RIGA GRADUATE SCHOOL OF LAW

Software source code escrow agreement and legal obstacles of its execution

MASTER'S THESIS

AUTHOR: Madara Štorha
LL.M 2019/2020 year student
student number M019037

SUPERVISOR: Rihards Gulbis
Dr.iur.

DECLARATION OF HONOUR:

I declare that this thesis is my own work, and that all references to, or quotations from, the work of others are fully and correctly cited.

(Signed) ………………………………...

RIGA, 2020

Abstract

Software source code escrow agreement is a relatively new occurrence, emerged together with the rise of software industry in 1970’s in the U.S. Taking into account the fact that business relationships between the software licensor and the licensee are mostly permanent, i.e. significant part is not only the license agreement, but also maintenance and support service duties assumed by the licensor, it is important that business continuity of the licensee is ensured. Software source code escrow agreement aims to provide this assurance by promise, that if the licensor ceases to exist (becomes insolvent), then the source code – building instruction of the software, will be turned over to the licensee and provide licensee with the option to maintain the software himself.

Thesis is dedicated to enquiry what obstacles impede execution of such contract and the answer lies in the consideration why the software escrow agreement is concluded in the first place – bankruptcy. That is to say, mandatory bankruptcy laws preclude any actions following the commencement of bankruptcy proceedings if they concern property of bankruptcy estate.

Summary

Master thesis „Software source code escrow agreement and legal obstacles of its execution" synthesized the legal and practical challenges software licensee encounters when trying to obtain software source code from the escrow in order to maintain and update software that is important for its business activities.

Firstly, it recognizes that software as a product in itself is not monolithic, that is to say – it is intangible, but at the same time those algorithms and concepts that are behind every software, produce objectively measurable results. Secondly, thesis acknowledges that the need for software source code escrow arrangements arises directly out of legal protection regimes for software, i.e. –trade secret, patent or copyright (or combined). Absence of clear legal regime for software protection at the time when industry emerged, has led to upsurge in software source code escrows services of whom are popular, because – if source code is kept confidential under trade secret laws, and at the same time its compiled version, to human eye unintelligible version – object code, is distributed to software users via licensing, then, if the licensor becomes insolvent, the licensee may find himself in a situation where software critically needed for its business operations, is either no longer maintained, updated or just available. Software source code escrow agreements try's to arrange for such a scenario in advance, by stating, that materials (source code, all relevant build instructions, programming documentation etc.) deposited with the escrow agent, and intended for unassisted software maintenance by the licensee, will be delivered to him, if and when a release event – most commonly, software's licensor bankruptcy, occurs.

However, such arrangements, although promising for every prudent software licensee, rarely serves their purpose, since, in contradiction to classical escrow arrangement, in software source code escrow the depositor (the licensee) does not terminate its dominium over deposited materials. On the contrary, according to standard software source code escrow agreement terms, the licensor may or may not consent that deposited materials are turned over to the licensee, even when (and especially so), when the licensor himself becomes bankrupt.

The foregoing initiate's dispute between the licensor and the licensee, which, according to software source code escrow agreement, should be dealt with in arbitration proceedings. But, given the fact that often substantial duties of parties towards each other are indicated in the license agreement itself, or maintenance and service agreement (to whom all escrow is being considered supplementary), and forum provisions there may differ, this proves a weak

reassurance for the licensee, as the dispute on whether there has occurred agreed-upon release event, may be dealt in civil court proceedings.

Finally, if parties find themselves in court due to the fact that the licensor, who has become bankrupt, refuses to assign deposited materials to the licensee, the latter faces several obstacles encumbering the obtainment of the source code and relevant materials. First, commencement of bankruptcy proceedings acts as an automatic stay in all proceedings where debtor licensor is involved as litigant, second, mandatory bankruptcy laws declare *ipso facto* clauses void, and, third, bankruptcy trustee or debtor himself, if acting as a debtor in possession, may sell intellectual property assets in question disregarding licensee's contractual rights to use licensed software for a term provided in license agreement.

In the light of aforementioned, software source code escrow agreement does not seem to be viable option to secure software licensee's business continuity in case the licensor becomes bankrupt.

Table of Contents

Introduction

In modern economy significance of software as a means to effectively conduct different business activities cannot be overestimated. However, together with economic benefits, for software licensees this brings risks, such as software licensor, i.e. developer, becoming insolvent. The importance of this, often common occasion, stems from *sui generis* nature of software and its selling practices, that is to say, the object licensee possesses has little value in itself if the licensee is not provided with continuing maintenance and support services vitally needed for unhampered use of software.

This paper is aimed to examine risk aversion instrument for software licensee's business continuity in case software developer becomes insolvent – software source code escrow agreement. In particular, whether the process of obtaining the escrowed source code is feasible, what legal obstacles licensees encounter while trying to obtain software source code and what legal strategies/agreement constructions would be most effective. It must be noted that although software source code escrow agreement sometimes is perceived as a legal instrument to safeguard the software developer's ownership rights[1], this paper focuses on the possible benefits it could bring to the software licensee. In the light of research subject – legal obstacles of execution of source code escrow agreement, the focus will be on standard escrow agreement terms, provided by business offering escrow services in the U.S. and European Union (hereinafter EU). Conducting research in a comparative manner (between common law legal system represented by U.S. and civil law legal system represented by EU and in specific parts the Federal Republic of Germany) is substantiated by the fact software industry emerged in the U.S., so it could provide approbated solutions for software licensee contractual rights protection in licensors bankruptcy which then, perhaps, could be useful in civil law legal system as well.

In the first part of the paper technical characteristics of software and source code will be provided. In the next one a comparative analysis between common and civil law legal systems on the legal modes implemented for software protection will be conducted. It is necessary in order to gain thorough understanding, why the businesses are concluding source code escrow agreements for a software that has already been delivered to the software licensee.

[1] *In re Bluberi Gaming Techs., Inc*., (4 August 2016, unreported), available on: file:///C:/Users/madarastorha.AT/Downloads/In%20re%20Bluberi%20Gaming%20Techs.,%20Inc.pdf. Accessed June 6, 2020.

The third part of thesis is devoted to enquiry what are the effects of bankruptcy of the software developer (the licensor) on software licensee's contractual rights – can it continue to use licensed software or not. Answer to this question in relation to thesis subject is important given the fact that software source code escrow agreements are of ancillary nature, that is to say, they are intended to secure the fulfilment of the principal agreement. As in the previous part, the inquiry into question will be conducted in comparative manner, but unlike in the previous part, instead of overview of EU regulation, the law of the Federal Republic of the Germany will be examined. This is done due to the limited thesis amount and taking into consideration the fact that some of the conclusions regarding normative regulation implemented in the Federal Republic of Germany could be useful for the legislation of the Republic of Latvia.

The fourth part of thesis deals with nature of escrow and software escrow agreement in particular, and also with the practicalities of such contract. In particular it explores what materials should be escrowed, on what conditions these materials are turned over to the software licensee and how bankruptcy proceedings of software licensor affects the execution of software source code escrow agreement. Due to the fact that relevant law of the Federal Republic of Germany does not provide specific regulation for software license agreement in case software licensor becomes bankrupt, this part of thesis contains analysis of U.S. bankruptcy regulation in relevant part.

The last, fifth part of thesis contains authors conclusions on the research subject.

In this thesis comparative and historical research methods are used to determine what obstacles and why exist, when execution of software source code is carried out.

1. Technical characteristics of software and source code

Terms of software and source code are often used interchangeably since the latter is the version of software as it is originally written (i.e., typed into a computer) by a human in plain text (i.e., human readable alphanumeric characters).[2]

Before attending to (brief) overview of history of development of law in software protection area, technical characteristics for terms „software" and „source code" will be provided. This is needed in order to be able to assess what kind of obstacles parties of software source code escrow agreement face, when the conditions for source code release occur, and whether aforementioned arrangement serves its purpose – safeguarding business continuity for software licensee. Given the fact that all the terms relevant to thesis subject (software, source code, license agreement and software escrow agreement), except the last two, are related to computers, then in order to understand substantiation behind the chosen protection for the subject matter, it is advisable to gain insight on how computer works and how those processes can be regulated from the perspective of law. In relation to computer – „a programmable electronic device for storing and processing data and displaying the results of these operations"[3], software is important because without it the computer cannot accomplish „the input, processing, output, storage, and control activities."[4]

1.1. Software as a quintessential part of a computer

At the core of every computer is a vast collection of on/off switches, known as circuits. By using these circuits in combination with one another „a computer is able to perform millions of calculations per second and thus execute the operations that make it so useful."[5] But for a computer to be able to perform these different tasks, it needs some kind of instructions and these orders are contained in a computer program or otherwise known as software. Software's function is to communicate tasks to computer hardware – commonly used term to describe physical parts of computer that contains central processing unit, motherboard and so on. At its most basic level, software comprises a related series of on/off commands to the computer's circuits that tell the computer what function to perform at a given time.

[2] Available on: https://perma.cc/UAP6-PR22. Accessed March 13, 2020.

[3] Available on: https://www.collinsdictionary.com/dictionary/english/computer. Accessed June 8, 2020.

[4] Available on: http://www.umsl.edu/~joshik/msis480/chapt05.htm. Accessed June 8, 2020.

[5] J. Gibson, „Once and Future Copyright," *Notre Dame L. Rev.*, (Notre Dame Law Review) Vol.81, Issue 1 (November 2005), p.174.

Due to the fact that objective of every software is to „embody the code in electronic circuitry, where it functions like, and often replaces machine parts"[6], deduction can be made that „the most important property of program is their behavior."[7] In other words, the value of software lies in the outputs it delivers. From the practical point of view software seems to be solely functional and utilitarian and correctness of such conclusion is reinforced by two factors. First one is the idea of the so called „tight code" which encompasses in itself the concept of concise coding, i.e. „to perform a necessary function in the least amount of lines of code while utilizing standardized structuring and indentations."[8] Thus the best coding practice is the most efficient one. The second factor speaking in favor of utilitarian nature of software is that the industry itself encourages interoperability and standardization for computer products. This is driven by economic considerations since computers run not one, but multiple software's and in order for them to be able to „communicate with each other" - to bring about certain outputs, compatibility, not variety, is the value. As Advocate General Mr. Bot emphasized in his opinion in Case-393/09, this communication between different software's embedded into computers is achieved by „interconnection interfaces, which are internal to the software and permit dialogue with other elements of the computer system."

Notwithstanding the utilitarian nature of software, it is commonly accepted that in order to create/code it, a significant degree of creativity and ingenuity is needed. Steps necessary to create a software consists of: 1) identification of specific target (what is the result to be achieved by software); 2) choosing the most appropriate method (programming language) for achieving defined end; 3) writing specific list of steps (commands) needed for a software to function in intended way (source code); 4) translating (compiling) the source code into object, i.e. binary (machine readable) code. When implementing first two phases, „the programmer appears to perform work similar to the inventor, i.e., trying to find a solution to a specific problem."[9]

In legal acts both at national and international level[10] the protected object is „computer program". In the United States of America (U.S.) the term „computer program" in Copyright Act is defined as „a set of statements or instructions to be used directly or indirectly in a computer to

[6] M. Paterson, „Properly Protecting Code: Solving Copyright and Patent Rights Overlap via Computer Software Suitability in Copyright," *IPJ*, (Intellectual Property Journal) Vol. 25(2), 2013, p.188.
[7] Paterson, *supra* note 6, p.174.
[8] Paterson, *supra* note 6, p.189.
[9] C. Heath and Anselm K. Sanders, „*New Frontiers of IP Law. IP and Cultural Heritage, Geographical Indications, Enforcement and Overprotection*," (United Kingdom: Hart Publishing, 2005), p.161.
[10] World Intellectual Property Organization (WIPO), Copyright Treaty, available on: https://wipolex.wipo.int/en/text/295157. Accesses April 3, 2020.

bring about certain results.”[11] Due to the high speed of obsolescence of any definition in such a field[12], Directive 2009/24/EC of the European Parliament and of the Council of 23 April 2009 on the legal protection of computer programs (Software Directive), avoids any sort of definition, only providing the recognition of legal protection to computer programs and their preparatory design material.[13] Although legal definition for concept of preparatory design material is nonexistent, recital seven in Software Directive explains that substantial indication for a material to be regarded as preparatory is „that a computer program can result from it at a later stage.”[14] Advocate General Mr. Bot in his opinion in case C-393/09 explained that such

> Design (...) can include (...) a structure or organizational chart developed by the programmer which is liable to be re-transcribed in source code and object code, thus enabling the machine to execute the computer program.” In practical terms for preparatory design material to be useful, it needs to contain „detailed flowcharts, drawings, program specifications, algorithms, formulas, etc.

1.2. Source code as an instructive part of a software

Because software's commands to a circuit can relate only one of two messages („on” or „off”"), the language in which commands are expressed is binary-i.e., it has just two characters: the number one, which represents „on,” and the number zero, which represents „off.” „These long strings of ones and zeros that a computer executes are known as a program's object code.”[15]

In nowadays though programmers do not write programs in binary/object code format since it is a time consuming and expensive way to write a program even for simplest functions. Instead, they use high-abstraction languages such as C, C++, Java, Python and others and the computer code written in one of those languages is called „source code”. Despite the fact that it is still unintelligible to non-programmers, „but it resembles English much more than object code does and is accordingly easier to write and comprehend.”[16] Even though it’s primary purpose (from the technical point of view) is to generate ready-to-install binaries[17] as binary code takes up significantly less space from computers memory than the source code, the latter is the most

[11] 1 U.S.C. § 101 (1976).
[12] Heath and Sanders, *supra* note 9, p.161.
[13] Directive 2009/24/EC of the European Parliament and of the Council of 23 April 2009 on the legal protection of computer programs, OJ L 111, 5.5.2009, pp. 16–22. Available on: https://eur-lex.europa.eu/legal-content/EN/LSU/?uri=CELEX:32009L0024. Accessed March 3, 2020.
[14] *Ibid*.
[15] Gibson, *supra* note 5, p.174.
[16] *Ibid*.
[17] https://perma.cc/UAP6-PR22

versatile, informative and permanent form of any software for several reasons. Firstly, having source code of the software allows system administrators while installing software a greater control in tailoring the software to particular requirements. Secondly, it is relatively easy to fix bugs (i.e., errors), find viruses or other malicious content, as well as to enhance or otherwise modify software using the source code, whereas it would be extremely difficult using the binaries. Thirdly, having the source code version of software makes it practical to port the software to other platforms (i.e., develop versions for other processors and/or operating systems). Without the source for a particular piece of software, such porting is usually extremely difficult.[18]

After establishing that software is a critical component needed for a computer in order for it to be able to process data, and that source code is the „key" that allows to modify the software, it is necessary to look back in history, to understand why copyright, eventually, was recognized to be the most appropriate protection regime for software.

2. Legal modes for software protection

Although protecting software under copyright regime might seem obvious choice due to source codes resemblance of literary work as the code is presented in written lines in programming languages such as C, C++, Java and Python, this has not always been the case. The interaction between trade secret, copyright and patents, which all are and have been used in providing legal framework for software protection, is continuous source of confusion.

An overview of historical discussions in the U.S. and the EU which of the aforementioned regime is best suited for software protection reveals the different approaches in civil and common-law legal systems, hence the latter based its regime on the presumption about software's utilitarian nature, that is to say, „its purpose is to communicate instructions and/or to execute functions in an electronic device"[19], while the former held strong traditions of droit d'autor[20]. Hence it was not self-evident that with time copyright internationally would be recognized as the most suited regime to protect software.

[18] *Ibid*.

[19] Heath and Sanders, *supra* note 9, p.162.

[20] Stefan Larsson, *Conceptions in the Code. How Metaphors Explain Legal Challenges in Digital Times* (New York: Oxford University Press, 2017), p.86.

2.1. Meandering approach between trade secret, patent and copyright in the U.S. for software protection

The rise of software industry and following disputes on how to best recoup investments made in crating it is attributed to IBM's decision on 23 June 1969 to „unbundle (…) its software [from hardware]"[21] and licensing software separately from its hardware and support services. From this date on, separate markets for computer hardware and computer software existed. Software was now „a stand-alone product which would require legal protection in the marketplace lest any unscrupulous individual attempt to free-ride on the investment of another."[22]

In the light of uncertainty in 1970's whether the software could be protected under copyright or patent law[23], the businesses turned to trade secret as a legal instrument which would serve both of their needs, i.e. distributing software to clients and not losing control over an asset which is most valuable whilst being un-accessible to competitors. Attractiveness of trade secret as a means of software protection can be explained by the fact that trade secret

> involve[s] methods or formulas or know-how that may or may not be eligible for patent protection, but that certainly do not qualify for copyright with its proscription against protecting ideas or factual information[24].

Notwithstanding the contradiction between making software available to public through licensing and at the same time keeping it secret through non-disclosure clause in the same agreement, courts in U.S. where supportive to such an approach.[25]

Entangled connection between protecting object and source code of the software as a trade secret, whilst also granting it a copyright protection in the U.S. can be explained by several reasons which are necessary to comprehend since the need for software source code escrow arises out of its secrecy. Although the overlap of trade secrets and copyright is not unusual occurrence since the „expression eligible for copyright can contain the kind of commercially valuable material that is covered by trade secret"[26], when the market for software as a stand-alone product arose, U.S. legislators were in a process of joining the Berne Convention for the Protection of

[21] Andrew Murray, *„Information Technology Law: The Law and Society,"* 4th edition (New York: Oxford University Press, 2019), p.219.

[22] *Ibid*.

[23] *See* Rice, A. David, „Whither (No Longer Whether) Software Copyright," Rutgers Computer & Tech. L.J. (Rutgers Computer & Technology Law Journal) Vol.16, Issue 2 (1990), p.342.

[24] Diane L. Zimmerman, „Trade Secrets and the 'Philosophy' of Copyright: A Crash of Cultures", (July 2009), available on: file:///C:/Users/madarastorha.AT/Downloads/SSRN-id1438706.pdf. Accessed February 15, 2020.

[25] *See supra* note 23, p.344.

[26] *Supra* note 25.

Literary and Artistic Works of September 9, 1886 (Berne Convention) in order to align its protection for copyrighted works with the rest of international society. Article 2 (1) of Berne Convention grants copyright protection for literary works whatever may be the mode or form of their expression (i.e., fixation of a work is not mandatory requirement to obtain copyright protection, however, part (2) of the relevant Article provides contracting states with a right to implement such prescription in their legislation).

The U.S. copyright law regime, on the other hand, was conferring copyright protection to published works, which as a matter of fact, needed to be registered with Copyright Office in order for authors to be able to bring claim for statutory damages in case of infringement. The need to register artistic and literary works with Copyright Office as a prerequisite for granting an effective copyright protection is linked to the principle which underpins American copyright regime – promotion of learning or public access principle, because copyright „was designed as a system of incentives to encourage authors to make their work publicly available."[27] The question then arises – how is it possible that software is being protected under both – trade secret and copyright?

Before copyright was considered a proper means for software protection from the so called free-riding, software's integration with hardware, coupled with the absence of protections in copyright and patent law, led to an initial focus on trade secrecy and contract law for protection, what some have described as the first phase of software protection under intellectual property law.[28] Now Justice Stephen Breyer in 1970, discussing proposal to amend Copyright Act by conferring in it a protection for computer programs, expressed considerable doubts whether software, especially the kind of „off the shelf" should enjoy copyright protection, reasoning that

> system software's (the ones who control the basic operations of the computer) are sold together with hardware for a single price and in this case it is unlikely that unauthorized copying could occur. Secondly, most application programs (solving particular problems) are tailored to suit individual customer needs. And thirdly, application programs that are usable generally are not sold directly „off the shelf", but rather they are bundled together with certain services from software developer such as fixing errors („bugs") in the program, providing updates from time to time and making adjustments in order the software be compatible with other programs in computer which

[27] *Ibid*.

[28] *See* Stephen Breyer, „The Uneasy Case for Copyright: A Study of Copyright in Books, Photocopies, and Computer Programs," *Harv. L. Rev.* (Harvard Law Review), Vol. 84, Issue 2 (December 1970), pp.281-351.

> means that significant part of business revenue is made from providing services, rather than the software.[29]

Concerns mentioned above where not persuasive enough and in 1976 Congress enacted Copyright Act which came to force in 1978 and its §101 states that computer programs are protected under copyright. This was not the only change. As mentioned above, U.S. in order to accede to Berne Convention, amended the aforementioned act by extending copyright to unpublished works. In scholarly writing it is argued that such a decision unintentionally „denigrated the importance of access principle"[30], since these changes in legal regulation offered the benefits of copyright to works that were intended to be kept secret. Even though the Berne Convention provides copyright protection for literary works only after they have been fixed in some kind of material form and the U.S. copyright system theoretically acknowledges copyright without any formal registration, the provision that only those authors/owners who register they works are entitled to statutory damages in practice means that the registration with state agency is necessary in order to have enforceable copyrights.

Given the fact that by the time the changes in Copyright Act were being discussed software industry had already actively utilized the benefits of trade secret regime, conferring copyright protection upon the software inevitably caused tension in regards to question how would industry players register their creations with Copyright Office, which is obliged to keep publicly available registrar of all works submitted to it. To put it differently – the adoption of copyright regime even though anticipated by industry, would require that software developers submit their products for registration, thus make available to public their most valuable asset – source code (registration using object code would not be feasible since it is not intelligible to humans as it reflects commands for hardware in binary form). If source code for software from such manufacturers as Microsoft and IBM would become available to public in 1970's, it is reasonable to argue that there would be no need for source code escrow agreement, since the part of the software which allows skilled programmer to maintain the program, fix it or even create a new one, would be in public domain. Due to the fact that there exists significant number of escrow agents operating, it can be inferred that neither conferring copyright protection to software, nor making such rights enforceability dependent upon registration with Copyright Office did not lead to massive registration, i.e., publication of software source code. Explanation to this somewhat strange outcome can be attributed to Copyright Office's practice in the mid-

[29] Breyer, *supra* note 28, p.345.
[30] Zimmerman, *supra* note 24.

1960s by applying the „rule of doubt" principle to allow for registration of software, because „computer programs were readable, written works of authorship."[31] Even more, as the Register of Copyrights, Ralph Oman explained around that time

> The Office originally asked for [protectability of] source code, because that best represents the copyrightable authorship. But many copyright owners say that the source code version of a program contains valuable trade secrets (…). So the Office gave special relief to allow registration without disclosing trade secrets. Usually, we accepted an abbreviated deposit or a deposit with the trade secret material blocked out.[32]

Although such an approach might seem contradictory to general requirement of Copyright Act (prior to changes made to it in 1909) that „(…) deposit of a work for which protection was sought had to occur by the date of publication or the copyright would not issue at all"[33], in 1909 the previously mentioned act went through alterations regards to Copyright Office duty to retain all the works deposited. As a result of alterations made to Copyright Act in 1909, deposit was abandoned as a criterion for a work to be protected, and „the Library of Congress and the Copyright Office were (…) offered (…) freedom to modify the deposit requirements (…)."[34] There were two reasons for that. The first one was preventing situations where obtaining copyright failed simply by virtue of not getting the depository to Washington in a timely way, that is, before publication. The second reason to abolish depositary system was the lack of space in library. In order to assure publishers that they would still have a proper system in place to prove the date of origin of copyrighted works, „the Copyright Office was given the authority to issue a certificate of registration."[35] The end result where software can be protected simultaneously by trade secret and copyright can be explained by following occasions: (i) declining depositary system in 1909 in order to accommodate increasing amount of copyrightable works to be stored at Copyright Office and replacing it by certificate of registration; (ii) „rule of doubt" practice in mid-1960s in Copyright Office regards to registering software without acquiring and examining source code; (iii) 1976 amendments in Copyright Act applying copyright protection to software when in fact it was practiced by Copyright Office even before modifications in the law and offering copyright protection since the moment of fixation, not publication.

[31] Sonia K. Katyal, „The Paradox of Source Code Secrecy," *Cornell L. Rev*. (Cornell Law Review), Vol. 104, Issue 5 (July 2019), p.1199.

[32] Katyal, *supra* note 31, p.1202.

[33] Zimmerman, *supra* note 24.

[34] *Ibid*.

[35] *Ibid*.

Due to the fact that in mid-1960s there was no consensus under which mode of intellectual property to protect software, the Patent Office was presented with question whether to issue patents for programs and for processes embodied in programs[36], since it perceived programs to be un-patentable printed matter and program processes to be un-patentable mental processes.[37] The ability of Patent Office to issue patents for software was encumbered also from practical point of view since the Patent Office lacked an appropriate database of prior art to determine novelty and non-obviousness and a suitable classification system.[38] The beginning upsurge of patenting software programs in mid-to-late 1980s was due to Supreme Court's 1981 decision in Diamond v. Diehr in which the Supreme Court extended patentability to a process of curing rubber that relied, in part, on a computer program. It was then further continued by Federal Circuit court decision in 1994 in re Alappat in which it stated that program instructions from software essentially transformed the machine from a „general purpose computer" into, in effect, a „special purpose computer" and that's why computer program is patentable. It is argued that the result of Alappat decision was upsurge of machine patent claims:

> calculators on a computer, auctions on a computer, financial management on a computer, as if each computer running a program were some new device that had been invented.[39]

The peak for patenting computer programs was Federal Circuit decision in 1998 in State Street Bank & Trust Co. v. Signature Financial Group, Inc., concluding that mathematical algorithms, previously dismissed as an abstract concept, could be patentable if they „transformed" a machine or were „performed" by a machine and provided „useful, concrete, and tangible" results.[40] The rapid extension of protecting software through patents lessened in 2010 when Federal Circuit in re Bilski repudiated patenting of a method that „simply comprised a computerized representation of some fundamental principles of financial risk and liability."[41] However software continued receiving patent protection, since the aforementioned court decision did not provided Patent Office and lower courts with clear guidance about possibility to grant

[36] P. Samuelson, „Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement," *Berkeley Tech. L.J.* (Berkeley Technology Law Journal), Vol.31, Issue 2 (2016), p. 1226.
[37] Samuelson, *supra* note 36, p.1228.
[38] Katyal, *supra* note 31, p.1217.
[39] Michael Risch. „Hidden in plain Sight," *Berkeley Tech. L.J.* (Berkeley Technology Law Journal), Vol.31, issue 3 (2016), p.1642.
[40] *See* Katyal, *supra* note 31, p. 1219.
[41] Katyal, *supra* note 31, p.1222.

patent protection to computer programs; it did merely rained on „business methods."[42] After three years the Supreme Court delivered its decision in Alise Corp. in which

> the patentee claimed a method of settling escrow accounts by keeping „shadow accounts" that tracked the results of accumulated transactions-sometimes one party spent more, sometimes the other party spent more. At the end of the day, the shadow accounts would be compared and reconciled, with any difference paid out of the actual escrow account.[43]

Rejecting the possibility to patent software for such a „shadow account" the Supreme Court reasoned that it „did not amount to 'significantly more' than just 'the abstract concept of managing risk with the use of a computer.'"[44]

2.2. Software protection in the EU – looking across the Atlantic

Considerations reflected in the 1988 Green Paper on Copyright and the Challenge of Technology from the Commission of the European Communities discloses concerns of European Economic Community that „European software industry was losing to its competitors from U.S."[45], since the biggest software providers in the Member States where from there. One reason for insufficient ability of the industry to compete with its counterparts was the lack of uniform legal framework for computer programs. Given the fact that by 1980 U.S. had become the first country in the world to grant software protection under copyright and the World Intellectual Property Organization (WIPO) in 1983 had abandoned its initiative for sui generis regime for computer program protection, the aforementioned political document suggested that Community too should provide legal framework for software protection within existing regime of copyright.

It should be noted that neither in that time in European Economic Community, nor now in the EU there is no such thing as unified Copyright Code. Instead, Community opted for regulating computer programs through directives – legally binding acts (prescriptions) to Member States, allowing though each state freedom to choose exactly through what means will it achieve goals set out in the directive. As a result of ongoing discussions in European Economic

[42] Risch, *supra* note 39, p.1644.

[43] *Ibid*.

[44] Katyal, *supra* note 31, p. 1223.

[45] *See* A. Lucas, „Copyright in the European Community: The Green Paper and the Proposal for a Directive Concerning Legal Protection of Computer Programs," *Colum. J. Transnat'IL.* (Columbia Journal of Transnational Law), Vol.29, Issue 1 (1991), pp.145-168.

Community, on how to best provide uniform protection for computer programs in all Member States, on May 16th 1991 the Council Directive 91/250/EEC of 14 May 1991 on the legal protection of computer programs entered into the force and was later (in 2009) repealed by Directive 2009/24/EC of the European Parliament and of the Council of 23 April 2009 on the legal protection of computer programs (Software Directive). It stated that Member States shall protect computer programs, by copyright, as literary works within the meaning of the Berne Convention for the Protection of Literary and Artistic Works and that for purposes of the directive the term „computer program" shall include their preparatory design material.[46] Both, the first and the following directive in their recitals recognizes that the protected subject matter is „the expression of a computer program", and that ideas and principles which underlie any element of a program, including those which underlie its interfaces, are not protected. It is specifically emphasized that to the extent that logic, algorithms and programming languages comprise ideas and principles, those ideas and principles are not protected under copyright.

The explicit language in Art.1 of Software Directive and considerations reflected in its preamble gives the impression that patent protection was disregarded as a valuable option to grant computer programs legal protection. But, for a neutral observer, it would seem improbable that Community, whose aim, while introducing unified legal framework for computer program protection, was, in a significant part, to assure the competitiveness of industry with the one in U.S., would ignore the fact that during the mid-1980s and 1990s there was a growing practice in the U.S. to grant software also a patent protection. Such an observation would be well founded. At the European level it influenced the European Patent Office (EPO) praxis, „bringing it to adopt a less restrictive attitude in granting patents via the use of sophisticated legal interpretations"[47] of Article 52 of the Convention on the Grant of European Patents.

The aforementioned convention (hereinafter European Patent Convention) was adopted on the 5th October 1973 and was signed as a „special agreement" under the Paris Convention. Its aim was „to establish „a system of law, common to the Contracting States, for the grant of patents for inventions""[48], and a European Patent Office (EPO) to administer it.

[46] *Supra* note 13.

[47] Paolo Guarda. „Looking for a feasible form of software protection: copyright or patent, is that the question?", *E.I.P.R.* (European Intellectual Property Review), Vol. 35, Issue 8 (2013), p.449.

[48] Justine Pila, „Software Patents, Separation of Powers, and Failed Syllogisms: A Cornucopia from the Enlarged Board of Appeal of the European Office," *Cambridge L.J.* (Cambridge Law Journal), Vol.70, Issue 1 (March 2011), p.203.

Article 52 (1) of EPC states that European patents shall be granted for any inventions, in all fields of technology, provided that they are new, involve an inventive step and are susceptible of industrial application. The phrase „in all fields of technology" was added at the EPC 2000 Revision Conference to reflect the wording of Article 27.1 of the Agreement on Trade-Related Aspects of Intellectual Property Rights (TRIPS), which in its turn was caused by U.S. case law with „regard to software patentability."[49] Nevertheless, EPC Article 52 (2) states that schemes, rules and methods for performing mental acts, playing games or doing business, and programs for computer shall not be regarded as inventions within the meaning of paragraph 1. The substantiation for statement that in order to accommodate for software patenting EPO adopted less restrictive attitude towards them by applying sophisticated legal interpretations lies in EPC's paragraph 3 of Article 52 according to which paragraph 2 shall exclude the patentability of the subject-matter (…) referred to therein only to the extent to which a European patent application or European patent relates to such subject-matter (…) as such.[50] Outcome of such praxis was

> the exclusion from European patentability of computer programs and business methods as such does not prevent the grant of a European patent for an automated method of commercial transaction, nor for any other computer-implemented method.[51]

Due to the facts that: (i) often if not always algorithm components are the most valuable part of the software; (ii) there was a need to safeguard competitive position of European software industry in relation to its major trading partners by means of eliminating current differences in the legal protection of computer implemented inventions; and (iii) under EPC article 52 (2) (3) programs for computers are excluded from patentability „as such", and EPO's praxis was that they did not issued software or business methods patents, but rather granted patents for computer implemented inventions (CII)[52], in 2002 the European Commission, aiming to harmonize patent application practices in the Union, initiated adoption of Directive of the European Parliament and of the Council on the patentability of computer-implemented inventions (2002/0047(COD).

This proposal was rejected in the face of strong opposition from open source community, coalition of commercial companies and non-profit associations promoting open source software –

[49] *Supra* note 47, p.449.

[50] The European Patent Convention. Available on: http://documents.epo.org/projects/babylon/eponet.nsf/0/17756BCD9D57462FC125853B0053E438/$File/EPC_16th_edition_2016_en.pdf. Accessed April 13, 2020.

[51] Pila, *supra* note 48, p.206.

[52] A computer-implemented invention (CII) is one which involves the use of a computer, computer network or other programmable apparatus, where one or more features are realized wholly or partly by means of a computer program. Available on: https://www.epo.org/index.html. Accessed March 7, 2020.

EuroLinux Alliance, and also small and medium size businesses, arguing that adoption of the Directive would stifle the competition and in fact it favors big U.S. software companies such as Microsoft, IBM and Apple, by allowing to introduce in Europe abusive software patent praxis existing in the U.S. On July 6, 2005, the European Parliament with a large majority rejected the Directive of the European Parliament and of the Council on the patentability of computer-implemented inventions and the European Commission announced it would not draft a new version.[53]

2.3. Inconsistencies in the U.S. and the EU regarding software protection as a favorable ground for software escrow

Although arguments set out above on how the common-law and civil-law approaches software protection reveal some differences between these two systems on approaching the question (which in authors view stems from different legal traditions), the fluid vagueness and changeability of legal framework on how to best protect software discloses its twofold nature and hence the hurdles with selecting the most appropriate regime. Software's dual disposition comes from the fact that despite the importance and value of text within code

> the most important property of program is their behavior (i.e., the set of results brought about when programming instructions are executed). In this sense it is arguable that the text found in computer software is solely functional and utilitarian, and not expressive enough to attract copyright protection.[54]

Whereas software in its nature is functional and aimed at performing highly practical tasks, that is to say it is a means to an end not an end in itself, it is hard to overlook the tension between its functional quality and all-embracing principle of copyright to protect creative (i.e., imaginative) works from unauthorized distribution and copying. In other words, while copyright initially was designed to protect artistic and original expressions in such customary works as books and music, software is different by its nature because in order to be acknowledged by the public/consumers, it does not need to be thrilling, fascinating or at least interesting. It needs to be workable[55] and its primary task is not to stir emotions.

[53] *See* Roy J. Rosser, „European Software Patents - When a Rejection is Not a Rejection," IPL Newsl. (IPL Newsletter), Vol.24, Issue 1 (Fall 2005).

[54] Paterson, *supra* note 6, p.187.

[55] „The idea of „tight code" is a key feature in software programming. The ideal in coding is to perform a necessary function in the least amount of lines of code while utilizing standardized structuring and indentations, thus the best method of computer programming is the most efficient." Paterson, *supra* note 6, p.189.

Even though it is an international legal approach that software is protected under copyright,[56] in praxis it's most valuable part – source code, additionally is protected also as a trade secret (or patent) and public interacts with software in such a form which does not constitute „copyrighted work" in traditional meaning of the phrase. It can be explained by the fact that the end user does not need source code in order to use software for its intended purpose. Software developers supply it to their customers in object code format – a string of ones and zeros which reveals nothing about the structure and functionality of software or algorithms embedded in it, but does the main work – makes the computer to perform elected functions. The component allowing for creation of a new or similar software – source code – written in highly abstract programming language and intelligible for reasonably skilled programmers, as it is regarded as a „copyrightable trade secret", is practically never delivered to an end user together with object code in fear of unauthorized copying and depriving software developers of their revenue. Thus it can be concluded that even though general understanding is that both, source and object code is protected under copyright, in actuality the part of the software that is available/delivered to public, i.e., buyers, - the object code, as it stands, is imperceptible to them and a need to give copyright protection to strings of ones and zeros seems unreasonable. The part that in theory qualifies for copyright protection (due to its visual resemblance with written text in analogue language) – the source code, is never presented to general public, because that would increase the possibility of competitors copying ideas/solutions for practical tasks implemented in it. In the U.S. such a situation was made possible abandoning the need to publish the work in order to gain copyright protection, whereas in European Union although copyright is not entirely harmonized, Berne Convention is binding to all Member States and it does not require to make a „work" publicly available, and also the fixation in any material form as a prerequisite for work to be protected under copyright is rather an exception, not a norm. This, in turn, created the need for (mainly) businesses to make sure that their mission critical infrastructure – software's of different kind, are disturbance proof.

Intricacies faced by software users extensively can be separated in to two groups – the software developer ceases to exist (goes out of business due to insolvency or is reorganized and the successor in rights does not wish to continue providing the software in question) or is unable to maintain software edition/provide maintenance necessary for specific business. In case such circumstances arise considerations regarding business continuity from software user viewpoint

[56] *See* Gibson, *supra* note 5, p. 171.

are the main reason to contract for software source escrow agreement – usually a supplementary contract towards software license. There had been assertions that such escrow arrangement is also in the interest of software developers whereas it can serve as a proof of first creation since escrowed source code contains date of receipt, but considering the fact that literal software copying occurs much more rarely than copying of software's structure, sequence and organization, this argument should be treated with skepticism and most likely the argument why software developers agree to put their most valuable asset – the source code, into escrow agreement with third party, is their will to assure software users against any threats to business continuity.

Following chapter of this paper will explore model of software commercialization – licensing, to better understand the possible impediments with source code escrow agreement as it serves as ancillary tool to ensure business continuity for entities, who rely on licensed software in their business operations.

3. The legal nature of software license

Licenses for all kinds of intellectual property, including the one covered by copyright, „are the primary means by which intellectual property is shared in modern economies."[57] By exploring how licensing became most common means for software distribution, and what are the contractual rights conferred on licensee in such a model, i.e., what is the object that the licensee acquires as a result of this transaction, author provides insight needed to better assess effects that the bankruptcy of software licensor imposes on contractual rights of the licensee.

As described in previous part, market for software existed before there was comprehensive understanding at national and international level that copyright is the most appropriate legal regime to protect it. Due to the fact that no clear statutes existed as for some special software protection regime, but software vendors where interested in distributing their products and recouping investments made, the solution for the problem was found in combining contract law with trade secret. Given the fact that the idea behind software concept is to automate previously manually performed tasks and automation can be expanded across unlimited number

[57] Andrew Dietderich, Thomas Bialson and Robin Phellan, „Treatment of Intellectual Property Licenses in Cross-Border Insolvency Proceedings", available on: https://www.uncitral.org/pdf/english/congress/Papers_for_Congress/118-DIETDERICH_and_GAA_and_PHELAN-Intellectual_property_licences_in_cross-border_insolvency_proceedings.pdf. Accessed March 25, 2020.

of industries, software vendors where facing challenge – how to distribute the same software (or with non-essential and easy to manage changes) to as many buyers as possible, if the prevailing wisdom was that „that the protection for computer software programs is precluded by the principle that mathematical formulas and algorithms were not patentable."[58] To put it differently, contract law alone could not preclude the buyers from distributing the software further down the line (what would be permissible according to first sale doctrine under which individual who buys copy of copyrighted work „may use and resell that particular copy free of any restraint by the copyright owner"[59]) and that way reducing software vendor's market share. Solution was found in trade secret:

> A trade secret may be disclosed to others without losing its protected status, as long as the persons to whom it is disclosed agree that they will not themselves disclose it.[60]

On the other hand, arrangements that purports to circumvent common-law tenet that „„„restraints on alienation" are generally unenforceable."[61] Discord between non-disclosure and restraints on alienation was solved by lawyers by introducing the software license.[62]

Due to the fact that possibilities for software application where first explored in U.S., and the biggest software vendors offering their products in Europe also came from the U.S., it is understandable that licensing became the accepted legal vehicle for software distribution also in Europe. Although EU does not have such doctrine as a „first sale doctrine", the question at issue, that is, the trade-off between interests of rights holders and the general public, is approached by applying principle of exhaustion, according to which:

> When a copy of a copyright-protected work (or other related subject matter) is sold or otherwise put on the market by the copyright holder or with his or her consent, the right holder may not invoke his or her exclusive right to control the further distribution of the copy.[63]

In the case UsedSoft GmbH v. Oracle International Corp. the Court of Justice of the European Union had an opportunity to clarify the question whether a license allowing perpetual

[58] Thomas M. S. Hemnes, „Restraints on Alienation, Equitable Servitudes, and the Feudal Nature of Computer Software Licensing," *Denv. U. L. Rev.* (Denver University Law Review), Vol. 71, Issue 3 (1994), p.578.

[59] H. Ward Classen, „Fundamentals of Software Licensing," *IDEA* (IDEA: The Journal of Law and Technology), Vol.37, Issue 1 (1996), p.3.

[60] Hemnes, *supra* note 58, p.579.

[61] *Ibid*.

[62] Hemnes, *supra* note 58, p.580.

[63] Ole-Andreas Rognstad, „Legally Flawed but Politically Sound: Digital Exhaustion of Copyright in Europe after UsedSoft," *Oslo L. Rev.* (Oslo Law Review), Vol.1 (2014), p.1.

use of computer program for „remuneration corresponding to the economic value of the copy"[64] amounts to sale that exhausts owner's distribution rights. By giving an affirmative answer, the court held that the usual meaning of a sale – it is a transfer of ownership title, can be applied to a license agreement by which transferee is granted a right to use software for an unlimited period.[65]

Although licensing has become generally accepted way for distributing software, the question, what is the legal nature for software license agreement – is it a sale, a lease or a „true license" remains open both in the U.S. and EU, and this becomes especially relevant when software licensor (owner) faces financial difficulties and software license agreements are included in debtor's estate due to its insolvency. Nature of the agreement through which intellectual property (software) is being disseminated into economy is important aspect in insolvency context since maximizing the value of debtor's assets is one of the objectives for insolvency procedures. United Nations Model Law recognizes that:

> Achieving the objectives of maximizing the value of the estate and reducing liabilities and, in reorganization, enabling the debtor to survive and continue its affairs to the maximum extent possible in an uninterrupted manner may involve taking advantage of those contracts which are beneficial and contribute value to the estate (including contracts that will enable the continued use of crucial property that may be owned by a third party) and rejecting those which are burdensome or those where the ongoing costs of performance exceed the benefit to be derived from the contract.[66]

If there are no clear guidelines on what kind of contract is software license agreement, the aforementioned and established principle that ongoing contracts to which debtor is party, may be revoked by administrator, if that would maximize value of debtors assets, becomes especially important, since software licenses in most cases sell nothing to licensee.[67] In common law it is rather a software developers promise not to sue licensee for using its intellectual property as long as the latter adheres to contract clauses, regulating permitted use, while in the civil law countries software license agreement is regarded as a sale of use rights which does not amount to ownership.

[64] Brandon A. McKenzie, „License to Sell: Kirtsaeng v. John Wiley&Sons, UsedSoft GmbH v. Oracle, and the impending disruption of the international software licensing landscape," available on: file:///C:/Users/madarastorha.AT/Downloads/SSRN-id3356775%20(1).pdf. Accessed April 2, 2020.

[65] *Ibid*.

[66] United Nations Commission on International Trade Law. Legislative Guide on Insolvency Law, available on: https://uncitral.un.org/sites/uncitral.un.org/files/media-documents/uncitral/en/05-80722_ebook.pdf. Accessed April 2, 2020.

[67] *See* Hemnes, *supra* note 58, p.580.

Some have compered software licenses to a fief because they like license agreements were „an effort to separate a right of possession from the right of alienation."[68] Although common clauses in software license agreements prohibiting licensees to alienate its „rights without the consent of the licensor"[69], emphasizes personal character of relationships between licensor and licensee, they seem to have no real strength in cases where license agreements contain such clauses as „fully paid up, royalty free, perpetual". Disparity between what the notions in license agreement say it does (for example, prohibits licensee from assigning license to third party without licensors permission) and the factual situation in which licensee has received customized software and the payment was designed as a lump sum, brings up the question can such a license really be distinguished „from the sale of a copy."[70] The paradox between software license as an instrument that grants nothing more to licensee „than a bailment of the copy of the software"[71], on the one hand, and the reality in which courts interpret software as a goods sold under Article 2 of Uniform Commercial Code[72], on the other hand, demonstrates the so called software licensing dilemma:

> If software is sold and not licensed, the licensor's ability to control unauthorized uses of its product is significantly curtailed; on the other hand, if software is licensed and not sold, the licensee's rights under the agreement are unduly restricted.[73]

This dilemma stems from sui generis nature of software and question whether software is sold or leased is in direct causality with the ability of software license to survive (or not) the bankruptcy of the licensor, as presumably a sales contract under which the title to goods (software) has been transferred could be considered as insolvency proof with higher probability than a license agreement with no transmission of title. The application of Article 2 of U.C.C. to software license agreements[74] from the practical standpoint is explained by the fact that „(...) court's lack of a better alternative."[75] Such an approach have been criticized by scholars since

[68] *Ibid*.

[69] Hemnes, *supra* note 58, p.586.

[70] Hemnes, *supra* note 58, p.590.

[71] Hemnes, *supra* note 58, p.580.

[72] „Article 2 applies to transactions of goods, offering the parties to sales agreement extensive contractual rights, protections and limitations." *See* Michael L. Rustad and Elif Kavusturan, „A Commercial Law for Software Contracting," *Wash.& Lee L. Rev.* (Washington and Lee Law Review), Vol. 76, Issue 2 (Spring 2019), p.778.

[73] Nancy S. Kim, „The Software Licensing Dilemma," *BYU L. Rev.* (Brigham Young University Law Review), Vol.2008, Issue 4 (2008), p.1103.

[74] Unlike the sale of goods, licensing does not involve the passage of title from the licensor to the licensee. *See* Kim, *supra* note 73, p.1162-1163.

[75] Rustad and Kavusturan, *supra* note 72, p.780.

Article 2 of U.C.C. „defines a sale as the passing of the title from seller to buyer for a price”[76], and as it has been shown earlier in this paper, software license, at least from the software licensor’s perspective, does not transfer title, regardless of price paid for it. Not only that, Article 2 of U.C.C. deals with sale of goods, but software in itself (if we exclude the physical medium on which it is carried) could hardly be considered a movable thing in the abstract.

The next subchapter will look at how software license agreements treated in software vendors insolvency procedure are, and, depending on the outcome of this process, what results can a licensee – software source code escrow agreement beneficiary, expect.

3.1. Validity of software license agreement when licensor becomes bankrupt under U.S. Bankruptcy Code

In the light of previously mentioned administrator’s right to revoke debtor’s contracts which, outside the insolvency proceedings, would be left intact, and the corresponding need to maximize debtor’s assets in order to fulfill creditor claims, U.S. courts have been dealing with a concept of „executory contracts” which then may be rejected by trustee (administrator) or debtor in possession. Even though no statutes contains definition, courts have been adhering to Professor Countryman’s definition according to which for bankruptcy purposes executory contract is such

> a contract under which the obligation of both the bankrupt and the other party to the contract are so far unperformed that the failure of either to complete performance would constitute a material breach excusing the performance of the other.[77]

The question whether software licenses can be treated as executory contracts in insolvency procedures is topical if a software in matter is customized and licensed exclusively to licensee because it indicates that licensee has made serious financial investment in obtaining it.

Further analysis will deal with question what happens with licensee’s right to use licensed software if a licensor becomes insolvent under Chapter 11 of US Bankruptcy Code (Bankruptcy Reform Act of 1978).

Chapter 11 provides regulation for the reorganization of a corporate debtor. Usually it allows for the management of the debtor to stay in control („debtor in possession”) which means that debtor gets to make operative business decisions. Trustee can be appointed by a court in case

[76] Stacy-Ann Elvy, „Hybrid Transactions and the INTERNET of Things: Goods, Services, or Software,” Wash. & Lee L. Rev., Vol.74, Issue 1 (Winter 2017), p.128.
[77] Viktoria L. Gres, „Rejection of Computer Software Licensing Agreements in Bankruptcy,” *Cardozo L. Rev.* (Cardozo Law Review), Vol. 8, issue 2 (December 1986), p.375.

there is „a finding of fraud, dishonesty, incompetence or gross mismanagement of the affairs of the debtor by current management."[78] As commencement of a case under any chapter of the Bankruptcy code „creates an estate comprised of all legal and equitable interests of the debtor in property"[79], licenses issued by licensor to third parties (licensees) also are drawn into estate of debtor in possession. Section 365(a) of Bankruptcy Code provides that (...) the trustee, subject to the court's approval, may assume or reject any executory contract or unexpired lease of the debtor. So, the question whether software licenses are executory contracts are of crucial point since in case of an affirmative answer they can be rejected by the debtor in possession and licensee loses its rights to use licensed intellectual property regardless of the fact that agreed-for term has not became due. Software license agreements generally are treated by courts as executory contracts

> Where there are ongoing, material obligations on both sides, such as the duty to indemnify, pay royalties, maintain confidentiality, provide updates, or adhere to quality standards.[80]

Even though licenses in most cases are treated as executory contracts, there are conditions under which it might not be so, for example, if one party had fully performed its material duties under a particular license agreement, the license agreement is no longer executory[81]. It could be argued that if the only obligation left is licensees duty to pay royalties, such contract would not be treated as executory.

The first case dealing with question whether licensee has a right to continue to use licensed intellectual property, if licensor has rejected license, was Lubrizol Enterprises, Inc. v. Richmond Metal Finishers, Inc. in 1985. It involved nonexclusive license of a metal coating process technology from Richmond Metal Finishers to Lubrizol Enterprises. Richmond Metal Finishers acting as a debtor in possession regarded license given to Lubrizol Enterprises as a burdensome executory contract and rejected it. Although Lubrizol Enterprises contested licensor's decision in court, the final outcome was in favor of the licensor. Rejection of license agreement as an executory contract was permitted on the grounds that both parties – the licensee

[78] Carl Felsenfeld, *International Insolvency. Part II. The Insolvency Laws of Major Countries,* (New York: Fordham Law School, 2000), U.S-6.
[79] Felsenfeld, supra note 78, U.S.-49.
[80] Ron E. Meisler, Elaine D. Ziff, Tracy C. Gardner, Carl T. Tullson, „Rejection of Intellectual Property License Agreements Under Section 365(n) of the Bankruptcy Code: Still Hazy After All These Years," available on: https://docplayer.net/9961810-Rejection-of-intellectual-property-license-agreements-under-section-365-n-of-the-bankruptcy-code-still-hazy.html. Accessed May 29, 2020.
[81]*See Ibid.*

and the licensor, had minimal contingent duties towards each other, and the bankrupt licensor would benefit from rejecting nonexclusive license.[82] Even though the case was about rejection of patented technology license, not a software license, the software industry anticipated that the same scenario could be applied to software license agreements because fee payments commonly are structured as ongoing and licensor has a duty to refrain from suing licensee for copyright breach as long as licensee complies with permitted use conditions. To put it differently, Lubrizol Enterprises, Inc. v. Richmond Metal Finishers, Inc. demonstrated that software licenses very easily could be regarded executory contracts under Chapter 11 bankruptcy proceedings and this, in turn, meant that every licensee, whose licensor become bankrupt, would lose rights to use licensed software.

> The (...) inequities of section 365(a) wrought on the licensee by the Fourth Circuit in Lubrizol v. Richmond Metal Finishers[65] and the refusal of the Supreme Court to grant certiorari led Congress to pass section 365(n), to strike a balance between licensor rehabilitation in bankruptcy and the need to preserve a licensee's contractual rights to the software or other qualified intellectual property.[83]

Congress's intent, when implementing Intellectual Property Bankruptcy Protection Act of 1987 was to protect licensees' rights to intellectual property in the event of a bankruptcy. To balance these two competing interests, section 365 (n) provides that if the trustee rejects an executory contract under which the debtor is a licensor of a right to intellectual property, the licensee under such contract may elect either to treat such contract as terminated, or to retain its rights (...) to such contract, to such intellectual property (including any embodiment of such intellectual property to the extent protected by applicable non bankruptcy law).

If the licensee elects to retain its rights after debtor in possession or trustee has rejected the license, it has to make all royalty payments due under such contract for the duration of such contract, and it (licensee) is deemed to wave any right of setoff it may have with respect to such contract. With respect to licensee's duty to pay royalties for the intellectual property licensed, it would be prudent from licensee's perspective to separate royalty payments for intellectual property use in question from other payments, for example, maintenance and support service fees. This is due to the fact that if the licensee elects to retain its rights under license agreement, which has been rejected by trustee or debtor in possession, it may be required to make payments

[82] *See* James E. Meadows, „Lubrizol: What Will It Mean for the Software Industry," *Santa Clara Computer & High. Tech. L. J.* (Santa Clara Computer and High-Technology Law Journal), Vol. 3, Issue 2 (1987), p.312.
[83] Osei Kingsley, „Pouring New Wine into Old Wineskins: Why on Premise Software Source Code Escrow Arrangements Are Ill-Suited for Remotely Hosted off Premises Software as a Service License Agreements," *J.C. & U.L.* (Journal of College and University Law), Vol. 39, Issue 2 (2013), p.396.

also for the services which are no longer provided to it. In situation where licensee elects to retain its rights, the debtor in possession (trustee) is obliged to provide to the licensee any intellectual property (including such embodiment) held by the trustee, to the extent provided in such contract, or any agreement supplementary to such contract, and not to interfere with the rights of the licensee as provided in such contract, or any agreement supplementary to such contract. However, licensor is not obliged to provide any further services (like software maintenance, debugging and updating) to licensee. A situation in which licensee elects to retain its rights to use software from insolvent licensor who no longer has maintenance and support duties towards licensee, is the one for which software source code escrow agreement was concluded and in such case source code held in escrow shall be released to licensee on the condition that escrow agreement is considered supplementary contract to the license agreement.

Although the response from legislator to legal issues regarding the protection of licensees rights brought up by Lubrizol Enterprises, Inc. v. Richmond Metal Finishers, Inc., was a clear one in that sense that licensee should be able to continue use of licensed software for a term provided in license agreement, scholars have indicated two questions that might still impede the licensees right towards software use. The first one was that Congress did not provided any explanation what does actually rejecting contract in insolvency means. Does that mean a material breach of a contract (license agreement) or does that mean an avoidance of the entire contract „placing the parties in the position they would be in if the contract had never been entered."[84] From the perspective of legal stability it seems reasonable to infer that bankruptcy laws should not give trustee or debtor in possession more legal rights than they would have outside the bankruptcy procedures and that's why it could be concluded that „rejection of executory aspects of a contract does not bring about a rescission of property transfers completed prior to bankruptcy."[85]

Proposed answer regarding the legal nature of contract rejection in the realm of bankruptcy, i.e. that it does not mean rescission, got approved by the Supreme Court of the United States in its decision from May 20, 2019 in case Mission Product Holdings, Inc. v. Tempnology, LLC, NKA Old Cold LLC. Court had to decide whether the debtor-licensor's rejection of trademark licensing agreement under Section 365 (a) deprives the licensee of its rights to use the trademark. By reversing decision of First Circuit, who in turn agreed with

[84] Jennifer S. Bisk, „Software Licenses through the Bankruptcy Looking Glass: Drafting Individually Negotiated Software Licenses That Protect the Client's Interests in Bankruptcy," *Fordham Intell. Prop. Media & Ent. L.J.* (Fordham Intellectual Property, Media & Entertainment Law Journal), Vol.17, Issue 3 (Spring 2007), p.630.
[85] Brisk, *supra* note 84, p.631.

Bankruptcy Court decision that the debtor's rejection must extinguish the rights that the agreement had conferred on the trademark licensee, Supreme Court reasoned that

> Fundamental principles of bankruptcy law command that rejection of a contract – any contract – in bankruptcy operates not as a rescission but as a breach. When rejecting burdensome executory contract, the debtor can stop performing its remaining obligations under the agreement. But the debtor cannot rescind the license already conveyed. So the licensee can continue to do whatever the license authorizes.[86]

Even though case in question involved trademark, not software license, Supreme Court's conclusion that rejection of contract constitutes a breach of contract and this means in the Bankruptcy Code what it means in contract law outside bankruptcy, gives clear indication that intellectual property licenses, including for software, are insolvency-resistant.

3.2. Validity of software license agreement when licensor becomes bankrupt under Insolvency Law of the Federal Republic of Germany

Contrary to previously examined U.S. Bankruptcy Code, Insolvency law (Insolvenzordnung, hereinafter InsO) of the Federal Republic of Germany „does not expressly regulate the effects of insolvency on a software license agreement."[87] Instead, general rules from InsO and Civil Code are the sources to look for guidance on

> One of the most relevant issue discussed in Germany whether the license contract will remain enforceable or not in the event of an insolvency of the licensor.[88]

Section 103 of InsO states that administrator may choose whether to perform or not a mutual contract which was not or not completely performed by the debtor and its other party at the date when the insolvency proceedings were opened. Such provision – in its aim similar to the doctrine of executory contracts in U.S., demonstrates that there are common principles for insolvency and one of which is that ongoing obligations of a bankrupt debtor need to be solved in light of the liquidation purpose. According to legal literature „license agreements will often

[86] *Mission Product Holdings, Inc., Petitioner v. Tempnology, Llc, Respondent* v. *Tempnology, Llc, Respondent*, 587 U.S. (May 20, 2019).
[87] Evan J. Dombrowski, „The impact of insolvency on license agreements," Managing Intellectual Property, Nov. 2015, issue 254, p.35.
[88] *Ibid*.

qualify as not (or not completely) performed agreements in terms of § 103 Insolvenzordnung."[89] Since the adoption of current InsO there have been debates on the question

> whether at least some license agreements should be exempt from the general principle according to which the (…) administrator is free to decide whether ongoing agreements of the bankrupt party will be (…) performed or not.[90]

Unfortunately attempts to legislate for specific regulation have failed so far. Arguments in public space that termination of ongoing license agreements is a serious disruption to the licensee's business and it cannot be justified by the general principle that all creditors should be treated equally, or that other major jurisdictions like U.S. and Japan provide better protection to licensees of a bankrupt debtor, have not gained response.

Government's proposal for a new section 108 a) in InsO under which „license agreements concluded by the bankrupt debtor (...) continue to be binding (...)"[91] was rejected in 2008. Returning to relevant conclusions made in third chapter on the legal nature of intellectual property license in itself, in particular that under specific circumstances license agreement can amount to an actual sale of intellectual property, law in Germany too distinguishes certain qualified constellations which result in license agreements not falling within the scope of § 103 InsO. This can be the case when „license agreements are exclusive or (otherwise) qualified as being of a quasi-proprietary (...) or if they are of a usufructuary nature."[92] Some scholars have made conclusion that due to the monist copyright approach existing in Germany[93], licenses for copyrighted intellectual property (including software licenses – my remark), do not fall within the realm of § 103 InsO and „such agreements are not affected by a bankruptcy"[94], while others have pointed out that the „grant of a license provides for ongoing obligations at least as long as the licensee is still under obligation to pay royalties"[95], and that

> Such license agreements can be subject to § 103 InsO and, accordingly, are generally subject to the opt-out right of the insolvency administrator.[96]

On November 17, 2005, the Federal Court of Justice announced its decision on the question whether subsequent condition in license agreement according to which the licensee acquired full

[89] Mark Anderson, Research Handbook on Intellectual Property Licensing (Edward Elgar Publishing, 2013), p.292.

[90] Anderson, *supra* note 89, p.287.

[91] Anderson, *supra* note 89, p.288.

[92] Anderson, *supra* note 89, p.292.

[93]According to monist copyright doctrine assignment of copyrights themselves is not permitted, instead an authorization to certain actions otherwise called exclusive license is permitted.

[94] Anderson, *supra* note 89, p.293.

[95] Kai-Uwe Plath, „German Federal Court of Justice paves the way for securing IP licenses in case of insolvencies," *C.T.L.R.* (Computer and Telecommunications Law Review), Vol. 12, Issue 6 (2006), p.177.

[96] *Ibid*.

rights to licensed software in the case of a termination of the license agreement, survives the termination of license agreement by debtor's administrator under § 103 InsO.

The license agreement in question had a clause in it corresponding to which both parties were entitled to terminate the agreement disregarding the term set out in the agreement if relevant cause occurs. The relevant cause would be such facts based on which continuation of the agreement could not be expected from the party terminating the agreement. Assessment of all circumstances in individual case and taking into account interests of all parties is required for establishing such facts. Regardless whichever of the parties choses to terminate the agreement, source code version existing at the moment of termination together with rights to use and distribute software, shall pass to the licensee. For the transfer of source code the licensee makes lump sum payment in set amount.

After when software licensor became insolvent and administrator announced his intention not to continue performance of the agreement under § 103 InsO, the licensee elected to terminate the agreement according to the aforementioned condition under which termination is possible if facts indicate that continuation of the agreement could not be expected from the party terminating the agreement. In addition, licensee asked the administrator to provide it with newest version of software source code. This request was refused by the administrator on the grounds that according to § 91 InsO

> No rights to any of the insolvent debtor's assets can be acquired by third parties after the insolvency proceedings have been initiated.[97]

The Federal Court of Justice rejected administrator's cassation complaint and in its decision[98] reiterated its previous case law according to which legal rights condition subsequent are to be considered existent in case of insolvency. The same holds true if the condition subsequent occurs after commencement of insolvency proceedings. The relevant question is whether those rights were excluded from insolvent debtor's estate before commencement of insolvency proceedings in a manner that administrator cannot unilaterally regain them. By agreeing to the terms of license agreement corresponding to which rights to use and distribute software and also to software source code were transferred to licensee on condition subsequent (termination of the license agreement by either party due to facts indicating continuation of the agreement impossible), licensor (the debtor) irrevocably excluded those rights from its assets.

[97] *Ibid*.

[98] BGH IX ZR 162/04 (unreported, November 17, 2005).

Under such circumstances the court could not agree with the administrator's position that provision of § 103 InsO (administrator can refuse to perform mutual contract that is not or not completely performed) could serve as a means to prevent the transfer of the intellectual property rights to the licensee[99], as those rights where already transferred before commencement of insolvency proceeding and as such they are not part of the debtor's estate.

Courts conclusion that condition subsequent transfer of title has retroactive effect in insolvency proceedings is especially important for the licensees who opt for source code escrow agreement. Decision upholds reasoning of lover court who established that insolvency in itself was not the condition subsequent, rather it was termination of the agreement based on any relevant cause and aim of such clause was not to harm other creditors by gaining preferential treatment, but rather it was securing licensee's rights to use software. This argument was stressed by courts reminder that contrary to its previous case law, its current position is that that the commencement of insolvency proceedings does not have material effect on previously concluded contracts. The consequences for opening insolvency proceedings are that that unimplemented claims in so far as they are not concerned with claims already implemented, cannot be enforced against debtor's estate.[100]

4. Intrinsic properties of escrow agreement

In Black's Law Dictionary escrow is defined as:

> A legal document or property delivered by a promisor to a third party to be held by the third party for a given amount of time or until the occurrence of a condition, at which time the third party is to hand over the document or property to the promisee.[101]

Generally escrow agreements are being viewed as either sui generis agreements, or as an agreement that legally secures commission agency. For example, in Germany it is considered a sui generis agreement which anticipates various options of reciprocal relationships between contractors with elements of bailment[102]. The need for an escrow arrangement in commercial dealings is based on the fact that „simultaneous exchange of all of the consideration being

[99] *Ibid*.

[100] *Supra* note 98.

[101] Black's Law Dictionary, 11th edition, (West Publishing Co, 2019).

[102] L. Yu. Vasilevskaya, „An Escrow Account Agreement: Problems of Legal Qualification," *Russ. Jurid. J. Elec. Supp.* (Russian Juridical Journal Electronic Supplement), Vol. 2016, Issue 2 (2016), p.38.

conveyed in a transaction is not always desirable or possible."[103] By choosing neutral third party and endowing it with authority to certify the occurrence of certain conditions necessary for fulfillment of the main agreement, businesses are reducing the risk of non-performance by counter party. That is to say, parties are placing the transaction beyond their reach and „except for the unfulfilled conditions, there is no turning back."[104] Thus, escrow agreement offers flexible, universal legal solution main function of which is guaranteeing execution of principal obligations arising out of the agreement between creditor and debtor. The guarantee effect stems not from different ways of securing execution of commitment, but rather by setting up special legal regime for property which is being isolated within specific account of the escrow agent[105]. Escrow agent is representing interests of both parties – the depositor and beneficiary, and is acting as an authorized representative, who carries out its duties in accordance with the agreement. Nevertheless, escrow agent shall remain neutral third party towards depositor and beneficiary thus guaranteeing interests of both parties.

The very nature and idea behind the escrow agreement – to safeguard contracting parties against non-performance, indicates the most important feature of such agreement - neither of them can exert any rights over the legal documents/funds deposited in escrow. In such circumstances, from the legal perspective escrow agreement should be recognized as a real (not consensual) contract. Real contracts are those that enter into the force after the object of transaction is delivered to entitled party, i.e. besides the consent of the parties, the delivery of some thing is required. Even though the depositor hands over „the thing" to escrow agent in order to fulfill its obligations towards beneficiary under main agreement, in fact depositor surrenders „the thing" to the control of the escrow agent simultaneously acquiring a claim against escrow agent to hand over the deposited asset to beneficiary if previously established condition have come into existence[106].

The general assumption that in order for escrow agreement to be effective there needs to be a „surrender of dominion and control over the instrument by the depositor to the escrow agent"[107], becomes not so self-bending in the case of software source code escrow where the

[103] Thomas M. Byrne, „Escrows and Bankruptcy," *Bus. Law.* (Business Lawyer (ABA)), Vol.48, Issue 2 (1992-1993), p.761.

[104] *Ibid*.

[105] Vasilevskaya, *supra* note 102, p.39.

[106] Vasilevskaya, *supra* note 102, p.42.

[107] Ollie Jr. Blan, „A Survey of Escrow - A Legal Adolescent," *Ark. L. Rev.* (Arkansas Law Review), Vol. 8, Issue 2 (Spring 1954), p.165.

principal agreement usually is a software license which predominantly retains title to intellectual property of the software to the depositor.

4.1. Software source code escrow agreement – not a classical escrow

Whereas source code is very precious if not the most precious software developer's asset, due that it contains information about the inner structure and logic of a software program, it is obvious why software developers consider it to be their „crown jewel" and guard it fiercely. Source code escrow as a legal instrument was introduced in effort to strike balance between the software owner's interests in

> Protecting the trade secret of its software, the source code, and a licensee's interest in obtaining the source code should a vendor become defunct or bankrupt.[108]

Throughout this paper it has been emphasized that source code escrow agreement is „a risk aversion and business continuity mechanism"[109], usefulness of which occurs from the nature of source code – it allows to maintain and update the software. In normal circumstances these actions are carried out by the software licensor. However, if a situation arises where the licensor cannot fulfil these obligations, the ordinary praxis that software licensee receive software (installs it on a computer) in object, not source code format, may call for provisions that secure the licensee with access to source code. This is so especially in cases where software is customized for specific business needs in business-to-business arrangement. Solution for this is source code escrow, where source code is deposited with trusted third party – software agent, who is obliged to deliver source code to the software licensee if contracted-for circumstances, like software developers' inability to provide maintenance and support services, arises. The consideration underlying source code escrow is that with source code licensee should be able to maintain software without assistance from the licensor. Maintenance involves „fixing errors, ensuring compatibility with other system upgrades and adding the functionality required in the customer's changing business."[110]

Such an arrangement can be concluded either between software developer (licensor) and escrow agent, or it can be three-party and in this case the software user (licensee) is also taking

[108] Walter D. Denson, „The Source Code Escrow: A Worthwhile or Worthless Investment," *Rutgers Bankr. L. J.* (Rutgers Bankruptcy Law Journal), Vol. 1, p.2.
[109] Kingsley, *supra* note 83, p.387.
[110] Shawn Helms and Alfred Cheng, „Source Code Escrow: Are You Just Following the Herd?", available on: https://www.cio.com/article/2437091/source-code-escrow--are-you-just-following-the-herd-.html. Accessed May 10, 2020.

active role in agreement. Two-party escrow agreements usually are created by software licensor and escrow agent and it means that the licensor has established a single general escrow account for the benefit of its licensees who are then added as beneficiaries to the account by executing a single form. Three-party escrow arrangements are best suited for a bespoke software in business-to-business relationships, where the program is expansive, tailored for special needs and due to this fact switching to another program in case software developer is failing to meet its obligations under license agreement would be a costly and entangled process.

Due to the fact that software source code escrow agreements are being represented as a legal means to ensure the licensee's business continuity in case the licensor becomes insolvent or bankrupt[111], further analysis will be concerned with materials deposited, what are source code release events, and, whether the agreement in question is insolvency-resistant.

4.1.1. The materials deposited into escrow and possible options for their verification

As noted before, even though the software source code (a text listing commands for computer hardware) is necessary for creating a program similar to the existing one or maintaining and editing it, it is not the only component needed for successful achievement of this aim. In order for escrow agreement to be effective, licensees should make sure that together with source code, licensor deposits all the relevant materials/documentation that was used by the licensor (software developer) in creating the software in question, and basically it would need to be a description of software's development environment:

> build instructions, programming documentation, configuration information, schematics, designs, and flow charts and any proprietary software tools, libraries, linkers, utilities, compilers, and other programs used by licensor's programmers to develop, maintain or implement the software, including instructions for compiling and linking the source code into executable forms or for building an executable version of the software.[112]

Besides that, parties should agree that each new or updated version of software and documentation related to that is also deposited into escrow in order to avoid situation where even though source code is released, the beneficiary (licensee) is unable to use it for software

[111] *See* information on Iron Mountain Incorporated web page, available on: https://www.ironmountain.com/information-management/software-escrow. Accessed on February 22, 2020. *See* also EscrowTech International, Inc. web page, available on: https://www.escrowtech.com/software-escrow.php#sbAgreements. Accessed on February 22, 2020. *See* also Codekeeper BV web page, available on: https://codekeeper.co/index.html. Accessed on June 8, 2020.

[112] Jon Christiansen, „Doing Software Escrows Right", p. 7-8, unpublished, available upon registration at EscrowTech International, Inc., web page, available on: https://www.escrowtech.com/.

maintenance since it does not contains any documentation developed after the initial escrow of the software.

All the above mentioned materials deposited into escrow should be designed in such a manner that a reasonably skilled professional (programmer) is able to develop, maintain or implement the software according to those materials. However, one might consider the fact that if licensee would have the technical and human resources necessary for developing software, it would not enter into license agreement in the first place. Instead, it would develop the software itself. Apparently, if licensee enters such contract, it does not have resources needed for independent creation of software. If that is the case, licensee should consider negotiating for exception of non-solicitation clause usually included in license and principally prohibiting solicitation of licensors employees – programmers who would be most intimately familiar with the software in question. Considering the fact that release events in their largest part are closely related with poor financial condition of the licensor, soliciting its (former) employees should be permissible.

Not least important from the licensees perspective is making sure that deposited material is duly verified in order to avoid situations where the released material is defective and can't serve its main purpose – maintenance of the software. For extra fee escrow agents offer variety of verification levels, starting from the most basic one – physical audit of the deposit materials, which includes a deposit material media readability analysis, a file listing, a file classification table, virus scan outputs,[113] to full usability tests when escrow officer compiles the source code and creates or builds an executable application which

> can then be run against a test plan or provided to the beneficiary for testing to ensure that it matches the software used by the beneficiary.[114]

Due to the high price of the latter (what could be validated for critically important software), another, more realistic option for licensee, is to engage with independent contractor who would observe the preparation of deposit materials before they are deposited into escrow. As it can be seen, assessment of whether deposit materials correspond to quality requirements needed to maintain or implement the software is crucial for escrow agreement to be worth the expenses. However, even the best quality deposit materials have no value to licensee if it can't access them if and when occasion (safeguarding against which are the consideration of the

[113] *See supra* note 111.

[114] Christiansen, *supra* note 112, p.14.

escrow agreement) for their release happens. Next subchapter will deal with the most important and complicated part of the software source code escrow agreement – deposited materials release events.

4.1.2. Source code release events

In previous subchapter it was concluded that it is not enough to deposit in escrow only the source code of software, and the deposited materials should be verified in best case scenario in fully fledged test environment. Nevertheless, having the deposit materials of adequate quality at the disposal of escrow agent,

> Tangible deposit materials and any copies thereof made by the escrow company in accordance with the escrow agreement should be owned by the escrow company, but such ownership does not include any copyrights or other intellectual property in or to the deposit materials[115]

is only a half of the battle. Escrow agreement serves its purpose – risk aversion and licensee's business continuity, only in case if the licensee receives the deposit material needed for maintaining and implementing the software, when the situation disrupting its business, occurs. Once again, due to the fact that source code and all the relevant to it materials can be used to create a competitive software thus having no development cost, is the reason why release events are construed the narrowest way possible.

Most commonly used release conditions are as follows:

1) Licensor discontinues business because of insolvency or bankruptcy, and no successor assumes licensor's software maintenance obligations under the license agreement;

2) Licensor (or its successor) defaults in its obligation to provide maintenance services as required by the license agreement and fails to cure such default within previously in escrow agreement agreed period of time;

3) Licensor ceases to maintain the software for beneficiary while under a maintenance obligation, and no successor to licensor continues to maintain the software for beneficiary;

4) Licensor discontinues its business relating to the software and no successor to such business assumes and carries out licensor's contractual obligations to maintain the software for beneficiary;

[115] Christiansen, *supra* note 112, p.10.

5) Licensor becomes insolvent or admits either insolvency or general inability to pay its debts as they become due;

6) Licensor files a petition for protection under the U.S. Bankruptcy Code or an involuntary petition in bankruptcy is filed against licensor;

7) Licensor is acquired by or merges with a competitor of beneficiary.[116]

From the aforementioned release conditions it is clear that unsatisfactory financial condition of the licensor and material breach of maintenance duties are the main occasions for the release of source code to beneficiary (licensee). Standard software source code escrow agreements[117] contain provision under which escrow agent will transfer the source code and relevant materials to beneficiary on two cumulative grounds: i) at least one of the agreed release events has occurred; ii) the licensor, after receiving notice from escrow agent that release event has occurred, explicitly permits escrow agent that source code be transferred to beneficiary. As it can be seen from construction, a quick release of deposited materials is possible if there is no dispute between the licensor and the licensee on the fact whether there exists release event. However, if there is a dispute, then, again, standard praxis for escrow agents is to provide arbitration clauses[118]. Even though arbitration shall be regarded effective and affordable forum for dispute resolution, it might not be available to parties where dispute exists on whether the licensor has breached its software maintenance duties as these duties are often regulated in either software license or master service agreement.

In case Exari Systems Inc. and Exari Group, Inc., v. Amazon Corporate LLC, the court was asked to settle a dispute whether release event „[Exari's] breach and failure to cure (…) any substantial, material obligation imposed on it pursuant to this License Agreement (…) had occurred."[119] Peculiarity of the case was that the provision cited was included in both, the Master License Agreement and Escrow Agreement. The former provided that parties irrevocably consent to the exclusive jurisdiction and venue of the federal and state courts with respect to any claims, suits or proceedings arising out of or in connection with the Master License Agreement. The latter, however, stated that any dispute arising among any of the parties concerning the rights or obligations of any Party will be submitted to, and settled by arbitration. When deciding which proceeding took precedence (lawsuit presented to court or arbitration), the court concluded that

[116] Christiansen, *supra* note 112, p.

[117] *See supra* note 111.

[118] *Ibid*.

[119] *Exari Sys. Inc.* v. *Amazon Corporate Llc,* Case No. C15-356 MJP (W.D. Wash. Jul. 9, 2015).

controlling question is, under which agreement the dispute regarding licensors obligation to maintain the software, arose. Taking into account that Escrow Agreement called for arbitration for „any dispute, difference or question arising among any of the parties concerning the construction, meaning, effect or implementation of this Agreement (...)"[120], the court inferred that

> „breach and failure to cure" a contractual obligation under the MLA does not constitute a „dispute, difference or question arising among any of the parties concerning the construction, meaning, effect or implementation of [the Escrow] Agreement" and thus questions addressed to any alleged breach of the Master License Agreement arise under the MLA and must be adjudicated according to the „Governing Law" provision of that contract. The Court takes notice that Escrow Agreement purports to cover [a]ny dispute, difference or question arising among of the parties concerning (…) the rights or obligations of any Party, but reading of that phrase in a manner according to which all questions concerning the parties' rights under all agreements shall be dealt by arbitration would render the Master License Agreement provisions meaningless and the Court could not endorse such an outcome.[121]

This case demonstrates the first, but not the last, challenge escrow agreement beneficiary (software licensee) encounters whilst trying to obtain the source code after presumable occurrence of release event. It is a common praxis to include arbitration clauses in standard escrow agreements and what is even more importantly, escrow agent transfers the source code and all relevant deposited materials to beneficiary only after receiving permission from the licensor. This is a significant difference from classical escrow agreement, which, as was mentioned previously, requires that depositor surrenders its dominium over the deposited objects.

Regardless of escrow being concluded under civil law or common law, several common vulnerability's under bankruptcy law can be discerned:

1) Who owns deposited materials – are they part of estates property;

2) Does the staying of all the proceedings against debtor due to the commencement of case precludes licensee's claim against escrow agent to deliver the deposited material;

3) Can the bankruptcy trustee/administrator use, sell, or lease the deposited material;

4) Whether escrow agreement is executory/mutually unperformed contract[122].

[120] *Ibid.*

[121] *Ibid.*

[122] *See* Periklis A. Pappous, „Software Escrow: The Court Favorite and Bankruptcy Law," *Santa Clara Computer & High Tech. L. J.*, Vol.1, Issue 2 (1985), pp.309-326.

4.1.3. Ownership of the deposited material and bankruptcy estate

Addressing the first identified obstacle, i.e., who can claim the deposited material (source code), both U.S. and German bankruptcy regulations contain similar provisions. Namely, section 541 in U.S. Bankruptcy code states that the commencement of a case creates an estate, comprised of all legal or equitable interests of the debtor in property as of the commencement of the case.[123] Germany's Insolvency Act (InsO) section 35 provides that „commencement of insolvency proceedings determines the estate (...)."[124] In Germany the estate comprises all assets of the debtor at the time when the insolvency proceedings are commenced, together with all the assets which are acquired or otherwise obtained thereafter until the termination of the proceedings.[125]

Due to the fact that licensor presumably enjoys protection of trade secret and copyright law for deposited material, licensor has property interests in that material[126], and, under the notion that bankruptcy estate consists of all legal or equitable interests (U.S.) and assets (Germany), it can be concluded that in both systems such property interests falls within the bankruptcy estate. Since the source code is covered by copyright and it in itself is intangible property, the conclusion can be made that „despite the fact that the tangible property [carriers which contains the deposited material] is out of the possession of the licensor"[127], it becomes part of the bankruptcy estate upon commencement of the bankruptcy case[128]. Letters of credit provide guidance for possible solution for this situation from the licensee's perspective. U.S. bankruptcy case law „generally holds that the customer's filing for bankruptcy does not effect the right of the beneficiary to seek payment from the bank."[129] Such evaluation stems from structure under which the transaction is carried out, i.e., it consists of three agreements: one between bank and its customer (depositor) under which bank takes on the obligation to pay funds to beneficiary under specified circumstances; the second – the letter of credit itself which obliges the bank to pay the funds to beneficiary; the third – principal agreement between depositor and beneficiary (it could be, for example, a construction agreement under which depositor has the duty build something). The fact that funds are handed over to the bank means „that the funds are property of the bank,

[123] 11 U.S.C. § 541 (2011).
[124] § 35 Insolvenzordnung (Insolvency Act), 5 October 1994.
[125] Felsenfeld, *supra* note 78, GER-53.
[126] *See* Pappous, *supra* note 122, p.326.
[127] Pappous, *supra* note 122, p.312.
[128] Pappous, *supra* note 122, p.311.
[129] Pappous, *supra* note 122, p.312.

not of the debtor."[130] To secure that escrow agreement fulfills the considerations underlying in it (securing unimpeded use of licensed software and providing options for its maintenance), it is advisable to construe it in a manner which separates the relationships between escrow agent-licensor and escrow agent-licensee. To put it differently, from the licensee's perspective escrow agreement would be effective if the licensor (depositor) would explicitly declare, at the moment of depositing source code, that it has departed possession of the deposited material in the sense that it has no property interest in it. Without licensor's property interests in the deposited material it cannot become property of the bankrupt estate. Aforementioned strategy requires that circumstances under which the deposited materials are released to beneficiary are not expressly conditioned on the financial situation or the bankruptcy of the licensor or the so called *ipso facto* clauses[131], which terminate contracts in case one of the party becomes insolvent. It is admitted both in the U.S.[132] and in Germany[133] that they should be invalid and the reasoning is that the bankruptcy estate should be protected from any and all actions that deteriorate value of it.

Iron Mountain, a company offering software source code escrow services to „more than 90% of the Fortune 500 companies"[134], and EscrowTech, offering the same services to „over a half of the Fortune 500 companies"[135], in their standardized escrow agreement forms as a release events include events such as „owner discontinues business because of insolvency or bankruptcy (…)", „depositor is subject to voluntary or involuntary bankruptcy." Even though two aforementioned escrow agents are based in the U.S., and assumption could be made that this is a feature characteristic only for common law legal system, standard terms in agreements from escrow agents based in EU also contain release events such as „Client [the licensor] ceases to exist as an entity."

4.1.4. Staying of all the proceedings against debtor due to the commencement of bankruptcy case

[130] Pappous, *supra* note 122, p.313.

[131] *Ipso facto* clauses automatically terminate the contract or lease, or permit the other contracting party to terminate the contract or lease, in the event of bankruptcy. *See* Albert Robin, „Ipso Facto Clauses," *J. F. Comm. Franchising* (Journal of the Forum Committee on Franchising), Vol.1, Issue 3 (Summer 1981), p.3.

[132] 11 U.S.C. § 365 (e).

[133] *See* §119 Insolvenzordnung (Insolvency Act), 5 October 1994 and *see* also BGH IX ZR 169/11 (unreported, 15 November, 2012).

[134] Available on: https://www.ironmountain.com/information-management/software-escrow. Accessed May 30, 2020.

[135] Available on: https://www.escrowtech.com/?gclid=EAIaIQobChMIzNjZh-366QIVDg8YCh3y8g7sEAAYASAAEgIl9fD_BwE. Accesses May 30, 2020.

Under U.S. Bankruptcy Act section 362, the:

> Filling of bankruptcy „operates as a stay, applicable to all entities, of (…) any action to obtain possession of property of the estate or of property from the estate."[136]

The same holds true for Germany since:

> The commencement of insolvency proceedings results in the immediate imposition of an automatic stay of all actions and other legal proceedings concerning the assets of the estate which have been commenced by the debtor.[137]

If the parties to escrow agreement have failed to separate the licensor's property interests in deposited materials, which results in their becoming a part of bankruptcy estates property, it should be obvious that automatic stay provisions invalidates licensees right to acquire source code and relevant materials, because automatic stay protects, firstly, the debtor „from creditor harassment"[138], and, secondly, it „prevents a race of diligence by creditors, thus furthering the bankruptcy goal of equality of treatment."[139]

In the U.S. bankruptcy legislation, even after the commencement of bankruptcy case and staying all the legal actions taken against the debtor, the licensee, theoretically, under section 362 (d)-(f) can argue for lifting such a stay if it can demonstrate the lack of adequate protection of an interest in property (deposited materials) or that the debtor (licensor) does not have an equity in such property and such property is not necessary to an effective reorganization[140].

However, most likely licensee would not succeed in its efforts to lift the automatic stay. Firstly, as the legislative history shows[141], the Section 362 (d) (1) of U.S. Bankruptcy Code is intended to protect secured creditors, if they can prove to court that automatic stay deprives them of adequate protection of an interest in property. Assumption could be made that the licensee as unsecured creditor would fail in convincing court to lift the automatic stay due to lack of adequate protection. Secondly, the Section 362 (d) (2), which provides that lifting of automatic stay is possible in cases where the debtor (licensor) does not have an equity in such property (deposited materials) and such property is not necessary to an effective reorganization, is impeded by the fact that deposited materials falls into the bankruptcy estate as they are a part of

[136] Pappous, *supra* note 122, p.314.

[137] Felsenfeld, *supra* note 78, GER-41.

[138] John M. Conley and Robert M. Bryan, „Software Escrow in Bankruptcy: An International Perspective," *N.C.J. Int'l L. & Com. Reg.* (North Carolina Journal of International Law and Commercial Regulation), Vol.10, Issue 3 (1985), p.592.

[139] Pappous, *supra* note 122, p.314.

[140] 11 U.S.C. § 362 (d).

[141] Pappous, *supra* note 122, p.317.

legal interests of the debtor. Although it seems reasonable to assume that the licensee who uses licensed software critical for its business operation would suffer damages if deprived of access to deposited materials (source code and relevant documentation) necessary for software maintenance, the fact that bankruptcy estate has property interests in it trumps these considerations. There have been arguments that:

> The debtor should not use the stay to deprive third parties of the benefits of acts that have been completed prior to the filing of bankruptcy[142],

but, again, such argument is based on the assumption that the property in question is property either of the escrow agent or the licensee. Returning back to previously discussed release conditions, escrow agents pride themselves that the release process is quick and without unnecessary bureaucracy, however, the promise to the licensee – quick access to source code if licensor becomes insolvent or fails to meet software maintenance obligations, obviously fails as the deposited material falls within the bankruptcy estate.

4.1.5. The right of bankruptcy trustee to use, sell, or lease the deposited material

U.S. Bankruptcy Code Section 363 provides bankruptcy trustee or debtor in possession with right to use, lease or sell property of the bankruptcy estate, while Section 365 (n) provides that licensee can retain its rights under contract and under any agreement supplementary to such contract. It must be noted that transactions taken under Section 363 does not bound the new acquirer of the property to carry on with any previous obligations incumbent on the debtor, because the principal aim of bankruptcy – to maximize estates value, would be forestalled if the property in question would be encumbered with different kind of duties. On the other hand, if the licensee elects to retain its rights endowed upon him by the contract in question, sale could render futile those rights.

Two cases concerning such situation, but with real property lease, has shown that there is no one concise answer to question whether sale made by trustee trumps non-debtors rights under contract concluded prior to commencement of bankruptcy case. In the first case[143] non-debtor lessee „claimed it could not be evicted because under § 365 (h) its possessory interest was

[142] Conley and Bryan, *supra* note 138, p.593.
[143] James E. Raymond, „Software Licenses, Source Code Escrows, and Trustee Powers under 11 U.S.C. Sec. 365," *J. Bus. Entrepreneurship & L.* (Journal of Business, Entrepreneurship and the Law), Vol.1, Issue 1 (2007), p.63.

protected even if the lease was rejected."[144] Disagreeing, debtor argued that Section 363 (f) allowed free and clear sale of property and lessee's possessory interest defers against this provision. The Seventh Circuit concluded that trustee could sell property unencumbered with lessee's possessory interest. In a later case, a district court chose not to follow such a precedent as it would allow the debtor to achieve result directly opposite to the provisions of Section 365 (h) – to dispossess lessee of leased property[145].

As in the previous subchapter, if depositor (the licensor) has retained property interest in the deposited materials, then, after commencement of bankruptcy proceeding and pulling all equitable interests of debtor under the bankruptcy estate encumbers the possibility for licensee to obtain and use deposited materials, as they can be sold to third party unburdened with licensees use rights.

4.1.6. Escrow agreement as executory/mutually unperformed contract

U.S. Bankruptcy Code Section 365 (a) grants trustee, subject to the court's approval, rights to assume or reject any executory contract or unexpired lease of the debtor. Similar provision is contained in Germany's InsO Section 103 – if a mutual contract was not or not completely performed by the debtor and its other party at the date when the insolvency proceedings were opened, the insolvency administrator may perform such contract replacing the debtor and claim the other party's consideration.

As is the case with intellectual property licenses discussed in third chapter and its subchapters, software source code escrow agreements – an ancillary agreement to it, in most cases fall into the category of executory contracts since it is relatively easy to establish that the parties have unfulfilled obligations, for example, the licensor has a duty to deposit software updates, and the licensee is obliged to comply with the terms of the escrow agreement.

Up until implementation of Intellectual Property Bankruptcy Protection Act of 1987 which amended U.S. Bankruptcy code with section 365 (n), which provides that if the trustee rejects an executory contract under which the debtor is a licensor of a right to intellectual property, the licensee under such contract may elect to treat such contract as terminated or to retain its rights to such contract, to such intellectual property (including any embodiment of such

[144] *Ibid*.
[145] *See* Raymond, *supra* note 143, p.64.

intellectual property to the extent protected by applicable nonbankruptcy law), the beneficiary of software source code escrow agreement (the licensee), most probably would not obtain deposited source code due to the fact that escrow agreement would be treated as executory, thus, to be rejected under bankruptcy laws.

Under current legal provisions, mainly section 365 (n) (3) (A), the software source code beneficiary finds himself in much more satisfactory situation, since the aforementioned section provides - if the licensee elects to retain its rights, as described in paragraph (1)(B) of this subsection, then on the written request of the licensee the trustee shall – (A) to the extent provided in such contract, or any agreement supplementary to such contract, provide to the licensee any intellectual property (including such embodiment) held by the trustee; and (B) not interfere with the rights of the licensee as provided in such contract, or any agreement supplementary to such contract, to such intellectual property (including such embodiment) including any right to obtain such intellectual property (or such embodiment) from another entity. In other words, U.S. Bankruptcy Code provides the licensee with rights not only to continue the use of licensed intellectual property (in this case software) under license agreement, but also ensures that supplementary agreement (escrow) to it should be bankruptcy-resistant in the sense that commencement of bankruptcy proceedings does not abnegate the access to deposited materials. At least, this was the legislator's intention when amendments to Bankruptcy Code were made:

> Section 365(n)(1)(B), thus, speaks of the retention by the licensee of rights to the intellectual property under „any agreement supplementary to such contract." The licensee retains both the rights set forth in the rejected license itself and any agreement supplementary thereto, whether the supplementary agreement was itself the subject of rejection by the trustee.[146]

The provision on licensees' rights retention under supplementary agreement is reflected in standard escrow agreements that contain conditions for permitted use of deposited materials, stating that this „permitted use license" is limited with rights conveyed to the licensee according to principal license agreement. In other words, escrow agreements comprises licenses in intellectual property recorded onto the deposited materials. The entry into the force for such licenses is somewhat dual – on the one hand, it will become effective only when and if a release event occurs, on the other hand, it claims that the date of grant for such a license is the date of the escrow agreement.

[146] Raymond, *supra* note 143, p.53.

Although the Senate Report on section 365(n) reveals the legislators intention to provide intellectual property licensees with legal remedies in case licensor becomes insolvent, especially attending to supplementary agreements designed as licensees' business risk aversion strategy[147], the question remains if licensees can enforce rights to access the escrowed source code if such rights are „conditioned on the act of filing a bankruptcy petition itself."[148] U.S. Bankruptcy Code section 365(e) in relevant part states that:

> Notwithstanding a provision in an executory contract (...) executory contract (...) of the debtor may not be terminated or modified, and any right or obligation under such contract (...) may not be terminated or modified, at any time after the commencement of the case solely because of a provision in such contract (...) that is conditioned on – (A) the insolvency or financial condition of the debtor at any time before the closing of the case; (B) the commencement of a case under this title; or (C) the appointment of or taking possession by a trustee in a case under this title or a custodian before such commencement.[149]

To summarize it, even though Section 365(n) protects licensee's rights to the intellectual property upon certain conditions, it is silent on how section these rights interact with section 362 (automatic stay of all proceedings) and 363 (free and clear sale).

Taking into the account the trustees' rights under U.S. Bankruptcy Code section 365(a) not only to reject, but also to assume debtors executory contracts, section 365(n)(4)(B) provide that unless and until the trustee rejects such contract, on the written request of the licensee the trustee shall – (A)to the extent provided in such contract or any agreement supplementary to such contract – (i) perform such contract; or (ii) provide to the licensee such intellectual property (including any embodiment of such intellectual property to the extent protected by applicable nonbankruptcy law) held by the trustee; and (B) not interfere with the rights of the licensee as provided in such contract, or any agreement supplementary to such contract, to such intellectual property (including such embodiment), including any right to obtain such intellectual property (or such embodiment) from another entity.[150] From the licensees perspective there would be a need to secure its business continuity proactively, i.e. if there are any signals that the licensor won't be able further to provide the services needed for software use, the licensee could try to acquire deposited source code even before the trustee rejects escrow agreement and the aforementioned rule of law, theoretically, could be the legal route to accomplish such result. However, the fact that trustee has not rejected escrow agreement does not mean that it has seized to provide

[147] S.Rep. No. 100-505, at 9 (1988).
[148] Raymond, *supra* note 143, p.54.
[149] 11 U.S.C. §365 (e).
[150] 11 U.S.C. § 365(n)(4)(B).

material obligations the licensor had under the license agreement. That is to say, for the licensee to be able to exercise retention rights under U.S. Bankruptcy Code section 365(n)(4)(B), although not specifically mentioned in the law, establishing that the trustee is not performing the duties licensor had under the license agreement, would be essential. Indication to the trustees' nonperformance as precondition for a licensee to obtain escrowed source code under section 365(n)(4)(B) is found in the Senate Report 100-505, which states that:

> The trustee shall turn over to the licensee intellectual property held by him, upon written request by the licensee, prior to rejection by the debtor licensor but upon nonperformance by the trustee.[151]

One of a few court cases dealing with dispute whether the licensee can utilize rights conferred on it by section 365(n)(4) provides a valuable conclusions not only as to the conditions under which those rights can be executed (specifically, when can a licensee compel the trustee to perform turning over the source code), but also one more time emphasizes conditional character of the escrow agreement compared to principal (software license) agreement.

After licensor became insolvent and before the term of the Service Agreement came due, the licensee, in order to prepare to replace the licensor as a service provider, alleged that the licensor is obliged to pass the source code to it in accordance with section 365(n)(4)(B) of the U.S. Bankruptcy Code. Such claim was based on the provision contained in disguised escrow agreement termed Indemnity Agreement, which provided that:

> Licensee shall have the right at any time during the term of this Agreement to retrieve the Games Escrow Materials from the Agent within one (1) business day of Licensee's notification to agent of the intention to retrieve said Games Escrow Materials (…).[152]

Denying such motion, the Court emphasized that „section 365(n)(4) affords no greater right to any party than that (…) „provided in such contract, or any agreement supplementary to such contract.""[153] Reasoning that parties had not contested the executory nature of the Definitive Agreement (the license agreement), the court rejected the possibility to evaluate whether conditions for source code release have occurred according to the Indemnity (escrow) Agreement, which provisions were beneficial to the licensee. Instead, the court adhered to the strict source code release provisions contained in the license agreement (the Definitive

[151] S.Rep. No. 100-505, at 9 (1988).

[152] *In re Bluberi Gaming Techs., Inc*., (4 August 2016, unreported), available on: file:///C:/Users/madarastorha.AT/Downloads/In%20re%20Bluberi%20Gaming%20Techs.,%20Inc.pdf. Accessed June 6, 2020.

[153] *Ibid*.

Agreement). The reason for such an outcome was the fact that provisions for source code release in the license and escrow agreements were not integrated, in fact they were independently enforceable – the former provided access to source code in case the licensor failed to perform software repair obligations, while the latter provided that the licensee could claim the release of source code at any time under any circumstances. When concluding that no release events under the license agreement had occurred (the licensee did not argued the source code to be released under it, instead, reference to escrow agreement was made), the court reasoned that:

> A tortured reading of section 365(n)(4) might lead a party to argue that it need not reconcile the two. As the Indemnity Agreement, in such an instance, would be an agreement supplementary to the Definitive Agreement, arguably it's provisions could be enforced without reference to the Definitive Agreement. See 11 U.S.C. §365(n)(4). That clearly could not have been Congress's intent (…).[154]

Outcome in this specific case shows that, although legislation provides legal remedies for a software licensee in case its licensor becomes bankrupt – licensee can pursue enforcement of such agreement which is supplementary to main intellectual property agreement, it may prove to be ineffective if it is inconsistent with the principal agreement. That is to say, including in source code escrow agreement provisions that do not take into account principal agreement terms regarding deposited material release events, renders it futile.

5. Conclusion

Aim of the thesis was to examine whether software source code escrow agreement fulfill its promise – securing beneficiaries (the licensees') business continuity, in case software developer (the licensor) becomes insolvent. After examining technical and legal nature of software, its distribution model – licensing, and escrow agreement itself, author proposes following conclusions regarding enforceability of software source code escrow agreement under bankruptcy laws.

The necessity (and often complex legal issues thereafter) for software source code escrow agreement is a result of specific nature of software (written and therefore copyrightable work insofar as regards its embodiment in programming language, on the one hand, and, purely utilitarian object, needed to ensure the functionality of computer, on the other). This duality led to applying for its protection not one, but at least two regimes of intellectual property – copyright

[154] *Ibid*.

for its expression, and trade secret for its essence, the source code. This, in turn, led to situation where product intended for perpetual use in different businesses by innumerable amount of persons and organizations, although delivered to software users, is not in their possession. This is also in large part due to the fact that software is intangible, and, when this condition is combined with long-term co-operation agreements such as software maintenance and support services, software users are becoming increasingly dependent on the licensor. Thus, contracting for escrow arrangements seems prudent business decision from the software licensee's perspective. However, in the light of aforementioned interaction between different software edges, and general rules of bankruptcy laws, the execution of such contract is practically improbable, and therefore, in authors view, impractical.

Releasing materials deposited with escrow agent after licensor's bankruptcy case has been commenced is both the promise of escrow agent and at the same time illicit action according to U.S. Bankruptcy Code Section 362 and section 240 of Code of Civil Procedure of the Federal Republic of Germany. In other words – provision in software source code escrow agreement that relevant materials (intellectual property) will be delivered to the licensee, when the licensor becomes insolvent, runs contrary to universal insolvency principle according to which no legal actions should be carried against debtor in order to keep its estates value as high as possible. Even if these legal prescriptions, for the sake of argument, would be set aside, according to standard formulations in software source code escrow agreements, the licensee will be subject to litigation on whether the agreed-for release event has occurred, since, unlike in escrow agreement in its classical sense, escrow agent will ask for a permission from the licensor whether deposited materials could be released to the licensee. It needs to be noted, that such events as „commencement of bankruptcy proceedings" are relatively easy to establish, but if the licensee claims deposited materials on the grounds that software developer (licensor) is not compliant with its maintenance and technical support service obligations, this would be a time-consuming process, because the dispute would concern establishing facts. The possible remedies for this, such as arbitration clause, are not effective – court case examined in this thesis showed that the process alone for determining which forum is appropriate for such a dispute, is time consuming, not to mention the resolution of dispute by substance.

Aforementioned provision of automatic stay on all the proceedings where debtor licensor is a litigant is not the only obstacle for releasing software source code to the licensee if the licensor becomes bankrupt. Prohibition to terminate or modify executory/mutually unperformed contract, also plays integral part. In other words, when software source code escrow participants agree that the insolvency of other party serves as an automatic termination, they tend to overlook mandatory provisions in bankruptcy laws which prohibit such clauses. The substantiation for rendering such clauses void is the need at least not to deplete the estate of the bankrupt. Conclusion can be made that provision, according to which licensors insolvency serves as a ground to turn over deposited source code and relevant materials to the licensee, is squashed by two factors: debtors insolvency stays all and every legal actions taken against it, and mandatory bankruptcy laws render *ipso facto* clauses void. It could be argued, whether prohibition on *ipso facto* clauses to serve as an automatic termination (and in context with software source code escrow consequently as a release event) of executory contract is really effective, hence judgment of 17 November, 2005, by German Federal Court of Justice in case BGH IX ZR 162/04 demonstrated that by applying veiled language (i.e., not naming licensors insolvency as a contracts' termination event, but, instead providing that contract can be terminated if circumstances occur, under which it would be unreasonable to expect further performance), parties can achieve the same result. However, relying on favorable assessment of the case facts in court seems too great risk in the sense that the outcome is hard to predict.

Even though analysis done in thesis demonstrates that the Federal Republic of Germany does not have a specific legal provisions intended for protection of licensee's contractual rights in case the licensor becomes insolvent, the outcome for software licensee, due to automatic stay and treatment of *ipso facto* clauses as void, seems rather similar both in common and civil law countries.

Considering the fact that software source code escrow agreement, according to previously made conclusions, cannot be enforced, when software licensor becomes insolvent, author as an alternative, proposes two options. First, when procuring for bespoke, business viability critically affecting software, licensee should insist that the licensor makes full assignment of intellectual property in question. Second, the software in question is built by default adhering to manageable portability requirements.

Bibliography

Primary sources

Court cases

1. *In re Bluberi Gaming Techs., Inc.*, (4 August 2016, unreported), available on: file:///C:/Users/madarastorha.AT/Downloads/In%20re%20Bluberi%20Gaming%20Techs.,%20Inc.pdf.

2. BGH IX ZR 162/04 (unreported, November 17, 2005).

3. *Exari Sys. Inc.* v. *Amazon Corporate Llc,* Case No. C15-356 MJP (W.D. Wash. Jul. 9, 2015).

4. *Mission Product Holdings, Inc., Petitioner v. Tempnology, Llc, Respondent* v. *Tempnology, Llc, Respondent*, 587 U.S. (May 20, 2019).

Legislation

1. Insolvenzordnung (Insolvency Act of the Federal Republic of Germany), 5 October 1994.

2. 11 U.S.C. (1978).

3. The European Patent Convention, adopted in Munich on October 5, 1973, available on: http://documents.epo.org/projects/babylon/eponet.nsf/0/17756BCD9D57462FC125853B0053E438/$File/EPC_16th_edition_2016_en.pdf.

4. World Intellectual Property Organization Copyright Treaty, adopted in Geneva on December 20, 1996, available on: https://wipolex.wipo.int/en/text/295157.

Secondary sources

1. Anderson, Mark. Research Handbook on Intellectual Property Licensing. Edward Elgar Publishing, 2013.
2. Bisk S. Jennifer. „Software Licenses through the Bankruptcy Looking Glass: Drafting Individually Negotiated Software Licenses That Protect the Client's Interests in Bankruptcy." Fordham Intellectual Property, Media & Entertainment Law Journal, Vol.17, Issue 3 (Spring 2007), pp.611-650.
3. Black's Law Dictionary, 11th edition, West Publishing Co, 2019.
4. Blan Jr. Ollie. „A Survey of Escrow - A Legal Adolescent." Arkansas Law Review, Vol. 8, Issue 2 (Spring 1954), pp.164-171.
5. Breyer Stephen. „The Uneasy Case for Copyright: A Study of Copyright in Books, Photocopies, and Computer Programs." Harvard Law Review, Vol. 84, Issue 2 (December 1970), pp.281-351.
6. Byrne M. Thomas. „Escrows and Bankruptcy." Business Lawyer (ABA), Vol.48, Issue 2 (1992-1993), pp.761-778.
7. Classen H.Ward. „Fundamentals of Software Licensing" IDEA: The Journal of Law and Technology, Vol.37, Issue 1 (1996), pp.1-86.
8. Conley M. John and Bryan M. Robert. „Software Escrow in Bankruptcy: An International Perspective." North Carolina Journal of International Law and Commercial Regulation, Vol.10, Issue 3 (1985), pp.579-608.
9. Denson D. Walter. „The Source Code Escrow: A Worthwhile or Worthless Investment." Rutgers Bankruptcy Law Journal, Vol. 1, pp.1-16.
10. Dietderich Andrew, Bialson Thomas and Phellan Robin. „Treatment of Intellectual Property Licenses in Cross-Border Insolvency Proceedings.", available on: https://www.uncitral.org/pdf/english/congress/Papers_for_Congress/118-DIETDERICH_and_GAA_and_PHELAN-Intellectual_property_licences_in_cross-border_insolvency_proceedings.pdf. Accessed March 25, 2020.
11. Dombrowski J. Evan. „The impact of insolvency on license agreements." Managing Intellectual Property, Nov. 2015, issue 254, pp.1-84.
12. Elvy Stacy-Ann. „Hybrid Transactions and the INTERNET of Things: Goods, Services, or Software." Washington and Lee Law Review, Vol.74, Issue 1 (Winter 2017), pp.77-172.

13. Felsenfeld Carl. International Insolvency. Part II. The Insolvency Laws of Major Countries. New York: Fordham Law School, 2000.
14. Gibson James. „Once and Future Copyright." Notre Dame Law Review,Vol.81, Issue 1 (November 2005), pp. 167-244.
15. Guarda Paolo. „Looking for a feasible form of software protection: copyright or patent, is that the question?" European Intellectual Property Review, Vol. 35, Issue 8 (2013), pp.445-454.
16. Gres L. Viktoria. „Rejection of Computer Software Licensing Agreements in Bankruptcy." Cardozo Law Review, Vol. 8, issue 2 (December 1986), pp.361-396.
17. Heath Christopher and Sander Anselm Kamperman. „*New Frontiers of IP Law. IP and Cultural Heritage, Geographical Indications, Enforcement and Overprotection*." (United Kingdom: Hart Publishing, 2005).
18. Helms Shawn and Cheng Alfred. „Source Code Escrow: Are You Just Following the Herd?" available on: https://www.cio.com/article/2437091/source-code-escrow--are-you-just-following-the-herd-.html. Accessed May 10, 2020.
19. Hemnes M.S. Thomas. „Restraints on Alienation, Equitable Servitudes, and the Feudal Nature of Computer Software Licensing." *Denv.* Denver University Law Review, Vol. 71, Issue 3 (1994), pp.577-606.
20. Katyal Sonia K. „The Paradox of Source Code Secrecy." Cornell Law Review, Vol. 104, Issue 5 (July 2019), pp. 1183-1280.
21. Kim S. Nancy. „The Software Licensing Dilemma." Brigham Young University Law Review, Vol.2008, Issue 4 (2008), pp.1103-1164.
22. Kingsley Osei. „Pouring New Wine into Old Wineskins: Why on Premise Software Source Code Escrow Arrangements Are Ill-Suited for Remotely Hosted off Premises Software as a Service License Agreements." Journal of College and University Law, Vol. 39, Issue 2 (2013), pp.383-404.
23. Larsson, Stefan. *Conceptions in the Code. How Metaphors Explain Legal Challenges in Digital Times*. New York: Oxford University Press, 2017.
24. Lucas Andrei. „Copyright in the European Community: The Green Paper and the Proposal for a Directive Concerning Legal Protection of Computer Programs." Columbia Journal of Transnational Law, Vol.29, Issue 1 (1991), pp.145-168.

25. McKenzie A.Brandon. „License to Sell: Kirtsaeng v. John Wiley&Sons, UsedSoft GmbH v. Oracle, and the impending disruption of the international software licensing landscape." available on: file:///C:/Users/madarastorha.AT/Downloads/SSRN-id3356775%20(1).pdf. Accessed April 2, 2020.
26. Meadows E. James. „Lubrizol: What Will It Mean for the Software Industry." Santa Clara Computer and High-Technology Law Journal, Vol. 3, Issue 2 (1987), pp.311-328.
27. Meisler E. Ron, Ziff D. Elaine, Gardner C. Tracy, Tullson T. Carl, „Rejection of Intellectual Property License Agreements Under Section 365(n) of the Bankruptcy Code: Still Hazy After All These Years." available on: https://docplayer.net/9961810-Rejection-of-intellectual-property-license-agreements-under-section-365-n-of-the-bankruptcy-code-still-hazy.html. Accessed May 29, 2020.
28. Murray Andrew. *Information Technology Law: The Law and Society,* 4th edition. New York: Oxford University Press, 2019.
29. Pappous A. Periklis. „Software Escrow: The Court Favorite and Bankruptcy Law." *Santa Clara Computer and High-Technology Law Journal,* Vol.1, Issue 2 (1985), pp.309-326.
30. Paterson Maximilian. „Properly Protecting Code: Solving Copyright and Patent Rights Overlap via Computer Software Suitability in Copyright." Intellectual Property Journal, Vol. 25(2), 2013, pp. 173-193.
31. Pila Justine. „Software Patents, Separation of Powers, and Failed Syllogisms: A Cornucopia from the Enlarged Board of Appeal of the European Office." Cambridge Law Journal, Vol.70, Issue 1 (March 2011), pp.203-228.
32. Plath Kai-Uwe. „German Federal Court of Justice paves the way for securing IP licenses in case of insolvencies." Computer and Telecommunications Law Review, Vol. 12, Issue 6 (2006), pp.177-178.
33. Raymond E. James. „Software Licenses, Source Code Escrows, and Trustee Powers under 11 U.S.C. Sec. 365." Journal of Business, Entrepreneurship and the Law, Vol.1, Issue 1 (2007), pp.43-66.
34. Rice, A. David. „Whither (No Longer Whether) Software Copyright." Rutgers Computer & Technology Law Journal, Vol.16, Issue 2 (1990), pp.341-356.
35. Risch Michael. „Hidden in plain Sight." Berkeley Technology Law Journal, Vol.31, issue 3 (2016), pp.1635-1668.

36. Robin Albert, „Ipso Facto Clauses," Journal of the Forum Committee on Franchising, Vol.1, Issue 3 (Summer 1981), pp.3-4.
37. Rognstad Ole-Andreas. „Legally Flawed but Politically Sound: Digital Exhaustion of Copyright in Europe after UsedSoft." Oslo Law Review, Vol.1 (2014), pp.1-19.
38. Rosser J. Roy. „European Software Patents - When a Rejection is Not a Rejection." IPL Newsletter, Vol.24, Issue 1 (Fall 2005), pp.25-28.
39. Rustad L. Michael and Kavusturan Elif. „A Commercial Law for Software Contracting." Washington and Lee Law Review, Vol. 76, Issue 2 (Spring 2019), pp.775-894.
40. Samuelson Pamela. „Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement." Berkeley Technology Law Journal, Vol.31, Issue 2 (2016), pp.1215-1300.
41. Senate Report No. 100-505, at 9 (1988).
42. United Nations Commission on International Trade Law. Legislative Guide on Insolvency Law, available on: https://uncitral.un.org/sites/uncitral.un.org/files/media-documents/uncitral/en/05-80722_ebook.pdf. Accessed April 2, 2020.
43. Vasilevskaya L. Yu. „An Escrow Account Agreement: Problems of Legal Qualification" Russian Juridical Journal Electronic Supplement, Vol. 2016, Issue 2 (2016), pp.37-49.
44. Zimmerman L. Diane. „Trade Secrets and the 'Philosophy' of Copyright: A Crash of Cultures." (July 2009), available on: file:///C:/Users/madarastorha.AT/Downloads/SSRN-id1438706.pdf. Accessed February 15, 2020.