



Testing is a waste of time

says Angela Edwards, Senior Consultant at RelQ

Yes! Testing IS a waste of time – if, at the end of testing, you still cannot say with certainty whether all the requirements, and only the requirements, have been met.

This article sets out to describe why “requirements traceability” is important, how it provides a significant element of risk management, and how it can be tailored to balance time, cost and risk.

Requirement traceability

So what is requirements traceability? Well, in a software development project, it is the tracing of links between the stakeholders’ requirements and the deliverables produced during the software development lifecycle.

Through requirements traceability, the project team can identify which deliverables implement which requirements, and therefore what tests are necessary to verify that these requirements have been implemented successfully. The objectives of requirements traceability are to:

- ensure that the deliverables meet the requirement
- ensure that no requirements are missed

and therefore that all are delivered

- identify any deliverables that cannot be traced back to a requirement. This may be due either to undocumented requirements or to the unnecessary introduction of ‘features’ that are not actually required by the stakeholders.

Figure 1 shows a simplistic version of the W-Model of the software development lifecycle. The W-Model is better than the more common V-Model because it makes clear the importance of testing throughout the project cycle. In particular it shows that the test planning and preparation activities take place as early as possible during the project lifecycle.

Hopefully most testers will identify with the W-Model. However, requirements traceability is applicable to projects that use other models of the software development lifecycle.

Full requirements traceability will produce a hierarchical model from the higher level requirements through detailed requirements and design, down to the individual software units. This will follow through the corresponding test components, finally to project deliverables themselves.

Figure 2 shows the W-Model with traceability links added, and figure 3 provides a diagrammatic example of this hierarchy using traceability matrices, which are the most commonly used technique. There are many other techniques used for documenting traceability, they differ mainly in the breadth and depth of details and links stored. This diagram shows traceability of requirements to both unit testing and user acceptance testing results.

Building a traceability matrix is straightforward. In figure 4, requirements have been traced to design specification components. All requirements are listed on the left-hand side of the matrix; all design specification components are listed across the top. Once this is done, the links between the requirements and the design specification components can be added and reviewed.

Where a design specification component has no requirement linked to it (eg #56), further investigation is required to identify whether the requirement is actually missing or whether this design specification component has been added without agreement with the stakeholders.

Where a requirement has no design specification component linked to it (eg #5) further investigation will be required to establish whether it is still a valid requirement.

You are likely to find some many-to-many relationships as is shown in the diagram. Many design specification components may be required to satisfy a single requirement and many requirements may be satisfied by a single design specification component. However if there are a high number of many-to-many relationships it may be that either the requirements have been expressed at the wrong level of detail or they need additional clarification.

Requirements will include non-functional attributes as well as the functional requirements. With the increasing pressure on businesses either to keep up with their market place or to fall behind, non-functional requirements are becoming more and more important, especially those concerned with usability,

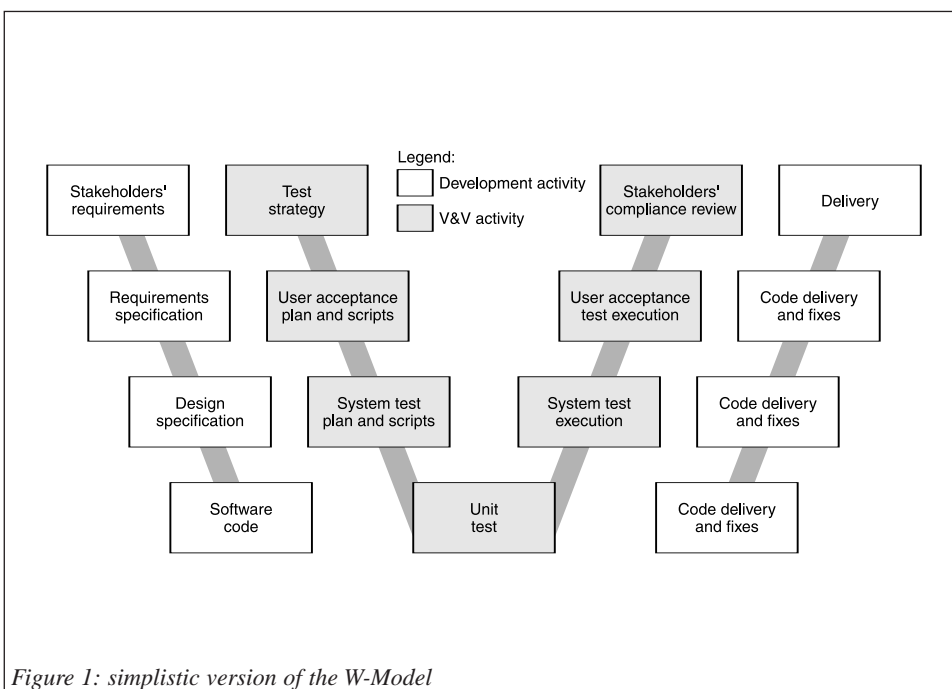


Figure 1: simplistic version of the W-Model

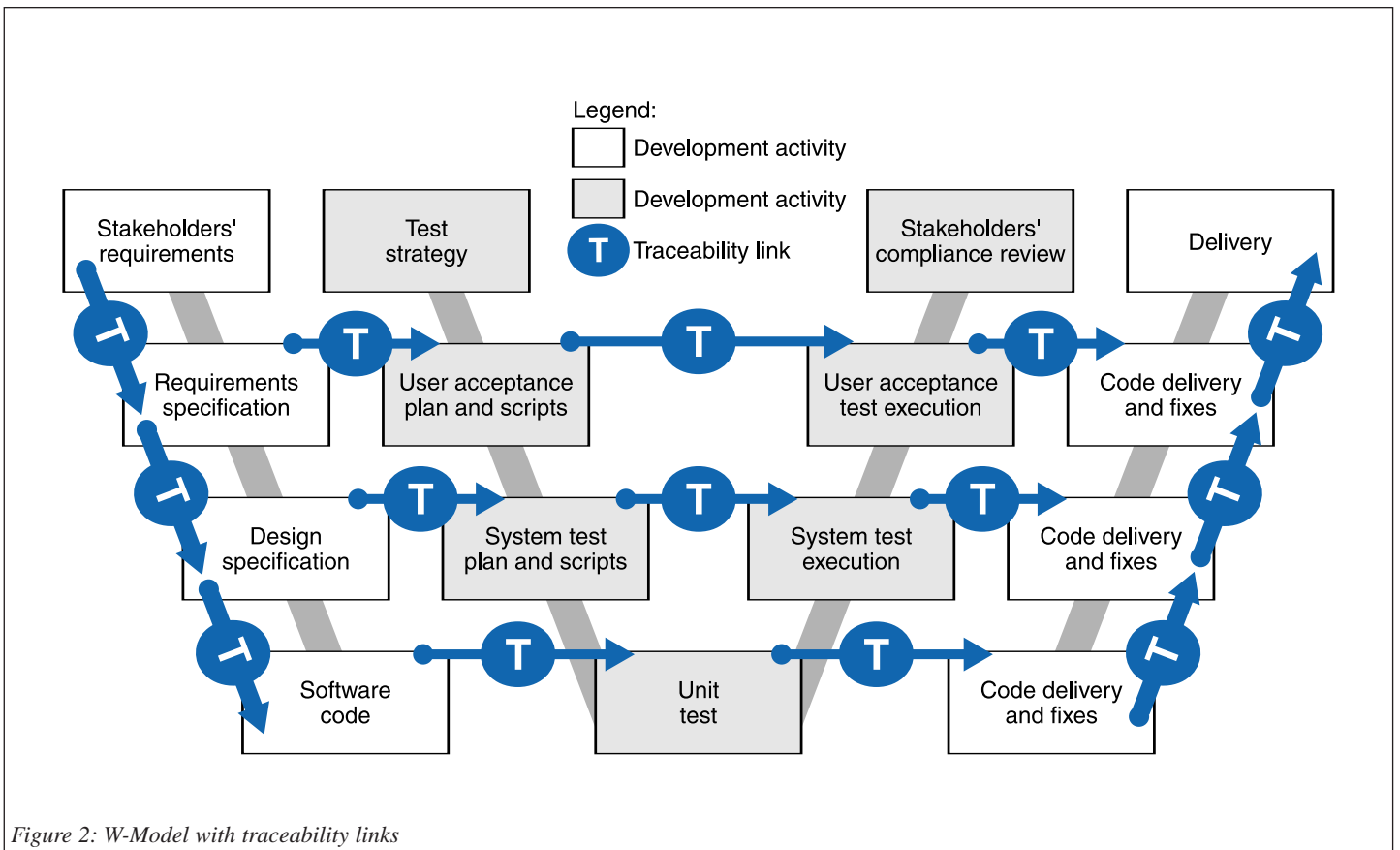


Figure 2: W-Model with traceability links

performance and security. This is because with e-business, the user is no longer a captive employee of the business but a customer with the option to go elsewhere. Thus, an application will have requirements to display data within a specified time frame, in an easily 'usable' manner (we can discuss 'usability' in some other article), in a secure fashion, and in addition to the requirement to function correctly.

Requirements change throughout the project lifecycle; there will be many reasons for this, including changes needed in response to competitor products, new audit requirements and changes in the regulatory framework or government legislation. The traceability process must be robust enough to cater for those changes. Once in place, traceability will greatly assist the change impact analysis process, because it is now easy to see which project deliverables are likely to be affected by a changed requirement.

Targeted requirements traceability

In an ideal world, all project managers would like to have the time, money and resource to see full traceability of the requirements throughout the whole project lifecycle. The ability to trace the requirements to the deliverables produced during the project is a significant factor in assuring a quality software implementation. This level of traceability is absolutely essential on safety and mission critical projects.

However in a time where projects are under increasing pressure to reduce both time to

market and cost whilst still maintaining quality, we must ask ourselves what value this traceability adds and if the cost of that value is acceptable. We must consider whether targeted, risk-based traceability would add a similar value at much lower cost.

Too high a level of traceability will add unnecessary time and cost to the project in terms of initial capture and subsequent maintenance. This may be too high a cost with insufficient gain. Too low a level of traceability and testing cannot be proved to have achieved its objectives.

As testers we are most likely to be interested in traceability between requirements, project deliverables and the tests we need to perform to satisfy those requirements. With traceability in place, we will be in a position to demonstrate to the stakeholders, at any stage during testing, the degree to which the requirements are satisfied.

However we do not need to trace all requirements everywhere. What is needed is a business risk analysis of all requirements that will assign a business criticality index to each. Those requirements that are 'critical' to the business and project needs MUST be demonstrated as being satisfied. These are the requirements that we will test thoroughly, as a minimum, and these are the requirements we will trace through to our test results.

Thus a minimum set of requirements will be identified and traced. Adding any other requirements to that minimum set will require justification.

This assumes that we have a well-documented set of requirements. What do we do if we have little or no requirements to work from?

Requirements traceability with no requirements

An approach that has been used successfully is to document the test team's understanding of the requirements in the form of Acceptance Criteria. These are not as detailed as requirements would normally be. However they will document what needs to be proven and will consist of testable statements. The acceptance criteria need to be signed off by the stakeholders. Otherwise - well, you guessed it - you may as well not bother because your testing will be a waste of time.

What tools are available to help?

There are many tools on the market place that will assist in the process of traceability. These are split into four categories, though the value of creating your own "in-house" tools should not be ignored.

- traceability tools
- requirements management tools
- test management tools
- code analysis tools

Traceability tools

These are tools tailored specifically to this process. Typically, these tools are only considered when there is a long-term project commitment to requirements traceability or

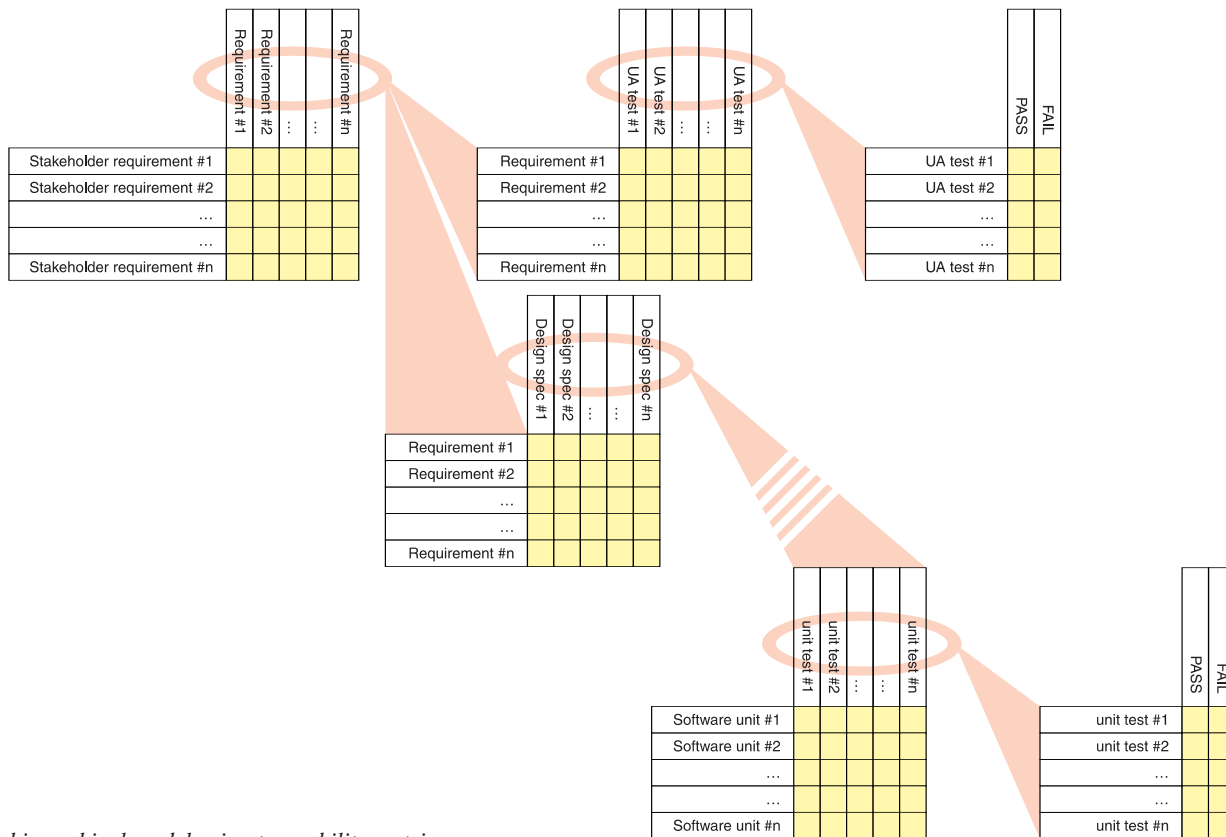


Figure 3: hierarchical model using traceability matrices

where the domain absolutely requires it. Some of these tools contain a multitude of details such as the history of the traceability link and details of the traceability link eg “is derived from”, “is tested by”. This level of detail is unlikely to be required on the majority of projects, though it may be extremely useful on safety or mission critical projects where there is no room for failure.

Requirements management tools

These tools usually come into service at the beginning of the software development lifecycle. If they are already being utilised by the project, and if the tool supports requirements traceability, they provide a comprehensive and easy way of documenting and retrieving traceability data at little additional cost. For these tools to be used, the responsibility for maintaining requirements traceability throughout the software project lifecycle must fall within the requirements area.

Many CASE tools support requirements traceability to deliverables further down the software development lifecycle, though not all. This traceability is often achieved via import/export mechanisms; this is obviously not ideal, as it is dependent on up to date documents. Where it is possible to influence the purchase of CASE tools this is an additional consideration to be explored.

Test management tools

These tools provide the ability to enter cross-references from requirements to the various test deliverables. Experienced testers

often maintain these details as a matter of course. Though as with the CASE tools, many test management tools achieve cross-referencing to requirements via import/export mechanisms. As discussed above this requires up to date documents. With testing being at the end of the various traceability chains, it will be effected more by changes made earlier in the project.

Code analysis tools

These tools analyse the actual software itself to identify the code structure and paths through the code when it is executed. Used in conjunction with full code coverage testing, they are particularly helpful in identifying code that is never executed during testing of the requirements. This is especially useful for checking safety or mission critical developments where every line of code must be traced back to a requirement. It is also extremely useful for identifying potential fraudulent code within financial applications.

Conclusion

Some form of requirements traceability is essential on all software development projects. Otherwise stakeholders’ acceptance of the project deliverables cannot be based on an objective assessment that the requirements have been meet and, therefore, the testing has achieved nothing.

In a world where we are being increasingly asked to achieve more

and more, for less and less, we need to ask ourselves what we can do to ensure that testing provides the most cost-effective value to the business. The answer is that we need a practical, pragmatic approach that is in line with the business and project needs. To achieve this we need to understand which are the business critical requirements. By tracing these and only these requirements through the tests and the subsequent test results, it is possible to provide the business with a cost effective test solution that meets their critical project requirements.

Applying a business-risk-based approach to requirements traceability will ensure that the critical requirements have been met. It helps to manage risk, to add value to the business and to ensure testing is not a waste of time.

	Design spec component #1	Design spec component #2	Design spec component #3	Design spec component #4	Design spec component #5	Design spec component #6	Design spec component #56	...	Design spec component #n
Requirement #1	✓	✓							
Requirement #2					✓				✓
Requirement #3		✓							
Requirement #4				✓	✓				✓
...									
Requirement #38					✓				
...									
...									
Requirement #n					✓				

Figure 4: traceability matrix - requirements to design specification components