

Preamble

This material is released under a Creative Commons License agreement <http://creativecommons.org/licenses/by/2.0/uk/> In informal terms we, the authors, give you permission to do most things with the material provided you:

- a) Acknowledge us as the authors of the original material, and retain the acknowledgement in copy of the material, and in any of your material based on ours,
- b) Do not pretend you wrote it,

Do identify any changes you make as your work, not ours.

The material includes opinions of the authors, and may be different from work published by others. We have quoted and referenced work from third parties, and in a few cases have knowingly used or adapted ideas or content from third parties. Where appropriate we ask for permission from the third parties.

However we make mistakes, as most people do, and may have made mistakes in this work. If you spot something you feel is wrong, or simply something you feel could be improved please let us know via this email address nft.introduction@commercetest.com

We are Stuart Reid and Julian Harty, and are the authors of this material: **An introductory course on non-functional software testing**. If you wish to create derived works you need include the following acknowledgement: at the start and end of the derived works: **This work is based on original material by Stuart Reid and Julian Harty**. We like to know whether our work is being used so please tell us how you're using the content.

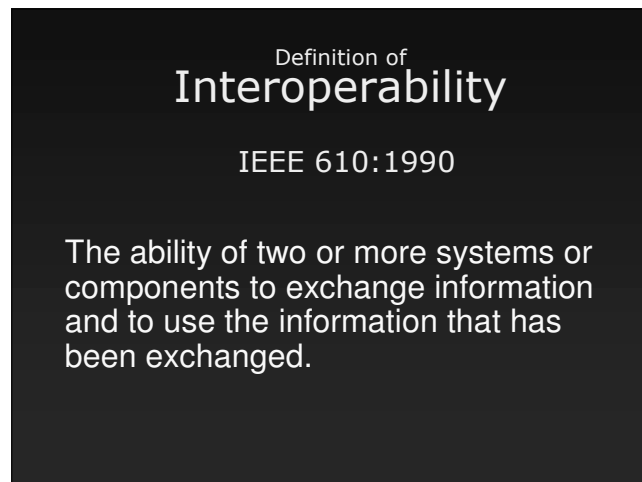
Finally, this work is still in a draft form so you may find inconsistencies, gaps, incomplete content, etc. We hope you will find the content useful anyway and we are very happy to receive suggestions for improvement to the email address mentioned above.

Miscellaneous Quality Characteristics

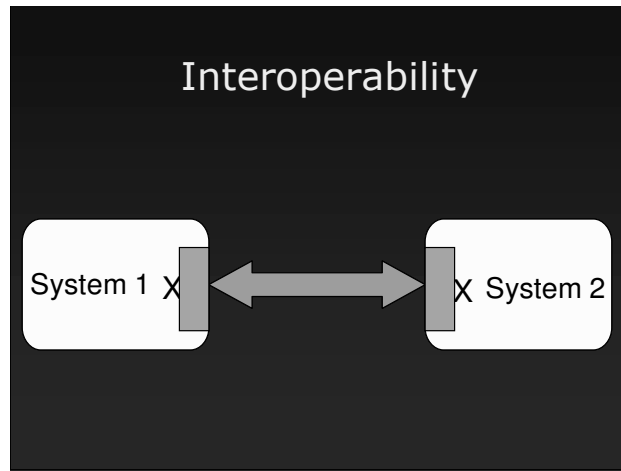
1. The other sections have considered the major quality characteristics of performance, security, usability and dependability. There are, however, several other attributes of a software-intensive system that warrant consideration when specifying the non-functional requirements.



2. In this section we shall consider the testing of the following: interoperability, compatibility, portability, configuration and installability.
3. Before covering these quality characteristics, it is first necessary to define what we mean by the terms 'component' and 'system'. Components are one of the parts that make up a system, while a system is a collection of components organised to accomplish a specific function or a set of functions.



4. As can be seen from the official definition above, interoperability is concerned with the ability of systems to communicate – and it requires that the communicated information can be understood by the receiving system - but it is not concerned with whether the communicating systems do anything sensible as a whole. Interoperability therefore considers the interfaces but not the bigger picture (while integration considers both the interfaces and the bigger picture). Thus interoperability testing is a subset of integration testing.



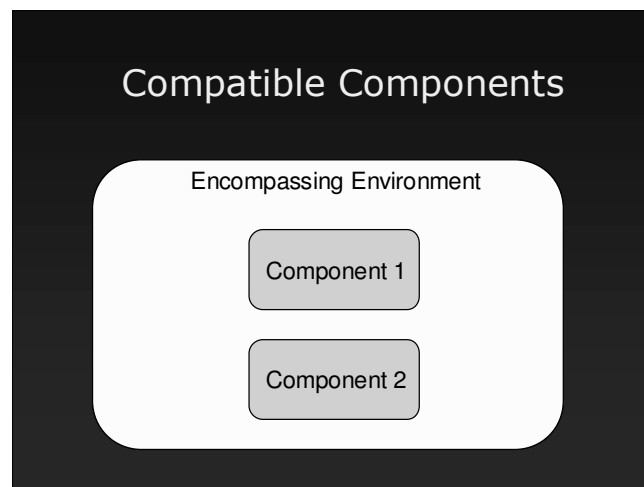
5. The figure above shows two systems communicating with an interface in each system to handle the communication. The interface provides the information for use by the receiving system at the point marked 'X'. Interoperability testing is limited to checking that information is correctly communicated from one system and arrives at the other system at the point marked 'X' in a state in which it could be used.



6. Interoperability is often specified in terms of standards. For instance, consider a requirement for European-wide health cards, which contain the holder's medical information. A standard would be agreed that defined the information held on the card and its format and the card interface. Any cards conforming to the standard would then be interoperable with the different equipment used by the various countries throughout Europe to read from and write to the cards.
7. An example of interoperability testing would be where flight information is passed between the (separate) booking systems for two airlines. The airlines would agree a standard for interoperability of flight information. Interoperability testing would test against the standard to determine whether the information reached the target system and still meant the same thing to the target system as the sending system. It would not test whether the target system subsequently used the booking information in a reasonable manner – this would be part of integration testing.

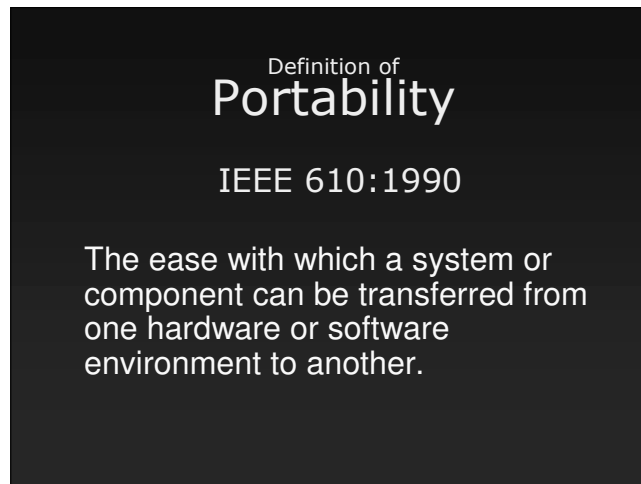


8. Compatibility, defined above, is concerned with the ability to operate correctly while sharing the same environment. The two components (or systems) do not need to communicate with each other, but simply be resident on the same environment – so compatibility is not concerned with interoperability. Two components (or systems) can be compatible, but perform completely separate functions – so compatibility is not concerned with integration, which would consider whether the two components together performed a function correctly.

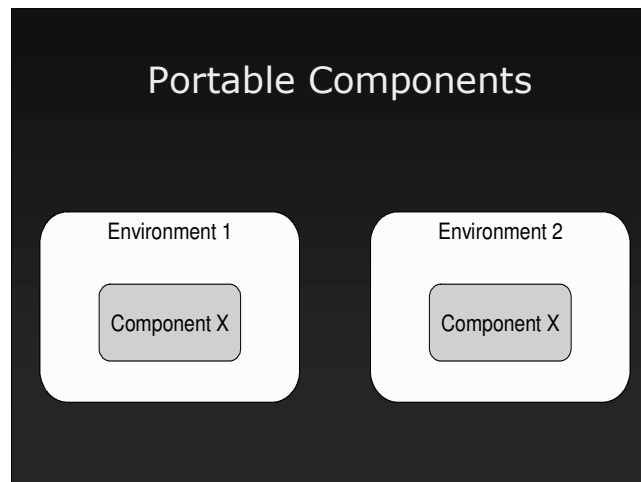


9. The above figure shows two components in the same environment. They are compatible with each other as long as both can run (or simply reside) on the environment without adversely affecting the behaviour of the other. Whether they communicate with each other is not relevant as far as their compatibility in this environment is concerned.
10. The specification of compatibility needs to include three parts. First, identify the components or systems which should be compatible. Next, the required functionality of each the components/systems should be stated (if not stated, it is generally assumed that all the original functionality of a component/system must still be available in the shared environment). Finally, the environment in which they are to co-exist needs to be defined.
11. An example of compatibility testing would be to test whether a word processor application and a calculator application (two *separate* applications) could both work correctly while both are loaded on the same PC. In an example such as this then the compatibility testing would involve scenarios where each application was tested with the other application simply

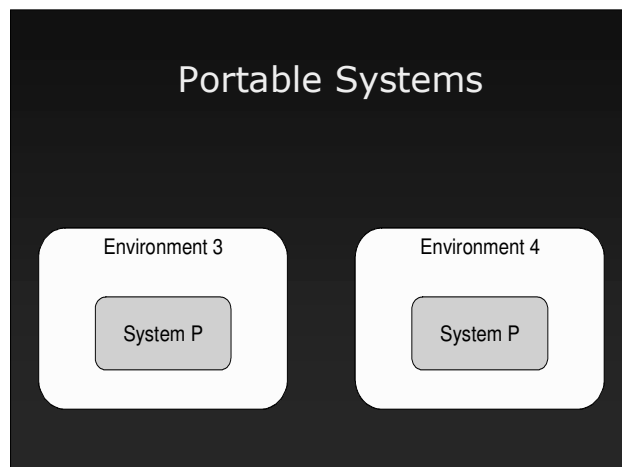
resident on the PC but not running, and also with the other application running as a task in the background.



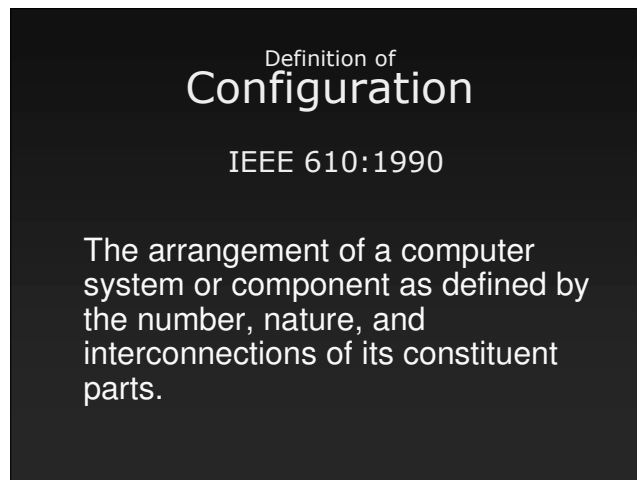
12. Portability is concerned with the ease of moving components or systems between environments (hardware and/or software environments). Not stated in the official definition is the assumption that the transferred component or system operates to the same level of effectiveness in its new environment as it did in the original environment.



13. In the figure above, component X can be seen in two different environments.



14. Similarly, in the figure above, system P can be seen in two different environments. As long as component X and system P can work correctly in the different environments then they are considered to be portable *between these environments*.
15. Portability can be specified both directly and indirectly. When specified directly a measure of the effort involved to port to the new environment is most appropriate. A difficulty that often occurs with specifying portability in this manner is that the target environment is not known at the time of specification, and the target environment often has a large impact on the effort required.
16. A typical approach to specifying portability indirectly is to require the use of a particular programming language. For instance, it is accepted that programs written in Java are far more portable than programs written in many of the older languages, such as C, where many variants of the 'standard' language are in use. Obviously programs written in assembler or machine code are processor-specific and so their portability is very low.
17. An example of portability testing would be when a computer game that worked on a PC running Windows 98 was then tested to determine the amount of effort required to adapt it to run on a PC running Windows XP. Obviously it is possible to specify a zero value for the amount of required effort in portability specifications, implying that the software can be ported to the new environment with no effort at all.
18. As many people know, if you were to perform portability testing of a computer game to *another* PC running a different operating system one of the difficulties might well be that the game did not run correctly in its new environment, but that the problem was not with the operating system, but with another component of the PC, such as a different sound card, or different graphics driver software, etc. These difficulties are really concerned with the PC's configuration rather than portability to a new operating system and so are often considered part of configuration testing.



19. When using the definition of configuration above to define configuration testing, it is possible to interpret it in two ways. The first interpretation, which links in with the previous example of the computer game, is that the configurations of interest are the possible configurations of the environment on which the software application is to run. The second interpretation is that the configurations to be tested are the possible configurations of the software application itself. These different configurations may allow the user to make choices, such as which format and which resolution to record audio for a media application. User choices may be made both during installation and after installation, although those made during installation are normally considered under installation testing (see later).
20. Specifying 'allowed' configurations for the target environment of software is one of the most difficult aspects of non-functional requirements specification, which is why many projects

ignore it. When working on embedded systems the problem is often manageable, but when producing 'shrink-wrapped' software for the general public the number of possible different configurations can become enormously high.

21. Consider the example of software for web retailing that is going to run on customers' web browsers. The software developer has no control of the type of web browser the potential customer is using, nor its version. On average, to hit 99.9% of the market approximately 20 combinations of browser type and version must be covered. If the developer also then considers the different operating systems being used there are again several types (Windows, Mac, Unix, etc.) and many versions. Thus, if we consider all the different combinations of browser and operating system types and versions, we would have to test literally thousands of configurations. This does, of course, ignore all the other factors which may affect the software, such as the different types of hardware and associated software drivers.
22. When faced with a very large number of combinations that cannot all be tested such as with configuration testing, then the use of a systematic test selection technique becomes necessary. There are a number of these including 'base choice', 'each choice', and 'orthogonal arrays'.



23. Installability is defined above, and is concerned with the ease with which a system or component can be installed on an environment so that it can then be used. If installability is identified as worthy of specification for a project, then it is often also necessary to specify the ease with which the software can be uninstalled, as well.
24. When specifying installability one of the major concerns is the "defined target platform" – this means that the configuration of the target environment must be clearly specified.
25. There are two major similarities between installation testing and configuration testing. First, they both require the configuration to be specified. Secondly, they normally both generate an impossibly high number of combinations to test, which means that both make use of the test selection techniques mentioned earlier. With installation testing, apart from the numerous potential target configurations, there are also all the combinations of installation options that can be chosen by the installer. For instance, many applications will require language and geographic region to be entered as part of installation, which, in itself, can generate thousands of possible combinations, and that is before any other application-specific selections are made.

Summary: Quality Characteristics

- Interoperability
- Compatibility
- Portability
- Configuration
- Installability

26. This section has covered a number of the 'less fashionable' quality characteristics. Despite this lack of popularity, these attributes still warrant consideration on every project, if only to note that "for this project this quality attribute is not applicable". For those projects where they are relevant, however, getting these attributes right can make the difference between success and failure.

Self-Assessment Questions

Q1. Which one of the following statements is correct?

- (a) A system is made up of a number of components.
- (b) The terms 'component' and 'system' can be used interchangeably.
- (c) A component comprises a number of systems.

Q2. Which one of the following is most likely to be tested for interoperability?

- (a) The digital camera internal memory and memory card.
- (b) The digital camera's aperture and shutter controls.
- (c) The digital camera image sensor and internal memory.
- (d) The digital camera's internal memory and LCD screen.

Q3. Which one of the following statements is correct?

- (a) Interoperability testing is a subset of integration testing.
- (b) Compatibility testing is a subset of integration testing.
- (c) Integration testing is a subset of compatibility testing.
- (d) Integration testing is a subset of interoperability testing.

Q4. Which of the following is most likely to be tested for compatibility?

- (a) PDA address book and 'Pocket Tetris' game.
- (b) ATM card reader software and the bank database.
- (c) MP3 Player software and the Windows Media Player.
- (d) Mobile phone software and base station software.

Q5. Which one of the following is least likely to appear in a compatibility specification?

- (a) Interface standard.
- (b) Components/systems.
- (c) Required functionality.
- (d) Target environment.

Q6. Which one of the following statements describes the portability of software?

- (a) The latest Star Wars Game is available on PlayStation and Xbox.
- (b) Version 7.1 of Doom allows up to 4 players to play across a network.
- (c) The upgrade to Sim City can only be loaded on top of the original version.
- (d) The new Barbie Princess game requires the XYZ Graphics Card to run.

- Q7. Which one of the following would not be an appropriate way of specifying portability?
- (a) It shall be possible to load the system onto all specified environments in less than 10 minutes.
 - (b) Modifying the system to run on the next version of Windows shall not exceed 3 person months of effort.
 - (c) The system shall be written in Ada using no compiler-specific options.
- Q8. Which one of the following statements about configuration testing is most correct?
- (a) It requires a subset of all possible hardware and software environment combinations to be tested.
 - (b) It requires the installation of software to be tested for a subset of all possible installation options.
 - (c) It requires a subset of all possible combinations of loaded applications to be tested together.
 - (d) It requires a subset of all possible interaction between loaded applications to be tested.
- Q9. Which one of the following testing techniques is least likely to be used as part of installation testing?
- (a) Branch Testing.
 - (b) Base Choice.
 - (c) Orthogonal Arrays.
- Q10. Which one of the following statements is correct?
- (a) None of the statements are correct.
 - (b) Installation testing is a subset of configuration testing.
 - (c) Configuration testing is a subset of installation testing.
 - (d) Configuration testing is a subset of portability testing.
 - (e) Portability testing is a subset of configuration testing.

Self-Assessment Answers

Q1. Which one of the following statements is correct?

(a) A system is made up of a number of components.

Good.

(b) The terms 'component' and 'system' can be used interchangeably.

No – components are one of the parts that make up systems, but components are not made up of systems.

(c) A component comprises a number of systems.

No - components are one of the parts that make up systems, but components are not made up of systems.

Q2. Which one of the following is most likely to be tested for interoperability?

(a) The digital camera internal memory and memory card.

Good – the memory card will be built to a certain standard (e.g. CompactFlash, SmartMedia) and may well be built by a different manufacturer than the camera.

(b) The digital camera's aperture and shutter controls.

No – these are both integral functions of the camera and so will need to be integrated, but will not communicate using particular standards thus will not need interoperability testing.

(c) The digital camera image sensor and internal memory.

No – these are both integral functions of the camera and so will need to be integrated, but will not communicate using particular standards thus will not need interoperability testing.

(d) The digital camera's internal memory and LCD screen.

No – these are both integral functions of the camera and so will need to be integrated, but will not communicate using particular standards thus will not need interoperability testing.

Q3. Which one of the following statements is correct?

(a) Interoperability testing is a subset of integration testing.

Good – interoperability checks communication across interfaces, which is part of integration testing.

(b) Compatibility testing is a subset of integration testing.

No – compatibility testing does not require the components/systems to communicate with each other.

(c) Integration testing is a subset of compatibility testing.

No - compatibility testing does not require the components/systems to communicate with each other.

(d) Integration testing is a subset of interoperability testing.

No - interoperability only checks communication across interfaces, while integration testing checks this and whether or not they function correctly as a whole.

Q4. Which of the following is most likely to be tested for compatibility?

(a) PDA address book and 'Pocket Tetris' game.

Good – these are two separate applications that do not need to communicate, but must run on the same environment without impacting the other application.

(b) ATM card reader software and the bank database.

No – the ATM card reader software and bank database run on different environments in different locations.

(c) MP3 Player software and the Windows Media Player.

No – these two applications do not share the same environment with the media player on the PC and the MP3 Player software obviously on the MP3 Player.

(d) Mobile phone software and base station software.

No – these two applications run on different environments.

Q5. Which one of the following is least likely to appear in a compatibility specification?

(a) Interface standard.

Good – compatibility considers whether applications can share environments, not whether they can communicate, so an interface standard is not relevant when specifying compatibility.

(b) Components/systems.

No – compatibility considers whether component/systems can share environments and so those to be considered must be specified.

(c) Required functionality.

No – the required functionality of the component/systems whose compatibility we are checking needs to be specified. Often this is the same as the original functionality, but not always.

(d) Target environment.

No – compatibility considers whether component/systems can share environments, so we need to be clear about what environment(s) the components/systems can share.

Q6. Which one of the following statements describes the portability of software?

(a) The latest Star Wars Game is available on PlayStation and Xbox.

Good – portability is concerned with being able to move to different platforms.

(b) Version 7.1 of Doom allows up to 4 players to play across a network.

No – portability is not concerned with networking, or communicating across a network, but rather about being able to move to different platforms.

(c) The upgrade to Sim City can only be loaded on top of the original version.

No – portability is not concerned with how applications are modified, but rather about being able to move to different platforms.

(d) The new Barbie Princess game requires the XYZ Graphics Card to run.

No – portability is not about the configuration of the target environment, but rather about being able to move to different platforms.

Q7. Which one of the following would not be an appropriate way of specifying portability?

(a) It shall be possible to load the system onto all specified environments in less than 10 minutes.

Good – portability is about being able to move to different platforms, not about the speed of installation.

(b) Modifying the system to run on the next version of Windows shall not exceed 3 person months of effort.

No – portability is about being able to move to different platforms, which this is describing and in a quantitative manner, which is better than average.

(c) The system shall be written in Ada using no compiler-specific options.

No – portability is about being able to move to different platforms, and one way of specifying portability indirectly is by specifying a language, such as Ada or Java, which makes it easier to port.

Q8. Which one of the following statements about configuration testing is most correct?

(a) It requires a subset of all possible hardware and software environment combinations to be tested.

Good – configuration testing is about testing different possible hardware and software configurations, but it is rarely possible to test them all.

(b) It requires the installation of software to be tested for a subset of all possible installation options.

No – the testing of combinations of installation options is known as installation testing.

(c) It requires a subset of all possible combinations of loaded applications to be tested together.

No – combinations of applications are tested as part of integration testing if overall functionality is the concern, or compatibility testing if simply checking the applications can reside on the same environment is the concern.

(d) It requires a subset of all possible interaction between loaded applications to be tested.

No – interaction between applications is the concern of interoperability (to simply check communication) or integration if overall functionality is the concern.

Q9. Which one of the following testing techniques is least likely to be used as part of installation testing?

(a) Branch Testing.

Good – branch testing is a white box testing technique aimed at exercising branches at the code level.

(b) Base Choice.

No – Base Choice is a systematic technique for selecting a subset from a large number of combinations, which is necessary for installation testing as there are usually a large number of installation options which can be combined.

(c) Orthogonal Arrays.

No – Orthogonal Arrays is a systematic technique for selecting a subset from a large number of combinations, which is necessary for installation testing as there are usually a large number of installation options which can be combined.

Q10. Which one of the following statements is correct?

(a) None of the statements are correct.

Good.

(b) Installation testing is a subset of configuration testing.

No – installation testing is concerned with testing installation options of the application being installed, whereas configuration testing, when applied to an application, is concerned with the resultant run-time configuration (after installation).

(c) Configuration testing is a subset of installation testing.

No – installation testing is concerned with testing installation options of the application being installed, whereas configuration testing, when applied to an application, is concerned with the resultant run-time configuration (after installation).

(d) Configuration testing is a subset of portability testing.

No – configuration testing is concerned with testing whether software will work with the various different combinations of hardware and software, while portability testing is concerned with being able to move applications to new environments.

(e) Portability testing is a subset of configuration testing.

No – configuration testing is concerned with testing whether software will work with the various different combinations of hardware and software, while portability testing is concerned with being able to move applications to new environments.

Note that the correct answer for all questions is currently set to (a).

References

www.testingstandards.co.uk - Useful reference on non-functional testing.

-- End of document --