Published on February 25, 2008 by CIO.com

SOURCE CODE ESCROW: ARE YOU JUST FOLLOWING THE HERD?

By: Shawn Helms and Alfred Cheng

Growing up, mothers train their children to resist mindlessly following their peers by asking the glib question: "if your friend jumped off a bridge, would you?" This simple question is packed with insight. Independent thought that questions others' habits, beliefs and actions is a hallmark of maturity. However, today's business culture sometimes fails this childhood test.

Take source code escrow. Without truly considering it, many organizations have a standing policy to require software developers to escrow source code of products the organization is licensing. If organizations would carefully analyze the risk / return investment, the business case for source code escrow arrangements would almost always come up negative. Dealmakers and lawyers spend countless hours negotiating escrow terms and conditions and pay escrow agents like Iron Mountain thousand of dollars to maintain the escrow account. This time and money is often a wasted investment, as the potential benefits are marginal. Customers should be skeptical of expending valuable time and money on an arrangement that is largely ineffective at accomplishing the very purpose for which it was created. Before explaining why, let's first discuss what software source code escrow is and why it has become a common part of many software transactions.

What is "Source Code" and Why is it Escrowed

Every day, companies around the world license and implement custom software applications that are critical to the operation of their businesses. Development and implementation can cost millions of dollars. Because these applications are critical, software development and maintenance contracts often require the software developers to store the "source code" of the software and explanatory documentation in an escrow account. Source code is the sequence of logical statements and operations written in a human-readable computer programming language that controls the processing of data and the functionality of software. The source code itself can be hundreds of thousands of lines of code and is normally designed and written by software programmers in programming languages such as C++, Java or Visual Basic. When completed, the source code is compiled into "executable code" that can be downloaded, installed and run on a computer. However, with only the executable code, customers have no ability to see how the software is processing data or performing functions and, for the most part, have no ability to change the operation of the software.

Because repairing problems or changing functionality is only possible with the source code, the escrow of source code is common in large software transactions involving custom developed or operationally critical applications. In a source code escrow arrangement, the source code and documentation are held in escrow by a trusted third party, the escrow agent. The source code and related documentation are to be released upon the occurrence of a "release event" such as the software developer filing bankruptcy or failing certain obligations under the license.

Following a release event, the promise of a source code escrow is that the customer can obtain the code to maintain the software without the original developer. This maintenance involves fixing bugs, ensuring compatibility with other system upgrades and adding the functionality required in the customer's changing business.

Software maintenance is essential to enterprise applications. Because the customer has no assurance that the software developer will always be around to perform software maintenance, and since such maintenance cannot be performed without the source code, escrow is considered a necessary part of certain software deals.

Why Escrow is an Ineffective and Costly Mechanism

For the reasons described above, escrowing source code of critical business software seems to be a prudent business decision for customers. However, for various reasons, the time, legal fees and other resources spent establishing and maintaining escrow accounts provide little protection for the customer.

1. Only a Small Percentage of Escrows are Ever Released

First, only a small percentage of escrowed software is ever released. Iron Mountain, the dominate escrow agent in the United States, has thousands of escrow accounts and more than 45,000 customers worldwide that have stored their software and intellectual property with Iron Mountain, including over 75 percent of the Fortune 500 (http://www.ironmountain.com/ipm/escrow/). Over the ten year period from 1990 to 1999, Iron Mountain released 96 escrow accounts, less than 10 per year (http://www.ironmountain.com/knowledge/ipm/deposit.asp). While it is unclear how many are typically released in a given year, these low release rates are indicative of the fact that rarely does a release event occur and, when it does, customers often find it easier to find an alternative software provider.

2. Most Escrowed Source Code is Defective

While some customers may view a source code escrow as an insurance policy to protect themselves in the slight chance that a release event occurs, often source code escrows fail to provide adequate protection because, upon release, the source code is frequently outdated, defective or otherwise fails to meet the customer's needs. According to Iron Mountain, 97.4% of all analyzed escrow material deposits have been found incomplete and 74% have required additional input from developers in order to be compiled (www.ironmountain.com/resources/escrow/IMD_DS_TechVerification.pdf).

Take, for example, what happened to Radisson Hotels Worldwide. Several years ago, Radisson hired a third party software company to maintain its mission critical reservation system. The software vendor agreed to escrow the Radisson software code. The code was eventually released as a result of the software vendor going out of business. But upon release, Radisson found that the source code in escrow could not even perform the fundamental and critical task of booking guests at Radisson's hotels. The released code was missing many components and the escrow account did not contain any documentation developed after the initial escrow of the software (http://www.bankruptcy.rutgers.edu/source_code_escrow.pdf).

This unfortunate discovery is not atypical in source code escrow arrangements, especially when the customer has not been vigilant in continually monitoring and auditing what is being stored with the escrow company.

3. Customers Lack Expertise to Use the Released Source Code

Even if the customer has been diligent and the released source code is properly updated, well-documented, and fully operational, most customers lack the resources or capability to utilize the code upon release. In most cases, source code has been escrowed because customers are licensing software from a vendor that is providing technology and expertise that the customer does not possess internally. Thus, once the software is released from escrow, the customer often is in no position to properly implement the software, train its employees on maintaining and supporting the software, or purchase the necessary hardware and third party software. In addition, this process of bringing such software in-house can be lengthy and demanding on a company’s resources. Further, the customer will likely be faced with a shortage of available talent knowledgeable on the released software, especially because many software license agreements prohibit customers from soliciting the vendor’s employees upon termination of the licensing arrangement. All of these factors combined make it extremely difficult for a customer to utilize source code upon release, even if the code is in pristine condition. The only alternative for the customer is to hire a third party software company. This is an expensive alternative and is often no better than moving to an alternative product. Furthermore, with this decision, the customer is back to where it started – relying on a third party to properly maintain the code.

4. Significant Delays and Legal Battles Often Accompany a Release

Another problem is that software vendors have the ability to prevent the timely release of the escrowed code and customers have limited recourse to prevent such a delay. It is not unusual for an escrow agreement to require the vendor’s approval before the source code is released; thus, even if the customer demands the escrow agent to release the software upon a release event, the escrow agent is prohibited from doing so until it receives the approval of the vendor. Delays in the release of the software are problematic because vendors may not keep the software properly updated during the period of time that the parties are disputing the release of the software.

Adding to the delay, escrow agreements often require parties to participate in alternative dispute resolution proceedings, such as arbitration or mediation, in the event of a dispute regarding the release of the source code. A commonly disputed issue is whether a release event has actually occurred. Although parties strive to clearly and completely define the triggers for a release of the software in software license agreements and escrow agreements, the language in these agreements may be ambiguous as to whether certain circumstances qualify as a release event. Therefore, the vendor can almost always dispute that such an event has occurred. A prime example of this can be found in the recent court case between Vemics, Inc. and Radvision, Ltd. (Vemics, Inc. v. Radvision, Ltd., No. 07-CV-0035, 2007 WL 1459290 (S.D.N.Y. May 16, 2007)). In this case, Vemics purchased a software license from First Virtual Communications (FVC), and the parties entered into a license agreement requiring the software to be placed in escrow and released upon specified release conditions. The parties also entered into a separate escrow agreement with the escrow agent, Iron Mountain. FVC filed for bankruptcy on March

11, 2005, and Radvision became the successor in interest to FVC's rights and obligations under the license agreement and escrow agreement with Vemics. Vemics demanded Iron Mountain to release the source code in January 2006, claiming that FVC's bankruptcy constituted a release event. Interestingly, even though parties typically enter into source code arrangements to, at a minimum, protect the customer's interest in the software in the specific event of the vendor's bankruptcy, Radvision argued that FVC's bankruptcy was not a valid basis for release of the software. Over two years have passed since Vemics demanded the release, with the parties still at odds over whether the source code will ever be released. Even if it is finally released, the long delay and expensive legal battle have been a costly investment for Vemics.

5. Utilizing an Escrow can be Expensive

Another obvious cost consideration is that the expenses related to the opening and maintenance of an escrow account are typically borne by the customer. These fees can amount to thousands of dollars. In addition to the fees paid to the escrow agent, the customer will often incur significant legal expenses related to the drafting and negotiation of escrow agreements. Software developers are resistant to provide their source code to anyone in fear of inadvertent or unnecessary release. Therefore, escrow arrangements are often intensely negotiated. However, these legal expenses pale in comparison to those that the customer will be forced to spend if the vendor disputes the customer's claim that a release event has occurred. The costs of resolving such a dispute can range from thousands of dollars in alternative dispute resolution proceedings to hundreds of thousands (or even millions) of dollars in protracted litigation. A point of great frustration for customers is that these added costs will not change the fact that the customer has no assurance that the source code is useable and has little control over the time period of release.

For these reasons, customers should consider whether or not it makes sense to ever enter into source code escrow arrangements. Despite their appearance of importance, source code escrow arrangements are almost completely ineffective at protecting customers from failing software companies and carry with them significant costs and risks.

About the Authors: Shawn C. Helms (shelms@jonesday.com) and Alfred Cheng (acheng@jonesday.com) are both attorneys at Jones Day in Dallas, Texas. Their practice is focused on complex technology and intellectual property centric transactions, including business process and information technology outsourcing and technology licensing. The views set forth in this article are the personal views of the authors and do not necessarily reflect those of Jones Day.