any judgment of the ECJ is available in the Eur-Lex database.

The identified query was "software & copyright" in the textual search. We consciously have chosen a broad query, to match also possible outliers. We used a query able to catch the highest number of potential law cases regarding reverse engineering, since it is considered by EU Law a sub domain of copyright. The total output were 27 cases. The query was carried out on June 3rd, 2015.

The outcome of the 27 cases for the EU was reproduced in Appendix A. After that, they were manually analyzed case by case. Cases discussing other topics were not taken into consideration in the table. In total, the excluded cases were 26. The remaining relevant case was then analyzed and deeply discussed in Section 4.

3.1 Preliminary remarks on our dataset

The included cases are a large minority of the dataset. This because the query was kept as general as possible to catch most of the relevant cases.

We took just the EU cases, at its highest level of the ECJ. The study of the US and other countries case law is of great interest and we are committed to pursue this study further. We remark that we only considered case law at EU level, since each Member State has its own case law in its own language. However, each courts of the member states has to comply and respect the ECJ rulings. So this research is significant for all Member States.

Interestingly, the ECJ set an important milestone with the SAS Institute Inc. v World Programming Ltd. case, which has several disruptive elements regarding copyright law, which are analyzed in the next section.

4. A RECENT RULING BY THE ECJ

In the EU, the paradigm about copyright of software changed after the SAS Institute v. World Programming Ltd[4] ruling by the ECJ. This ruling has clear and strong implications for software programs and related lawsuits. More in detail, the ECJ stated three important principles about the interpretation of Directives 91/250 and 2001/29:

1. The first part of this ruling states that *the legal protection of computer programs is to be interpreted as meaning that the functionalities of a computer program and of a the programming language are not eligible,* ***as such****, for copyright protection. It will be for the national court to examine whether, in reproducing these functionalities in its computer program, the author of the program has reproduced a substantial part of the elements of the first program which are the expression of the author's own intellectual creation.* This means that a program's features are "principles" or "ideas," so not copyrightable expressions by themselves. Software program's copyright is no more an absolute assumption but a relative one. For sure programs are copyrightable but in the sense that they are a form of expression of the intellectual creation of the programmer.

2. Furthermore, in the most relevant part for our purposes, regarding reverse engineering, the ECJ states that *it is not regarded as an act subject to authorization for a licensee to reproduce a code or to translate the form of the code of a data file format so as to be able to write, in his own computer program, a source code which reads and writes that file format, provided that that act is absolutely indispensable for the purposes of obtaining the information necessary to achieve interoperability between the elements of different programs. That act must not have the effect of enabling the licensee to recopy the code of the computer program in his own program, a question which will be for the national court to determine. [...] Acts of observing, studying or testing the functioning of a computer program which are performed in accordance with that provision must not have the effect of enabling the person having a right to use a copy of the program to access information which is protected by copyright, such as the source code or the object code.* This means that anyone, who acquired a license of a program can freely observe, study or test it to fix interoperability or for education purposes. Any program, which has been acquired legally, can be studied and the copyright holder is not able to prevent it. Interestingly, even though someone would study the program, to copy it, this could not be considered a copyright infringement.

3. Finally, the last point, which is possibly less relevant from our perspective, is the copyright of the user manual. In fact, *the reproduction, in a computer program or a user manual, of certain elements described in the manual for another computer program may constitute an infringement of the copyright in the latter manual if - a question which will be for the national court to*

[3] www.eur-lex.europa.eu

[4] C-406/10, 02.05.2012

> *determine - the elements reproduced in this way are the expression of their author's own intellectual creation.* Also for this case, the expression, original creation of the author, is protected by copyright law. Not protected are *keywords, syntax, commands and combinations of commands, options, defaults and iterations* singularly, but *the choice, sequence and combination of such elements that the author may express his creativity in an original manner and achieve a result which is an intellectual creation.*

This judgment is clearly disruptive from the case law point of view. Even though before this one there were no real case law regarding reverse engineering issues, this sentence has a big impact in the software community because it states relevant issues that have a direct impact for programmers, at least in Europe. With the C-406/10 ruling we can figure out the following suggestions for programmers:

- it is possible to reproduce "principles" or "ideas" of other people's software programs;
- it is possible to get revenues from others' "principles" or "ideas," because they are not copyrightable;
- "principles" or "ideas" are not copyrightable, so everyone can get full inspiration from them;
- legal fora are not the right place where to defend "principles" or "ideas" because no legal paradigm protects them;
- European courts may intervene only if the program code itself is copied;
- the source code of any program can be studied, without any permission of the licensor for "study" purposes. Therefore, if someone studies the source code of a program to get its "principles" or "ideas" to exploit them, no one can, *de facto* prevent it. Even though the court deals with a case which regards reverse engineering specifically for interoperability issues, it is quite easy to go beyond this limit. In principle, any programmer could claim to have "studied" it for interoperability. So, even if nothing would come out (in terms of interoperability) still the programmer could have been studied the program, without any restriction. Finally, no one could claim copyright issues if he gets "principles" or "ideas" of that program, for his own program.
- it is legal to copy "principles" or "ideas" of any program, even for profit.

5. IMPACT ON THE COMMUNITY

This paper describes the 'way of thinking' of EU courts when it comes to reverse engineering in software and, more generally, to IPR issues.

For all EU Member States, the SAS Institute v. World Programming Ltd ruling will have a disruptive impact regarding reverse engineering. So, from academic, non profit or FOSS application, up to commercial, closed source and business applications, reengineering has to be considered, at least permitted. Anyone can use "principles" or "ideas" of another software artifact for the proper use. And no one can legally claim an IPR protection if someone else exploit the own original "principles" or "ideas" used in someone else artifact. If the software community in Europe wants to have a more stringent protection about the software, European courts (of all Member States) are not the suited place to get this protection.

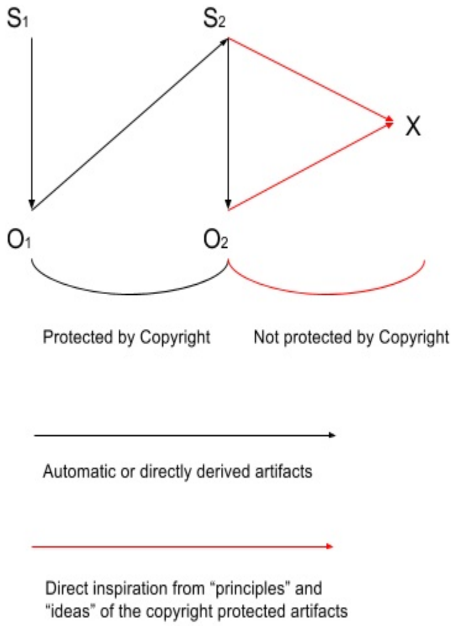
S1
S2
X
O1
O2
Protected by Copyright
Not protected by Copyright
Automatic or directly derived artifacts
Direct inspiration from "principles" and "ideas" of the copyright protected artifacts

Figure 1: Copyright protected and not protected reengineering according to the ECJ.

One big issue is the architecture visualization of a system [5]. In literature, there are different visualization tools proposal, to cope with the complexity of computer programs [8]. But to what extend, from a legal point of view, a software engineer can apply these visualization tools to study and understand better the architecture of a program? Which kind of authorization has he to ask for? According to the ECJ the answer is simple: there is no limitation. Anyone can *de facto* study the architecture for any purposes. First of all for interoperability issues but also for education ones. This means, basically everything. Furthermore there is no need for any kind of permission to do that.

Furthermore, it is of highest interest interpreting in an operative way the SAS ruling. The ECJ refers explicitly to "principles" or "ideas". So, direct derived work is still to consider protected by copyright, since it is much more then not just "principles" or "ideas" . The typical example is the relationship of the source code and the object code, explained in Figure 1. The object code O_1 is automatically derived by the source code S_1, elaborated by a compiler. Therefore, also the object code is protected by copyright, as it is an automatic execution of a human creative activity, namely the source code.

Going ahead, what happens to an object code O_2 which is decompiled in a source code S_2, which is slightly different with regard to e.g., the identifiers? Is it still a copyright protected artifact? The answer is yes. As the automatic compilation, also the decompilation is an automatic and directed process to get to a derivative product S_2 from an original S_1 protected one.

But what happens is if the relationship is not automatic or direct? Well, the answer is it depends. However, we can reasonably state that if the change to the software exposed to reverse engineering is not trivial and the new artifact X follows just "principles" or "ideas" of S_1, it is for the ECJ a new artifact with no legal relation to S_1 or S_2.

Ultimately, the fine line between derived artifacts protected by copyright (S_2) or new one (X) inspired by the original one, depends on the "degree of derivation". If the derivation is verry loose, and the X artifact recalls just "principles" or "ideas" of S_1, the ECJ would not claim any copyright infringment. On the other hand, if this recall is more than just a simply "inspiration", well the ECJ would seen this as infringment.

Unfortunately, there is not a easy answer to this issue. Since we are dealing with courts, there will always be a degree of arbitrarity. Nonetheless, we found out that copyright protection of software within the EU is relatively loose.

To sum up, the software engineer that has to reverse engineer a computer program for education and interoperability purposes, will always be protected by the ECJ and has not to worry any legal action by the software owner. This, without asking for any kind of permission. Moreover, there is no worries of infringements if he wants to reengineer a software, taking into account only "principles" or "ideas" of the original one.

6. CONCLUSIONS

In this paper we analyzed the behavior of the ECJ regarding software reverse engineering issues.

The European Court of Justice has a loose approach to copyright protection, regarding software. With the SAS Institute v. World Programming Ltd ruling, the ECJ has been quite disruptive for all Member States courts within the European Union, since it is a Supreme Court pronouncement. The Court stated that reengineering of "principles" or "ideas" can not be an infringement of copyright, since "principles" or "ideas" are not copyrightable. Likewise, reverse engineering is legally permitted, without asking for permission to the software owner.

Since the SAS ruling by the ECJ is quite recent, there will be in the next future new judgments by lower courts about these issues. It will be interesting to see how future judgments will evolve, after this supreme court judgment.

Future research will focus on the US case law. We did not take into account, in our study, the comparison of the two legal systems. Moreover, a broader analysis on IPR issues would also be interesting in a comparative dimension. Beside reverse engineering, there is also the broader field of software cloning i.e., a form of software reuse. In any case such debate is crucial for software engineers, since it has a huge impact on the copyright and commercialization of software. Further works could figure how such rules are considered within the software engineer community, according to the past case law and the software engineering literature. Furthermore, a qualitative study on the most relevant European software houses can be pursued, to understand, how and if the SAS ruling changed their business model and IPR policies. It would be interesting to study the change of behavior (if there is a change) of software houses to protect their copyright. An unstructured interview with key target software houses could give good insights for first results and further steps.

7. ACKNOWLEDGMENTS

This paper has been partially supported by MIUR PRIN IDEAS and the Consorzio Interuniversitario Nazionale per l'Informatica (CINI) - the EU ARTEMIS EMC2 Project.

8. REFERENCES

[1] Canfora, G., Di Penta, M., Cerulo, L.: Achievements and Challenges in Software Reverse Engineering. *Comunications of the ACM.* 54 (4), 142–151 (2011)

[2] Chikofsky, E., J., Cross, J, H. II: Reverse Engineering and Design Recovery: A Taxonomy *IEEE Software.* 7 (1), 13–17 (1990)

[3] Ciancarini P., Poggi F., Rossi D., Sillitti A.: Improving bug predictions in multi-core cyber-physical systems *4th Int. Conf. on Software Engineering for Defense Applications (SEDA 2015).* Rome, Italy, 2015

[4] Davis, A., M.: Principles of Software Development. McGraw-Hill, (1995)

[5] Gansner, E., R., North, S., C.: An open graph visualization system and its applications to software engineering. *Software Practice and Experience.* 30 (11), 1203–1233 (2000)

[6] Di Bella E., Sillitti A., Succi G.: A multivariate classification of open source developers *Information Sciences.* 221, 72–83 (2013)

[7] Klimek, I., Keltika, M., František, J.: Reverse engineering as an education tool in computer science. 9th Int. Conf. on Emerging eLearning Technologies and Applications (ICETA). 123–126 (2011)

[8] Lanza, M., Ducasse, S.: Polymetric views-a lightweight visual approach to reverse engineering *IEEE Transactions on Software Engineering.* 29 (9), 782–795 (2003)

[9] Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering *Empirical Software Engineering.* 14 (2), 131–164 (2009)

[10] Sommerville, I.: Software Engineering. 10th ed., Pearson (2015)

APPENDIX

A. THE EU CASE LAW

Case	Citation	Court	Year	Relevant Output
British Broadcasting Corporation and BBC Enterprises Ltd v Commission of the European Communities.	61989TJ0070	Judgment of the Court of First Instance (Second Chamber)	1991	Not relevant - Abuse of dominant position.
Independent Television Publications Ltd v Commission of the European Communities.	61989TJ0076	Judgment of the Court of First Instance (Second Chamber)	1991	Not relevant - Abuse of a dominant position.
Radio Telefis Eireann v Commission of the European Communities.	61989TJ0069	Judgment of the Court of First Instance (Second Chamber)	1991	Not relevant - Abuse of a dominant position.
Société d'Informatique Service Réalisation Organisation v Ampersand Software BV.	61993CJ0432	Judgment of the Court (Sixth Chamber)	1995	Not relevant - Convention on Jurisdiction and the Enforcement of Judgments.
Micro Leader Business v Commission of the European Communities.	61998TJ0198	Judgment of the Court of First Instance (Third Chamber)	1999	Not relevant - Trade issues.
Taurus-Film GmbH Co. v Office for Harmonisation in the Internal Market (Trade Marks and Designs) (OHIM).	61999TJ0136	Judgment of the Court of First Instance (Second Chamber)	2001	Not relevant - Acquisition of a community trade mark.
Taurus-Film GmbH Co. v Office for Harmonisation in the Internal Market (Trade Marks and Designs) (OHIM).	61999TJ0135	Judgment of the Court of First Instance (Second Chamber)	2001	Not relevant - Acquisition of a community trade mark.
SAT.1 SatellitenFernsehen GmbH v Office for Harmonisation in the Internal Market (Trade Marks and Designs) (OHIM).	62000TJ0323	Judgment of the Court of First Instance (Second Chamber)	2002	Not relevant - Acquisition of a community trade mark.
Les Éditions Albert René v Office for Harmonisation in the Internal Market (Trade Marks and Designs) (OHIM).	62001TJ0311	Judgment of the Court of First Instance (Fourth Chamber)	2003	Not relevant - Opposition proceedings.
Les Éditions Albert René v Office for Harmonisation in the Internal Market (Trade Marks and Designs) (OHIM).	62003TJ0336	Judgment of the Court of First Instance (Third Chamber)	2005	Not relevant - Opposition proceeding.
Microsoft Corp. v Commission of the European Communities.	62004TJ0201	Judgment of the Court of First Instance (Grand Chamber)	2007	Not relevant - Abuse of a dominant position.
Les Éditions Albert René Sàrl v Office for Harmonisation in the Internal Market (Trade Marks and Designs) (OHIM) and Orange A/S.	62006CJ0016	Judgment of the Court (Sixth Chamber)	2008	Not relevant - Opposition proceeding.
Danjaq, LLC v Office for Harmonisation in the Internal Market (Trade Marks and Designs) (OHIM).	62005TJ0435	Judgment of the Court of First Instance (Second Chamber)	2009	Not relevant - Opposition by the proprietor of the non-registered marks.
Apis-Hristovich EOOD v Lakorda AD.	62007CJ0545	Judgment of the Court (Fourth Chamber)	2009	Not relevant - Legal protection of databases.
Systran SA and Systran Luxembourg SA v European Commission.	62007TJ0019	Judgment of the General Court (Third Chamber)	2010	Not relevant - Claim for damages.
Systran SA and Systran Luxembourg SA v European Commission.	62007TJ0019	Judgment of the General Court (Third Chamber)	2010	Not relevant - Claim for damages.
Bezpe'cnostn'i softwarov'a asociace - Svaz softwarov'e ochrany v Ministerstvo kultury.	62009CJ0393	Judgment of the Court (Third Chamber)	2010	Not relevant - software copyright issues, not related to reverse engineering
Eva-Maria Painer v Standard VerlagsGmbH and Others.	62010CC0145	Judgment of the Court (Grand Chamber)	2011	Not relevant - Copyright protection of photos.
Scarlet Extended SA v Société belge des auteurs, compositeurs et éditeurs SCRL (SABAM).	62010CC0070	Judgment of the Court (Grand Chamber)	2011	Not relevant - Processing of personal data and on the free movement of such data.
Football Association Premier League Ltd and Others v QC Leisure and Others.	62008CJ0403	Judgment of the Court (Grand Chamber)	2011	Not relevant - Illicit device which give access to the satellite broadcasting services of a broadcaster.
SAS Institute Inc. v World Programming Ltd.	**62010CJ0406**	**Judgment of the Court (Grand Chamber)**	**2012**	**Relevant - Copyright and related rights in the information society.**
UsedSoft GmbH v. Oracle International Corp.	62011CJ0128	Judgment of the Court (Grand Chamber)	2012	Not relevant - software copyright issues, not related to reverse engineering
Microsoft Corp. v European Commission.	62008TJ0167	Judgment of the General Court (Second Chamber)	2012	Not relevant - Abuse of a dominant position.
Football Dataco Ltd and Others v Yahoo! UK Ltd and Others.	62010CJ0604	Judgment of the Court (Third Chamber)	2012	Not relevant - Legal protection of databases.
European Commission v Systran SA and Systran Luxembourg SA.	62011CJ0103	Judgment of the Court (First Chamber)	2013	Not relevant - Action in non-contractual liability against the European Community.
ITV Broadcasting Ltd and Others v TV Catch Up Ltd.	62011CJ0607	Judgment of the Court (Fourth Chamber)	2013	Not relevant - Broadcasting by a third party over the internet of signals of commercial television broadcasters.
Nintendo Co. Ltd and Others v PC Box Srl and 9Net Srl.	62012CJ0355	Judgment of the Court (Fourth Chamber)	2014	Not relevant - software copyright issues, not related to reverse engineering
Ryanair Ltd v PR Aviation BV.	62014CJ0030	Judgment of the Court (Second Chamber)	2015	Not relevant - Databases are not protected by copyright or the sui generis right.