Risk-based test planning during system development[1]

Felix Redmill
Redmill Consultancy

Abstract

This paper proposes carrying out test-oriented risk analyses during the software and system development life cycle in order both to improve the software and to inform test planning. It offers an overview of risk analysis and explains where in the life cycle it would be useful to apply it.

1 Introduction

Because of the complexity of software, its testing can almost never be exhaustive. It must therefore be selective. The way in which test planning is carried out is therefore crucial, not only to finding defects but also to doing so cost-effectively, and in the time available. Many testers and test planners claim to base testing on risk, but most cannot do so successfully because of a lack of knowledge of the subject of risk. Yet it makes admiral sense to focus testing where the risks are greatest. In a previous paper the current author explored the subject of risk-based testing and showed that its application requires not only an understanding of software testing but also knowledge of risk and its analysis [Redmill 2004].

The simple model of Figure 1 shows testing as the second stage of a three-stage process. If test planners consider risk at all, it is almost always the ‘forward risks’ that they look for. These are the risks associated with system operation and which involve the consequences of software failure. But there are also ‘backward risks’, which arise out of development problems, such as incorrect specifications, inappropriate design, casual programming, and inadequate management. They also include what takes place across the interface between the developers and the testers. Late delivery of software items, and their delivery out of the agreed order, can considerably affect the time and effort needed to carry out the planned test programme.

----
[1] This was an invited paper at the Polish national software engineering conference, KKIO 2004, Gdansk, Poland, 5-8 October 2004

Risk is a function of two components: the probability of occurrence of a defined undesirable event and the severity of the consequences if the event does occur. When considering the risks attached to software failure, the consequence component is the focus of the testers' forward risks, but the probability component arises out of the backward risks. Because in most cases the consequence of failure is linked to the software's purpose, reducing it is not an option. Thus, risk reduction must be achieved by reducing the probability of failure. Typically, testing seeks to commence the process of reducing probability by finding defects. Then, actual reduction depends on whether and how the developers implement corrections.

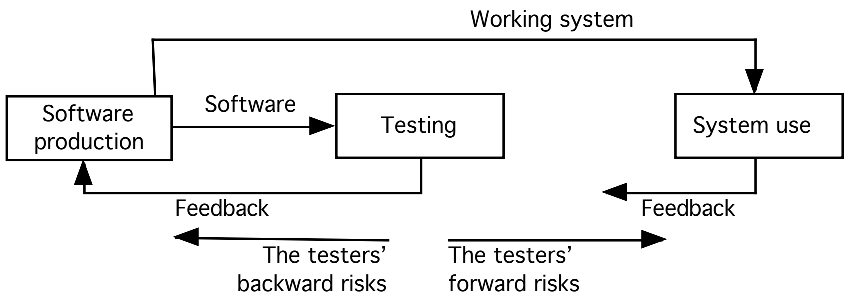


Figure 1: Testing as the second of three processes

In mechanical equipment, for which failure is a random occurrence, it may be possible to determine the probability of failure from historic data. In the case of software, however, for which failure is systematic and not random, historic data is not predictive, even if it is carefully collected. And when the software has not yet been produced, as is usually the case when test planning is carried out, there is no basis for determining the likelihood of failure. At that time, however, the potential consequences of failure may be estimated. Then, a risk-based approach to testing may take the form of a 'single-factor analysis' based on consequence [Redmill 2004], the results of which are used to focus testing on the software whose failures would have the greatest consequences. The purpose is to reduce the risks by reducing the associated probabilities of failure. This is not a trivial process, for it requires a thorough investigation of failure consequences with respect to all the system's stakeholders. An intuitive approach to risk analysis is inadequate, and an approach based on knowledge of risk analysis is required.

Later, when the software has been written, it may be possible to arrive at a probability estimate, by examining characteristics of the code, such as structure and complexity, and by considering the quality of code previously produced by the designers and programmers. This probability-based single-factor analysis cannot produce an infallible estimate, but it can help the testers to focus on the code that is likely to contain the largest numbers of defects.

But these forms of test planning are carried out after the defects have been built into the software. There is a proverb that 'You can't make a silk purse out of a sow's ear'. Quality needs to be present from the start. Testers may find imperfections and show the way to improvement, but they do not themselves make the software better. Indeed, testing comes late in the development cycle, when most of the principal decisions have been made and the foundations for quality have already been laid, as shown by the fishbone diagram of Figure 2.

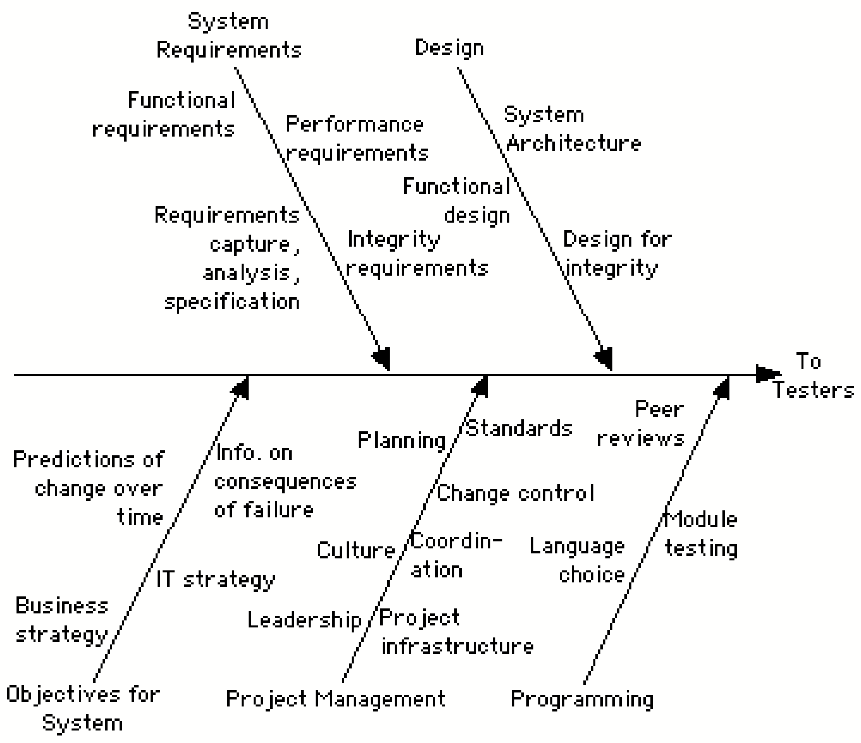


Figure 2: Examples of activities prior to software being delivered to testers

Figure 2 does not present an exhaustive list of development activities, or place them in a definitive order. It does not, for example, include test planning or activities that recur at each stage, such as management and verification. It is intended to emphasise that the testers' ability to carry out their own tasks efficiently is strongly influenced by preceding events. They do not have control over software quality. Deficient work in the management of a project or in the technical development of a system creates problems for the testers and results in an increased probability of residual bugs, failures in service, and the need for maintenance and change to the system – all of which increase system life-cycle costs. The backward risks are the root causes of the forward risks. Reducing them would go a long way to achieving higher-quality software and reducing system life-cycle costs. This paper examines how a risk-based approach could be used to improve both development processes and test planning.

The next section offers an overview of risk analysis and management, both to inform those unfamiliar with the processes and to provide a basis for the proposals subsequently put forward. This is followed by an explanation of the points in the development life cycle when a formal risk analysis is appropriate. Finally there is a discussion on the application of risk analysis as proposed in the paper.

2 An overview of risk analysis and management

The vocabulary of risk is not universally standardised. The terms 'risk analysis' and 'hazard analysis' are both used variously and sometimes synonymously, and 'risk management' is used sometimes to embrace them both and sometimes only to cover the decision-making and action-taking that follow them. In this paper, risk analysis is taken to embrace the identification, analysis and assessment of the potential for future problems, and to consist of four stages or activities:

- Scope definition, in which the scope and terms of reference of the analysis are defined.
- Hazard identification, in which the things that could go wrong, given the context of the analysis, are identified.
- Hazard analysis, in which the identified hazards are analysed so as to estimate the risks that they pose. This involves estimating the likelihood of a hazard materialising into an accident (or hazardous event), and the potential consequences if it did.
- Risk assessment, in which the risks posed by the hazards are assessed against tolerability criteria.

As mentioned above, the terms used in risk analysis and management are not universally agreed. Sometimes the scope definition stage is not mentioned in definitions of risk analysis. Sometimes the hazard analysis stage is referred to as risk analysis, with the overall process being called risk management. Sometimes a 'risk management' stage is added to the process, and sometimes this final stage is referred to as 'risk reduction'. However, the title of 'risk reduction' is unsatisfactory because the final stage is not limited to risk reduction but is concerned with making decisions between various risk-management strategies, including not reducing some risks. For consistency in this paper, 'risk analysis' embraces the above four-staged model and is followed by the decision-making, action-taking activities of 'risk management'. In the context of risk-based testing, risk management involves the use of the results of risk analysis to inform test planning.

In general, risk analysis would not provide the only basis for decision-making, for there may also be political, legal, financial, and other information to consider. In system development, there may be similar factors, as well as time and other project constraints. However, the purpose of this paper is only to address the use of risk to inform test planning, and the consideration of other information is left to project decision-makers.

In software testing, risk analysis may be carried out at a high or low level, rapidly or thoroughly, with a broad or narrow remit, depending on its objectives. Sometimes the hazard identification and hazard analysis stages may be merged. But a significant advantage of risk analysis is that it imposes a methodical approach and, in order to maintain this, the four stages should always be identifiable, even when they are not separately defined. A further point is that, as the risk analyst is usually not the decision maker, care needs to be taken in both the documentation and the communication of the results of risk analysis. The following sub-sections offer further information on the four stages, with an added note on risk management.

2.1 Scope definition

A crucial determinant of any study – feasibility study, public inquiry, or risk analysis – is the definition of its objectives and terms of reference. If the objectives are not explicitly defined, they are unlikely to be met; if the terms of reference are not clearly stated, the study may be conducted too narrowly to achieve its purpose or may range so widely as to cover irrelevant issues and exceed its intended budget. Because they define the field of inquiry, the terms of reference very strongly influence a study's results, and care should be taken in defining them. Examples of issues that should be defined in this stage are:

- The physical and logical boundaries of the system to be analysed. A single system may output only to paper or may control equipment that is geographically co-located with it. A distributed system may consist of logical and physical elements that are located in many parts of the world. An internet system may physically be located in only one place but may potentially have effects on numerous people globally. In each case the boundaries of both the system and its relevant effects must be carefully defined if a risk analysis is to be effective.
- The scope and limits of the study. The study's boundaries may not coincide with those of the system to be studied, so they need to be defined separately.
- The types of risk to be addressed. In commercial or industrial contexts, a study may be intended to address only financial or only safety risks. In a development project, the purpose of an analysis may be to investigate the risks to on-time completion rather than those associated with failure of the system – which are the risks most likely to be of concern in the context of software testing. Test planning may also be interested in the risks of software being of low quality, or those of security breaches of the system, from both external and internal sources.
- The types of information that should be collected. In the present context, the information required would certainly include the consequences of system failure, and it may also include details of the system's development. In most cases it is important to include details of the system's operational environment.
- The necessary, proposed, or allowable, sources of information. For testing, the system's stakeholders are likely to be appropriate sources. Omitting a significant

stakeholder could lead to a considerable underestimate of potential consequences.

- How the results will be used. The intended use of results could affect the way in which a risk analysis is conducted. In the present context, it is assumed that the use will be to inform test planning.
- Tolerability criteria. Criteria against which the acceptability or prioritisation of risks will be judged should be defined.

There should be a scope-definition stage at the start of every risk analysis. In planning an analysis to inform risk-based testing, some (but not necessarily all) of these points will arise, and it is the responsibility of the analyst to ensure that the study's scope is fully defined and preparation for it is carried out thoroughly. However, the subject of scope definition will not be described in further detail.

2.2 Hazard identification

In risk analysis the sources of risks – the things that could go wrong and cause undesirable events – are typically referred to as hazards. Hazard identification is the stage of risk analysis in which they are methodically searched for and identified.

A risk analysis may be given the broad objective of addressing all types of undesirable event, or it may be confined only to particular types. In some cases, a system owner may be most concerned with the risk of the system not being brought to market on time, and a study may be launched with the objective of addressing the hazards that could lead to that single undesirable event. For test planning, the undesirable event may be system failure, of which there are likely to be many types, with many possible causes of each. Or, considering risk from the point of view of development rather than use, the undesirable event may be the production of software that does not meet its quality requirements.

Effective hazard identification demands not only a methodical approach but also creativity. In simple cases, where there is plenty of historic information on failures and how they occurred, a checklist can be helpful. A disciplined examination of the results of audits, complaints, and staff interviews can provide the basis of learning from what has occurred in the past. But when novelty is involved, as in bespoke software, there is the likelihood – indeed, almost always the certainty – of new hazards whose identification requires creative investigation. Brainstorming sessions, if well managed, can throw up many good ideas in a short time. But brainstorming is not based on a model of the system under study and so cannot be guaranteed to uncover its most hidden or most severe faults. The most powerful method of hazard identification is hazard and operability studies (HAZOP) [Redmill et al 1999], which not only includes planning, careful team selection, and strong management of the process, but also the use of 'guide words' to focus the study.

Methodical hazard identification can be time-consuming, and corners are often cut in carrying it out. Yet it is critical to effective risk analysis, for hazards not identified are neither analysed nor mitigated.

2.3 Hazard analysis

Hazard analysis is intended to explore the identified hazards and determine the risks that they pose. As risk comprises two components, the probability of an undesirable event and the event's potential consequences, the purpose of hazard analysis is to determine values for these two variables and to combine them, quantitatively if this is feasible, or otherwise qualitatively, to arrive at risk estimates. This is the general case. But with software, and particularly newly written software, arriving at a value for the probability of failure is problematic, and a legitimate form of analysis is single-factor, based only on the consequences of failures, which provides a basis for determining which probabilities it is most worthwhile to reduce by finding and fixing defects [Redmill 2004]. The risk analyses recommended in this paper are carried out during the stages of software development and may address both the consequences and the probabilities of identified undesirable events.

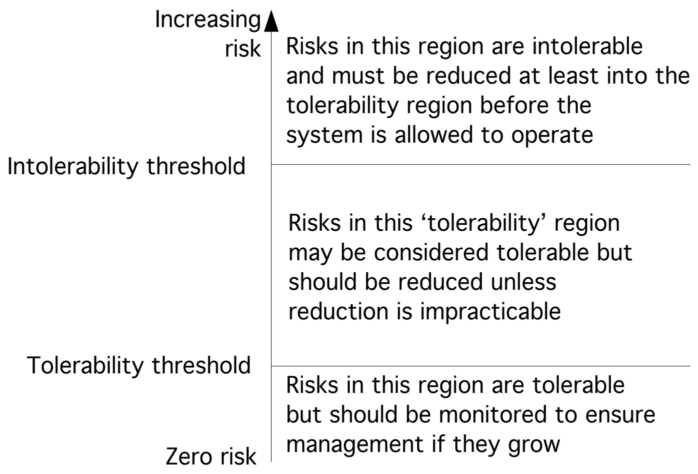
2.4 Risk assessment

All risks are not of the same magnitude. Their consequences and probabilities both vary. Similarly, mitigating actions differ according to the risk. Some risks are small and may be considered tolerable as they are. On occasions, significant risks may be accepted because of the benefits to be had from taking them and because of the cost of reducing them. Thus, mitigating action depends on how tolerable a risk is deemed to be. Determining tolerability is the purpose of the risk assessment stage.

When quantitative risk analysis is carried out, assessment is made by comparing the determined risk values against numeric criteria. When analysis is qualitative, risk values may be expressed as categories (e.g. high, medium and low). In either case, a structure for assessing risk may be based on a simple model (see Figure 3, in which three regions are stacked vertically against an axis of increasing risk). If the value of any risk is above a defined 'intolerability' level or threshold, the system cannot be deployed unless the risk is reduced to a tolerable level. If a risk is below a defined 'tolerability' threshold, it is considered acceptable as it is. Risks between the two thresholds may be deemed tolerable, but they should be reduced unless reduction is not cost-effective. This three-tiered model reflects the way in which humans deal with risks in every-day life, and it also reflects the Health and Safety Executive's ALARP (as low as reasonably practicable) model [HSE 2001].

This basic model of risk tolerability needs to be calibrated for each application by defining its thresholds in the scope-definition stage of a risk analysis, the units of calibration being determined according to the context of the study. In most studies, they are units of currency. In safety they may be numbers of lives lost per year.

Three regions are in most cases adequate, but a fourth can be included by dividing the central 'tolerability' region into two. Further, when the model is used not to define tolerability but as the basis for risk-management action, or for prioritising the actions to be taken, any number of regions (or risk classes) may be defined, depending on the number of categories of action or prioritisation deemed necessary.



Increasing risk
Risks in this region are intolerable and must be reduced at least into the tolerability region before the system is allowed to operate
Intolerability threshold
Risks in this 'tolerability' region may be considered tolerable but should be reduced unless reduction is impracticable
Tolerability threshold
Risks in this region are tolerable but should be monitored to ensure management if they grow
Zero risk

Figure 3: Three regions of risk space

2.5 Managing the risks

At the end of risk analysis, the risks and their tolerability are known, so far as the available information and the thoroughness of the analyst allow. Decisions on what to do about them must then be made and implemented.

The purpose of the assessment stage is to inform risk-management decisions, and there are three ways in which the principles of the assessment process may be used. First, they can be applied according to tolerability, such that they define the appropriateness of a system to be deployed. The general principle is that risks below the tolerability threshold may be left as they are, though monitored to make sure that they don't increase, those above the intolerability threshold must be reduced or eliminated, and those between the two thresholds should be reduced or otherwise mitigated if practicable. Second, the regions of the model can be equated to programmes of risk-management action to be taken, and this is the method used in some risk-based testing. Third, the regions may be equated to levels of prioritisation, such that counter-measures against the risks of highest priority are implemented first.

3 An overview of test-orientated risk analysis in the development process

The most effective risk analysis is carried out not once, or even repeatedly, but continuously. That is to say, a mechanism is put in place – and used – for analysing, recording, and managing new hazards, as and when they are identified. Yet, there are points during system development when it is appropriate to carry out formal analyses, each conforming to the risk-analysis model outlined in Section 2. These points arise when new information becomes available and when significant changes are planned. In most projects they occur when the following have been produced:

- The objectives for the system;
- The specification of requirements;
- The architectural (or structural) design;
- The detailed (module) design.

The results of risk analyses at these points could be used as the bases not only for reducing identified risks and improving the development process, but also for test planning. Indeed, all but the first of them coincide with the critical points in the V Model [STARTS 1986] (see Figure 4), which models the development life cycle from a testing perspective. The rectangles in Figure 4 represent life-cycle phases and the ellipses represent their products. The dotted horizontal lines point out that the entities on the left are the bases for those on the right, that the latter should conform to the former, and, therefore, that the tests to prove those on the right should be derived from those on the left. The model takes a testing perspective.

The plans for testing the products on the right of the V should be aimed at determining their conformity with the entities on the left – which, therefore, test planners must understand fully. Often, however, they do not know which parts of those entities are of greatest importance, which are the most difficult to implement, or which would throw up the greatest problems if defective or not implemented. If risk analyses were carried out on the entities, risks could be managed in three principal ways.

- Immediate action could be taken by making adjustments in the life-cycle phase in question. For example, a risk identified at the objectives stage might be reduced or eliminated by changing the objectives, abolishing the relevant aspect of the project, or abandoning the project altogether.
- Risk mitigation action could be taken in the subsequent life-cycle phase, because the identified risks show where particular care should be taken. For example, a risk identified at the requirements stage might be reduced by providing appropriate redundancy in the design.
- In all cases testing could be appropriately planned to ensure the efficacy of the risk-mitigation features, and, more generally, to provide a further layer of risk management.

Risk analysis also attempts to attribute values to risks, either in absolute terms or relative to other risks, so it allows prioritisation for management action. But risk management is not simply about avoiding risk. It is, importantly, about making decisions about how to handle them. In some cases, high risks will be taken because the expected benefits are high – but at least the risks are understood and measures are taken to manage them, including, perhaps, the putting in place of contingency plans. In some cases, it is not worth spending a great deal of time trying to calculate probabilities, especially when the necessary data is absent. If the risk is going to be accepted, despite the potentially high consequence if it matured, the issue is not what the probability is but rather how can it be reduced to a tolerable level. Risk management therefore depends as much on decision-making – based on evidence – as on taking action.

A point worth making is that analysts should in all cases take the system's operational environment into consideration. Each study's terms of reference should include it. The risk of failure during system operation is increased if risk analyses do not address operational conditions, and if the information available to analysts does not include them the omission should be documented with the analysis results. If this cannot be done, an exception report should be prepared.

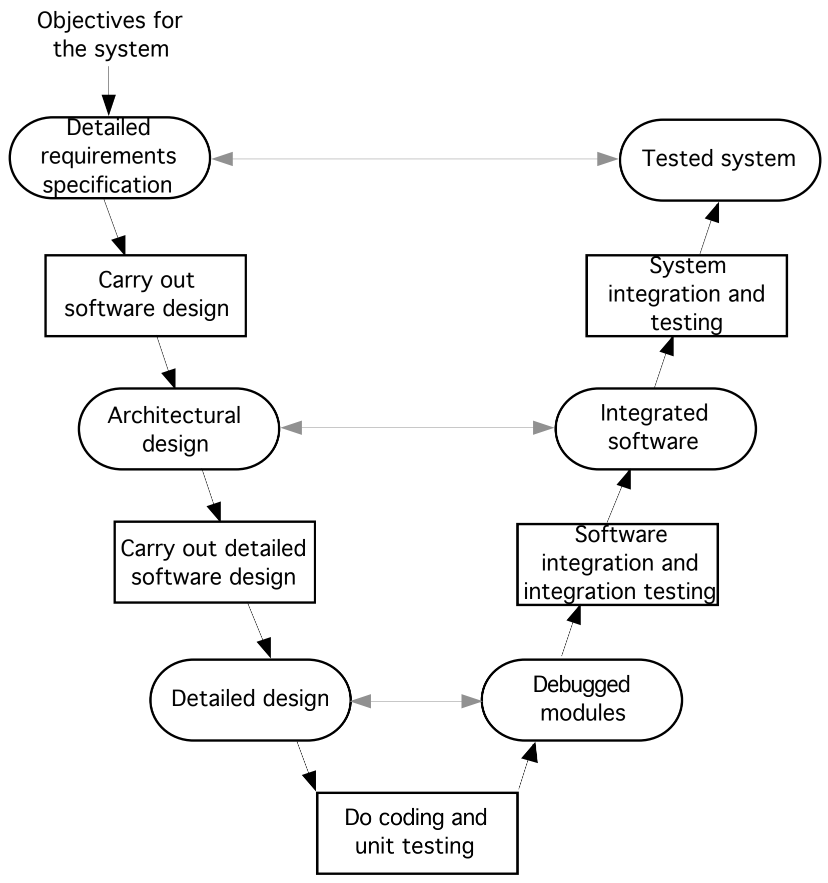

Objectives for the system
Detailed requirements specification
Tested system
Carry out software design
System integration and testing
Architectural design
Integrated software
Carry out detailed software design
Software integration and integration testing
Detailed design
Debugged modules
Do coding and unit testing

Figure 4: The V Model, which takes a testing perspective

4 Addressing risk during development

4.1 When the system's objectives are defined

In its representation of the development life cycle, the V Model begins with project initiation and the preparation of a detailed specification of requirements. But prior to that the objectives for the intended system should have been defined at a strategic business level. Only objectives that address system functionality and other technical issues are within the scope of this paper; those that concern issues such as increases in profit and reductions in staff numbers are not.

When objectives are defined, there is usually not a great deal of information about the proposed system, so an elaborate risk analysis would be inappropriate. But an analysis of what is intended can be of considerable value. For example, an analysis of the safety risks posed by an intended chemical factory could determine where it is – and, importantly, is not – appropriate to site it. It should also reveal the major hazards posed by the system and, thus, inform the developers of the system specification of the need for particular safety requirements. Thus, risks recognised and accepted at the objectives stage must be mitigated by requirements at the specification stage.

Suppose that the management of a hospital wishes to introduce a computerised patient-records system. They might state their objectives as to:

- Store the clinical and administrative records of all patients;
- Allow physicians quick access to find out the medical histories of patients and also to update the records with diagnoses, prescriptions, and progress reports;
- Allow nurses access to discover what treatment is necessary and plan dosing;
- Allow management to keep track of all patient transactions, including the details of their entry, where they were accommodated, and what medical and other services they received.

Even a brief risk analysis would almost certainly reveal that loss of medical records would be catastrophic, that doctors would not like a slow system, that nurses should not have access to full patient records, and that if management lost track of patients the hospital would not be paid for services provided. The first and fourth of these demands that records must be highly available and that their integrity must be preserved. The second requires high performance and training of the doctors. The third calls for confidentiality and the facility for different users to have access only to certain functions. So out of knowledge of the major risks arises a number of high-level requirements, including security, which was not stated as a system objective. These requirements then need to be analysed in order that the full specification should contain detailed, quantified statements of what is needed. Further, they provide notice to testers of items that should receive particular attention, both in system testing and in the customer's acceptance testing.

It may be seen that the objectives, as stated, are to provide benefits from the points of view of a number of system stakeholders, among whom the patients do not feature. Yet, the patient is at the heart of the system. If the patient's perspective is taken, it turns out that a breach of security could lead to a threat to the patient's safety – which was not obvious from the initial statements. Thus, a risk analysis that addresses this point will lead to the definition of requirements to provide patient protection.

A further advantage of carrying out a risk analysis at the objectives stage is that it facilitates the early detection of objectives that are undefined or unclear, and of conflicts between multiple objectives. Such conflicts arise, for example, when each of several senior stakeholders defines one or more objectives for the system. It is not uncommon for projects to come to grief in later stages because of the impossibility of achieving all objectives within the allocated time and budget, or even at all. From a test-planning point of view, the risks identified at this stage should be used in planning business-case tests at the system and acceptance testing stages.

4.2 At the specification stage

There is a great deal more information in a requirements specification than in a statement of objectives, and this needs to be structured and validated before being subjected to risk analysis. An attempt to analyse a document that is ambiguous, in which there are gaps, and where related items of information are randomly scattered, is likely to show only that the biggest risk arises from the uncertainties of the specification itself. But given a good specification, risk analysis is both possible and useful.

The first things to look for is whether the risks arising out of the objectives have been properly catered for in the specification, and whether documentation has been produced to guide the test planners on thoroughly testing the critical requirements. For analysing the full specification, the HAZOP technique [Redmill et al 1999] is preferred, mainly because of its guidewords, but failure modes and effects analysis (FMEA) [IEC 1985] is also an option. The HAZOP guidewords focus attention on what the consequence would be if specified functions were absent, if they were incorrect, or if they operated at an inappropriate time. Similarly, they raise questions about so-called 'non-functional' requirements such as performance. When incorrect or inadequate functionality is deemed probable, or would give rise to severe consequences, particular attention should be paid to those functions during design, and notice should be given to the test planners.

Specification risks that are usually not recognised are those attached to requirements that do not meet any of the system's objectives. All requirements should be strategically validated to ensure that they fall within, and contribute to, the system's objectives. Those that do not should not be allowed. Many projects run over-time and over-budget, and many systems fail to meet their objectives, because of requirements that should not be there. Further, many such systems carry unused functionality, which costs a great deal of wasted money to produce. If the extra requirements are important, the objectives for the system should be extended. If they are not important, they should be removed from the specification. But identifying such extraneous requirements requires an understanding of the system's objectives, and often these are not known by the project manager. Thus, risk analysis at the specification stage is likely to lead to a re-examination of the system's objectives, which, in turn, may result in a review of the requirements.

The requirements that are found in the analysis to be critical or risky should be given special attention. One possible remedy is to remove them, but this requires a review of the system's objectives, for it may mean that they too have to be adjusted. If it is found that a risky requirement is there because of a key objective, another solution must be found. In this case, the risk should be mitigated in the design, and testing should be planned to exercise the design solution so as to ensure that it is as robust as necessary.

4.3 At the design stage

A design is even more appropriate to a HAZOP study than is a requirements specification, for almost by definition it is well structured. As already suggested, two studies are appropriate – of the architectural design and the detailed design (see Figure 4). At the architectural design stage, high-level design decisions, such as the choice of technologies and ways in which reliability and availability will be achieved, should be made. The architectural design should also address the risks identified at the requirements stage. For example, design features should be used to mitigate risks to safety and security. A risk analysis of the architectural design should therefore not only address the design with respect to whether and how it meets the system requirements, but also whether and how it mitigates the risks previously identified.

If the design proposes the use of a new technology, or one that has been untried in the organisation, the risk analysis should seek to identify the risks of doing so and analyse them to determine their potential effects on the organisation, the project, and the system. Lessons should be derived from other uses of the technology – in other projects and other organisations. Was it easy to use? How much training was required? Did it work first time? If the risks turn out to be significant, what management action is appropriate? Should the notion of employing the technology be abandoned? Or would training of staff be sufficient to reduce the risks to tolerable levels? Perhaps, even after training, an expert should be brought in as a short-term consultant to ensure that the technology is used both efficiently and effectively.

The detailed design defines how individual functions will be implemented in software. The content of software modules, and their inter-relationships, are revealed. Thus, the functions identified at earlier stages as being critical, or posing high risks, can be given particular attention. They can be allocated to the best programmers and made the focus of more intensive testing.

The detailed design, based on top-down decomposition, also provides test planners with a basis for deriving an integration plan. This should define the preferred schedule for building the system from the various software components. Usually, if such a plan is created at all, only the testers know about it. Yet, if component integration and integration testing are to be carried out in a given order, the modules to be integrated must be delivered by the developers in that order. So the integration plan needs to be the basis not only for integration and testing but also for development. Otherwise, it is almost certain to be useless as a plan. Development in accordance with the integration plan reduces the need for stubs and drivers, saves time, and makes testing more effective. It also paves the way for the use of business cases in system and acceptance testing. Further, it stops testers from being at the mercy of the programmers and places them on an equal footing with them. Indeed, the test planners need to have influence over the coding process – which is unheard of in most organisations. A risk analysis at the detailed design stage therefore needs to consider the risks not only of software modules being delivered late, but also of their not being delivered according to the integration plan.

Identifying such risks throws up new problems, for their causes often lie not in technical problems but in issues of project and development management, planning and discipline, and these may not be familiar to the risk analysts. Thus, there is the lesson that if risk analysis is to be carried out during the development life cycle it must be expected that analysis of the identified hazards will require exploration of project, administration, and management issues as well as technical problems. Analysts must therefore be chosen, and trained, accordingly.

Once a risk analysis of the detailed design has been carried out, the identified risks should be used by the development manager in selecting appropriate staff for programming, appropriate techniques for inspecting and testing the modules, and appropriate levels of quality assurance of the processes involved. Test planning too should be informed by the results.

5 Discussion

Testing helps to reduce the probability of failure by detecting deficient software. But it would be preferable for a low probability to be achieved in the first place, during the development processes. This is not a new concept, but it is not usual for testing concepts or risk analysis to be employed in addressing the testers' backward risks. This paper proposes ways of doing so.

Taking a risk-based approach during development allows risks to be identified and mitigated early, and tracked throughout development, thus resulting in fewer defects in the software. It also provides risk-based information to test planners so that they can focus testing on the most critical functions and their software. The paper therefore takes a testing perspective. It introduces the subject of risk analysis, provides a four-staged model of carrying it out, and shows how and at what stages it may be used during system development.

No risk analysis should ever be considered to be definitive. Always the results could be improved if more information were available. Each analysis during the development life cycle will have access to greater detail than its predecessor, and the techniques employed and the time and other resources allocated should reflect this. However, each analysis should be carried out with sufficient thoroughness to identify and analyse the major risks identifiable at that stage. Then, the resulting risk-management requirements must either be met by immediate changes or they must be defined as constraints on the future definition and development of the system. Thus, the risks identified at the objectives stage should be mitigated either by changing the system's objectives or by defining appropriate requirements, which then become a part of the system's requirements specification. The need to mitigate risks identified at the requirements stage should be met by changing the requirements (which may also require changes to the system's objectives) or by placing constraints on the system's design; and so on. Clearly, one essential purpose of each risk analysis after the first should be to enquire into whether adequate mitigation of previously identified risks has been put in place. The results of each analysis should be used to inform the relevant level of test planning, thus ensuring that testing is used to verify whether or not the defined risk mitigations have indeed been put in place and are adequate. Doing this may require simulation of the risk conditions to determine if the relevant undesirable events occur and, if they do, how they are coped with.

Risks arising during development may have their origins in risky objectives or requirements for the system, or in high-level strategic decisions. They may have technical causes, such as the use of unproven or unfamiliar technologies. They may also result from administrative, managerial, or project management problems or deficiencies. Thus, the risk analyses must be carried out by teams whose members, between them, are capable of understanding all of these issues. Many risk analyses are abortive because the analysts are blind to the types of hazards involved.

Development-process improvement is usually done from a project-management or quality-management perspective. Applying risk management and taking a test planner's perspective are not common and they offer possibilities for both academic research and trials by industrial practitioners.

References

[HSE 2001]. Health and Safety Executive: *Reducing risks, protecting people.* HSE Books, 2001

[IEC 1985]. *International Standard 812 Analysis Techniques for System Reliability: Procedures for Failure Modes and Effect Analysis*. International Electrotechnical Commission, Geneva, 1985

[Redmill 2004]. Redmill F: Exploring risk-based testing and its implications. *Software Testing, Verification and Reliability* 14, 3-15, 2004

[Redmill et al 1999] Redmill F, Chudleigh M and Catmur J: *System Safety: HAZOP and Software HAZOP*. John Wiley & Sons, 1999

[STARTS 1986]. *The STARTS Purchasers' Handbook.* NCC Publications, Manchester, 1986