White Paper

Escrow Verification

ESCROW4ALL

Contents

Introduction

Why verify an escrow deposit – why not sign a contract, have developer provide material and be done with it? The reason is simple – to minor or major extent no (initial) escrow delivery is complete.

Issues range from recoverable issues like:

- Overview of third party software requirements is not complete/not present
- Documentation is not complete.

To none recoverable issues like:

- Media provided are not readable
- Object code provided instead of source code
- Parts of the source code are missing.

Do none recoverable issues happen often? Unfortunately, yes – if we analyze the results of our most advanced verification most (initial) deliveries have major issues.

There are a lot of reasons why most escrow deliveries are not complete. Many of these are often understandable:

- The person responsible for the deposit has limited time to fulfill extra work and often works with deadlines. The escrow deposit is not considered important enough and is delegated to a junior developer who does not have the necessary expertise

- The application (and its development environment) is the result of 10 years work. In most cases nobody has maintained proper documentation during this period.

In the past decade we have seen that a lot of developers find the concept of escrow verification nonsense and a waste of their time. When the deposit of source code is not properly addressed they are often quite right.

In order to continue maintenance of a software product more is required. A simple backup is not enough. To properly address this issue the following applies:

- Is a complete copy of source code provided?
- Can the development environment be rebuild?
- Can the source code be compiled, the results deployed and tested?
- Can the knowledge base present at developer be secured?

Let us address these issues in more detail in the next chapter.

Escrow Verification Issues

Is a complete copy of source code provided

Providing a complete copy sounds simple - most developers have switched to some form of Version Control System and by providing a copy using this system one should have a complete copy. There are unfortunately often issues with this approach – among others:

- Most software products are the results of many years of work – older parts are often not in the main source code tree and consequently not part of the escrow delivery
- A properly maintained version control system depends for the larger part on the people using it - unless there is stringent control there are bound to be issues (never mind how difficult it is to agree on the initial procedures)
- Parts of the software are often developed by other departments and source code of those parts are either not part of the main source code tree or the parts are provided in a compiled format.

Obviously – working without a version control system often makes above issues even more relevant.

Significance: escrow deposits are initially hardly ever complete.

Can the development environment be rebuild

In some cases setting up a development environment is simple – install the development software; set a few parameters; deploy the source code in a directory and you are done.

In most cases it is not this simple as the environment is often the result of many years of development (and nobody keeps track of what exactly was used). Furthermore there is often no time or desire to recreate the development environment with each release. Issues often encountered:

- Next to the compiler – what kind of third party components were used
    - o Do we have an overview available?
    - o Which versions are we using?
    - o Can they still be obtained (depending on how old they are)?
- How to properly set the parameters
    - o Does anybody know what is required?
    - o Did we ever document this (most often not which is logical as there is a backup in case the main system breaks down)?

Significance: it will take a new developer much time and resources to research the particularities of a development environment.

Can the source code be compiled, the results deployed and tested

Compilation

When the development environment is properly set up and a full copy of the source is provided, compilation is almost always straightforward and in a lot of instances takes less than 30 minutes (and sometimes far less). Compilation as such is often not an issue – dealing with issues because the development environment was not properly created or missing source code is something else. A compilation often needs to be run multiple times to deal with such issues.

Please note that such a compilation should take place in a contained environment without any network connections as otherwise source code might be used which resides on a server half a world away (and which would likely not be included with the escrow deposit).

Significance: compilation is an essential part and paves the way for establishing that the correct source code has been provided (see deployment/testing). When not properly executed there can still be no guarantee that the source code delivery is complete (see also Validation in next chapter).

Deployment/testing

It is of great benefit to know that the source code provided can be compiled but that says little about the completeness of the source code – without installing the results of the compilation in a test environment and actually testing the results there cannot be any guarantee that the correct source code has been provided (we might as easily have compiled a freeware program downloaded from the internet).

In most cases deployment or testing is a minor issue as both deployment and testing are done by licensee and there is a lot of expertise/documentation on this subject. In certain cases, however, deployment is done by the licensor which adds an additional continuity issue to the escrow arrangement as only the licensor in such cases has the expertise in-house.

Significance: deployment and testing are mainly required to establish that the correct source code was compiled. Depending on licensee's wishes there can be an extensive or only a minor test of the deployed escrow application.

Can the knowledge base present at developer be secured

Securing knowledge is an issue where escrow is concerned. While availability of a complete set of source code will enable a party to maintain the software the presence of as much of the knowledge base as possible will speed up any maintenance process.

While most developers comment their source code and many programs exist to create technical documentation a lot of knowledge is not easily available. Developers are often too pressed for time to properly document their product. In other words the knowledge required can be found solely with the unique individuals.

Significance: the lack of documentation is an issue that cannot be easily remedied. The only thing an escrow agent can do is obtain as much information as possible and make certain that a list of staff is available (who can be contacted should this become necessary).

How does Escrow4all work

The verification approach used by Escrow4all is guided by the following philosophies:

- Offer the best solution available in the market
- Make available flexible solutions based on various standards
- Provide flexibility by using a standard framework which can be modified to suit parties interests
- Provide the most efficient and practical solutions which means licensor and licensee have to spend as little time on the verification as possible
- Provide the most cost effective solution <> quality.

Verifications available from Escrow4all

VerifOne

Basic verification which checks for presence of vital escrow material.

- Readability checks
  - Media
  - Virus
  - Decompression
  - Decryption
- Source code of product licensed present
- Documentation provided

VerifTwo

VerifTwo – advanced verification which focuses on analyzing licensor's environment, collecting the escrow material and compilation/build of that material.

- Discuss verification with parties
- Visit licensor to discuss/analyze development environment
- Compile/build source code in existing development environment
- Deploy/test application in existing environment
- Report on results

VerifThree

VerifThree – complete verification which focuses on making certain that the escrow deposit is complete.

- Discuss verification with parties and create project plan
- Validate source code
- Visit licensor to discuss/analyze application environment
- Recreate development environment starting with a clean system
- Compile/build validated source code
- Recreate test environment starting with a clean system
- Deploy created application in test environment
- Test
- Report on results

Matrix issues/deliverables <> verification

Issue	VerifOne	VerifTwo	VerifThree
Readability checks	Yes	Yes	Yes
Source code product present	Yes	Yes	Yes
Check for technical documentation	Yes (limited)	Yes	Yes
Check for user documentation	Yes (limited)	Yes	Yes
Verification Report provided	Yes	Yes	Yes
Project Plan provided	No	No	Yes
Compilation/build application	No	Yes	Yes
Deploy application	No	Yes	Yes
Test application	No	Yes	Yes
Hardware required	No	Supplier	Variable
Software required	No	Supplier	Variable
Build instructions	No	Yes	Yes
Build instructions fully tested	No	No	Yes
Third party software 100% known	No	No	Yes
Verification procedures outside standard software creation workflow	No	No	Yes
Overview staff	No	Yes	Yes
Complexity process	No	Minor	Variable
Practical large systems	No	Yes	Expensive

Matrix issues <> verification

Issues mentioned in chapter 1 – how certain can parties be that issues have been properly addressed.

Issue	VerifOne	VerifTwo	VerifThree
Is a complete copy of source code provided	Limited	Reasonably certain	Fully certain
Can the development environment be rebuild	Unknown	Reasonably certain	Fully certain
Can the source code be compiled, the results deployed and tested (in a new environment	Unknown	Reasonably certain	Fully certain
Is the knowledge base present at developer secured?	Unknown	Yes	Yes

Validation

To speed up compilation it is possible in most environments to use "components". In other words an object, library or other that has been compiled in the past, and which is now used in its compiled format.

If an escrow deposit is made, such components are often part of the delivery and unless removed, are used during the compilation. If used during compilation and when such a component is from a third party this is not a major problem (might mean your list of third party software is not 100% complete) but if such a component is from licensor ('self-developed component')?

Answer: parties are fortunate when the source code of such components are part of the delivery and when especially fortunate such source code is compiled during the compilation run (the component is replaced by the newly compiled component).

The issue is: nobody knows if such is the case. A compilation run is an automated process. In other words unless action is taken there can be no guarantee that the source code is complete.

How does Escrow4all deal with this in its VerifThree verification?

We validate the source code – all components are removed from the escrow delivery before compilation starts. If the compilation fails there are two likely suspects:

- When creating the development environment not all third party software was installed – install and restart the compilation run

- We are missing source code – provide missing source code, deploy and restart the compilation run.
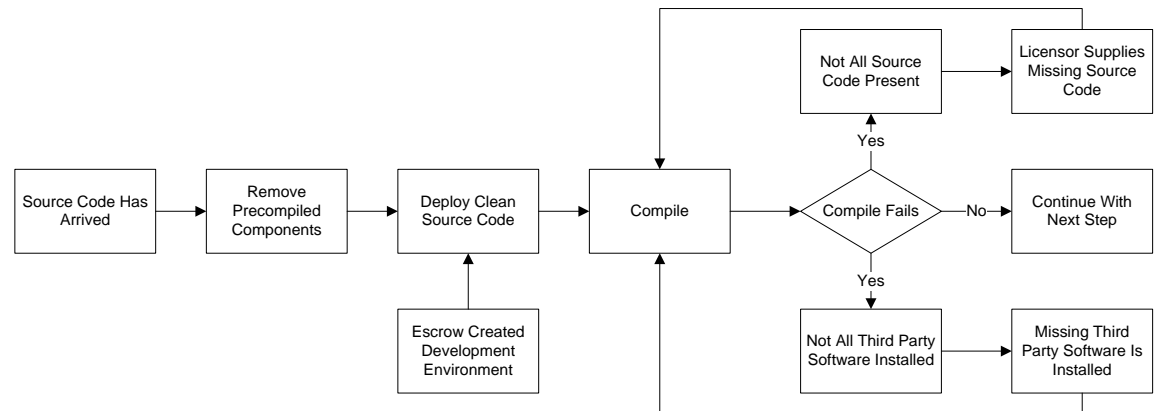
Various issues

- We have found that few escrow deposits are complete at the start of a verification – important is that an escrow deposit is complete at the end; not the problems during the process.

- For VerifTwo and Three verifications Compilation/Build instructions are important. We will use instructions already available during the verification exercise improving them throughout the verification.

- Support during verification:
  - VerifOne – unless there are major issues; none
  - VerifTwo – as we are working on licensor's platforms work is most often done by licensor's staff
  - VerifThree – we prefer to execute most activities ourselves so only support on demand is required (by executing the work ourselves our experts can guarantee the best results).

- Location where main part of verification is executed:
  - VerifOne – Office Escrow4all
  - VerifTwo – Office Licensor
  - VerifThree – Office Licensor.

- Software required for verification provided by:
  - VerifOne – Escrow4all
  - VerifTwo – Licensor
  - VerifThree – Licensor when Escrow4all cannot provide.

- How much time is required on licensor's end:
  - VerifOne – apart from creating the escrow deposit none
  - VerifTwo – depends on the size and complexity of the application but on average 1 - 2 days
  - VerifThree – depends on the size, complexity of the application and internal organization (for example are there already compilation/build instructions) but on average 2 - 3 days.

- How much time is required on licensee's end:
  - VerifOne - none
  - VerifTwo – depends on how the licensee wants to test the escrow created application but normally no more than a day
  - VerifThree - depends on how difficult it is for the licensee to create a new test environment and on how the licensee wants to test the escrow created application but normally no more than one to two days.

- Some licensees request to be present during the verification exercise. If licensor allows this, this is no issue for Escrow4all.

Summarize

When searching for a car insurance, you are more likely to take the best insurance for a (new) expensive car than for a cheaper one. This does not mean that the insurance for the cheaper car is bad – just less extensive.

Escrow verification is the same. While a full guarantee can be achieved with a VerifThree verification this might not always be necessary.

We advise that an initial delivery is verified at VerifThree to deal with/uncover major issues and verify future deliveries at VerifOne until major functionality changes occur.

Contact Information

Escrow4all B.V.

Kingsfordweg 241
1043 GR Amsterdam
the Netherlands

T. +31 (0)20 – 3420 250
F. +31 (0)20 – 3420 259

www.escrow4all.com