A Lightweight Technique for Assessing Risks in Requirements Analysis

Kenneth Boness[a], Anthony Finkelstein[b] and Rachel Harrison[c]

[a]School of Systems Engineering, University of Reading, Berks, UK, k.d.boness@reading.ac.uk

[b]Department of Computer Science, UCL, Gower Street, London, WC1E 6BT, UK, a.finkelstein@cs.ucl.ac.uk

[c]Stratton Edge Consulting, Glos., GL7 2LS, UK, Rachel.Harrison@strattonedge.com

Abstract: This paper describes a simple and practical technique for assessing the risks, that is, the potential for error, and consequent loss, in software system development, acquired during a requirements engineering phase. The technique uses a goal-based requirements analysis as a framework to identify and rate a set of key issues in order to arrive at estimates of the feasibility and adequacy of the requirements. We illustrate the technique and show how it has been applied to a real systems development project. We show how problems in this project could have been identified earlier, thereby avoiding costly additional work and unhappy users.

1 Introduction

It is a common experience, regardless of what process is adopted, that requirements gathering is stalled by a premature leap to design and coding. This leap is compelled by project managers, and, strangely, customers who have a poor understanding of the consequences of neglecting requirements. Requirements processes sometimes look like a handy place to save time on projects that must be completed quickly.

Experienced developers usually attempt to prevent this from happening, and to do this they advance the established arguments that faults found late in the development process are exponentially more costly to fix than those found earlier. They are able to use published data [4] to support this argument. Such general and rather abstract arguments are however difficult to sustain in an industrial setting, particularly when some requirements work has already been undertaken and the question boils down to the adequacy of that work [17]. It must also be acknowledged that schedules do matter. A less than adequate system delivered quickly may, on some occasions, be better than a fault-free system delivered late. If external factors have resulted in slippage, the time does need to be made up somewhere.

What we describe below is a simple, and we believe practical, technique for assessing risks in requirements analysis. It provides a framework within which developers and managers can identify occurrences of those factors known to be associated with subsequent development problems and by this means reach a balanced and informed assessment of risk.

The result of applying the approach is a Risk Profile that combines project data and expert estimates. The approach can be used very early in a project life cycle. If used correctly the profile can indicate where effort in requirements analysis should be directed to eliminate issues giving rise to latent high-severity problems. The benefit of the approach is that the risk assessment is based on the current state of the requirements analysis and is directly related to the project and its particular risks.

We begin by discussing the framework for our technique. Section 3 presents the rationale and section 4 gives details of the technique, including the risk factors and an example for illustration. Section 5 discusses how we tested the technique and section 6 discusses future work. Finally section 7 presents related work on metrics for risk assessment.

2 Framework

Our technique uses, as an underlying framework, a requirements goal-graph (see Figure 1). A goal graph represents stakeholders' hopes for a system-to-be, which will operate in an expected environment, in fulfilment of a contract. The graph can represent the rationale and understanding of the problem to be solved along with a set of domain assumptions and the stated requirements for a system. Goal graphs of this type are widely used in goal-based requirements engineering and most notably in the KAOS approach [15] which informs our work. In this paper we use the word 'requirement' to refer to all goals in the goal graph, not just leaf goals.

A requirements goal graph is usually composed of a number of root goals, the motivating goals, with a hierarchy of sub-goals connected by refinement relations. Sub-goals may satisfy the super-ordinate goal in conjunction (and) or as alternative realisations (or). The hierarchy has the potential to be cross-cutting although this is beyond the scope of this paper, as is a detailed treatment of risks associated with requirements traceability. Leaf goals have no refinements but can be expressed in operational terms and can be assigned a method by which the goal may be satisfied (that is, implemented); this may be performed by a system component (in which case the leaf is an operationalised requirement) or by the system environment (in which case the leaf is an operationalised assumption).

We will not argue here as to why such an approach is, in fact, a good basis for requirements engineering, though we believe it to be so. It is not required that a project using our technique adopts a goal-based approach as its sole or even principal method of gathering and organising requirements. The technique does however require a 'best effort' goal-graph to be constructed representing the decomposition and refinement of the requirements at the point at which a risk assessment is to be made. If an approach other than a goal-based approach is used, for instance a conventional functionally driven natural language specification, such a goal graph can be derived from it. The better the quality of the requirements work, the easier this is to do. If, of course, a goal-oriented approach has been used the first part of our approach comes for free.

3 Rationale

There are two primary concerns in requirements engineering; to ensure that what is set down is what is ‘actually’ wanted (the *adequacy* of the requirements) and to ensure that this is achievable (the *feasibility* of the requirements). If the requirements prove subsequently to *not* represent the needs of users or prove impossible or too costly to implement then there is a good chance that this will prove a serious problem.

The adequacy of the requirements can only be established indirectly. Firstly by assessing the extent to which the key stakeholders (customers, clients, analysts, designers, developers, managers, sales representatives etc.) have mandated the root goals. Secondly by estimating confidence in the refinement of the goals, that is that the sub-goals preserve the intentions expressed in the top-level goals (following a method such as KAOS may help to ensure this). The feasibility of the requirements as a working basis for subsequent development can also only be established indirectly, firstly by ensuring that all the leaf goals are in fact operationalised, and secondly by ensuring that each of these requirements or assumptions has been checked against project or domain constraints to ensure that they could reasonably be achieved within project resources or that it is reasonable to expect the system environment to satisfy them.

4 Technique

Our technique uses a goal-graph as an armature to record expert judgements which rate the goal graph against a set of factors that are associated with risk. The *cost* and *value* of the goals is also assessed in order to produce an overall profile. The risks of goals which have not been explicitly assessed by experts are calculated using the metrics associated with their parents or children.

We assume here that the risk analysis is performed by a requirements engineer for the benefit of a development manager working closely with customers on bespoke fixed price contracts. Systems that have been delivered but require re-development can also be accommodated by re-analysing the evolving subsystem.

In summary, this technique identifies and rates a set of key issues by providing a minimal set of independent subjective metrics using information from stakeholders or expert opinion. As such it represents a meta-level assessment technique that assesses *the information that is known* about the requirements rather than assessing the requirements themselves by (say) using a formal language representation of the requirements. As the metrics are subjective they can be obtained very early on in the project life cycle, and provide information that would not otherwise be made explicit.

4.1 Risk Factors

From our previous experience we identified four independent risk factors: (1) the environmental assumptions, (2) the achievability of the implementation of the requirements, (3) the integrity of the refinements and (4) the stakeholders' mandate. These are described in more detail below.

RF 1. **Environment:** leaf goals assigned to the environment for satisfaction despite inadequate grounds for believing this is a reliable assignment.

RF 2. **Achievability**: leaf goals assigned to the software for satisfaction despite inadequate grounds for believing an acceptable implementation is achievable.

RF 3. **Refinement:** refinements where the stated refinement is open to question (due to semantic entailment), goals with no justifiable parents or where there is an uncertainty about the degree to which a goal contributes to a root goal, possibly due to errors with the semantic entailment or to 'gold-plating'.

RF 4. **Stakeholders' mandate**: goals where stakeholder endorsement is uncertain, in other words where the stakeholder(s) agreement with the goal is in doubt.

Independent work has referred to these factors as *categories* in the context of project risk [14, 21, 22]. Table 1 lists the risk factors together with the names and descriptions of their raw data. We use the convention that raw data is shown capitalized. The metrics are subjective; appropriate expert assessors (who are often project stakeholders, but may also be managers and requirements engineers who are independent of the project) are asked to express their confidence that the risks have been addressed. Guidelines are given to the assessors to help to ensure uniform interpretation of the risk factors. Since the assessment is subjective relevant information can be obtained very early on in the project life cycle. An in-depth analysis into the reasons for the different values given by the assessors can also be very revealing. Table 2 describes the calculation of the metrics in more detail. These metrics are in fact measures of the probability of *success*: if a risk has a probability p of occurring, our metrics measure (1-p).

The Risk Factors RF1 (ENV) and RF2 (ACHIEVE) are estimated for the leaf goals and then calculated from these estimates for the remaining goals under consideration. Table 2 shows how to do this and how to average over the assessors' estimates (a more detailed data analysis would include consideration of medians and outliers). Note that the probabilities for RF1 and RF2 are multiplied to obtain the root goals' values as this is how probabilities accumulate. Any design alternatives (indicated by an 'Or' in the goal graph) are removed prior to calculation of the metrics. This may be done by the production of additional alternative goal graphs. Admittedly this could lead to a combinatorial explosion of goal graphs. However, we assume that incremental evaluation is being performed, leading to a reduction in the number of 'Ors' as the project proceeds.

The Risk Factors RF3 (SOUND) and RF4 (MANDATE) are estimated for each individual goal. The estimates for RF4 for child goals are considered when estimating a parent's RF4. Conflicting scores for RF4 should stimulate further analysis and negotiation. However, this is outside the scope of the paper: this technique is intended for analyzing the stakeholders' concerns, not for managing them.

Clearly it would be possible to decompose the four risk factors into more fine-grained factors. However we prefer to proceed with the four factors shown above in order to keep the technique as simple and practical as possible for automation. For a top-level summary it may also be necessary to produce a total for the whole goal graph. Table 10 in the Appendix shows how this can be done by averaging over the project goals.

We treat any obstacles in the goal graph as surrogate root goals. This has the advantage of keeping the technique simple and is intuitively obvious for stakeholders.

Note that our technique takes no account of the possible relationships between different goals other than child to parent, and assumes that the leaf goals are independent (i.e. a child node should not be shared by 2 or more higher level goals). Nor does it attempt to rank the risk factors in any way or to weight assessors' judgements. These matters are the subject of our on-going research.

A manager will want to know whether the necessary work is feasible and whether the project will deliver an adequate result. The former (Feasibility, F) can be answered by considering the metrics RF1 and RF2 together and the latter (Adequacy, A) by considering RF3 and RF4 together. As the four risk factors are all probabilities we calculate Feasibility and Adequacy for the leaf nodes by multiplying the appropriate factors together (see Table 3). The formula for Adequacy uses the product of the RF3 metric for all goals from the root goal to the leaf goal in question. This is in order to estimate confidence about all the relevant refinements.

A manager will also be interested in the Proportional Value (PValue) and Proportional Cost (PCost) of the goals, where Value is the business value of a goal and Cost is the development cost of an operationalised requirement relative to the other operationalised requirements. This paper assumes that we are concerned with projects that are currently under development rather than re-development. Value and Cost are converted into PValue and PCost, so that systems with different numbers of goals can be compared easily. This is done by dividing by the total Value or Cost (respectively) of the operationalised requirements. Operationalised requirements are used because it is these that are implemented. PValue and PCost are expressed as percentages (see Table 4). For this Risk Profile experts are asked to assign a relative Value to all the root goals (i.e. the value is relative to the other root goals' values). The Values for goals that are not explicitly given by experts are taken from the Values of the immediate parent goals such that the sum of the child goals is equal to the parents' Values. Clearly there are other ways to distribute Value throughout the graph, but we chose this technique for simplicity. As noted earlier, design alternatives are removed prior to calculation of the metrics so that the computations can be performed in a straightforward and inexpensive manner. Once Costs and Values have been assigned to the leaf goals they are then normalised such that the total for each sums to 100%.

The costs of environmental assumptions are taken to be zero as such costs are regarded here as outside project constraints.

We could also calculate the *priority* of the requirements. Two possible exemplar metrics are given in Table 4. The first, Priority1, is defined as the value of a goal divided by its proportional cost, the second, Priority2, as the value of the goal multiplied by the complement of its proportional cost. These are both obviously simplifications of metrics for prioritization that allow valuable goals that are cheap to be given the go-ahead in favour of valuable expensive goals. This allows requirements to be ranked in terms of priority for development. Also the values of the Risk Factors for each goal could be weighted by the goal's priority in order to produce an overall Risk Profile. A manager may have a number of different priority metrics taking value for money, cost and other factors into account [2, 12, 13, 10]. We prefer not to prescribe a particular prioritisation scheme here but rather to leave it to individual choice, depending on the particular circumstances.

4.2 Example: calculating body mass index

By way of demonstration, we present a small but typical problem involving the calculation of body mass index.

From the requirements definition in Figure 2 the mandated goals for the new software are as follows:-

1) Normal operation of the walk-on scales must be maintained.

2) The scales are for use in public places.

3) WeighCom's good reputation must be maintained.

4) The scales are to be constructed from prescribed components.

We will use the first goal to illustrate our technique. The first step is to annotate the goal graph [6] with the estimated values for the risk factors. This is shown in Figure 3, which extends the model from the tool Objectiver [7] to indicate operationalised requirements or assumptions in ovals and metric annotations in italics. For this example we produced the expert opinion by adopting the role of the product manager for WeighCom. In different situations however it may be more appropriate to approach other assessors for opinions, such as developers when estimating project costs, and customers when estimating the stakeholders' mandate. Clearly the technique benefits from automation when a larger number of assessors is involved.

The four risk factors in Table 1 are used to measure the probability that the project will succeed in each of the four areas (Environment, Achievability, Refinement and Stakeholders' Mandate) for our Risk Profile. Figure 3 also shows an obstacle. An obstacle is something that may prevent a goal from being achieved. Obstacles are not assessed in our technique, but goals that overcome them are.

The Risk Factors were obtained using expert judgement and the formulae in Tables 2 and 4. The scores for ENV and ACHIEVE were collected for the leaves as appropriate. They were then converted to RF1 and RF2 scores for the leaves and then propagated through the graph to the roots as indicated in Table 2. Scores for SOUND were taken for all goals and then RF3 was calculated for each goal from its SOUND score and those of all its descendent goals using the formula in Table 2. RF4 scores were obtained from the MANDATE scores for each goal (if provided) according to the formula in Table 2. There is no propagation of MANDATE scores through the graph. Next, COST was judged for all leaf requirements and then propagated to the roots using the formulae in Table 4. Finally the scores for VALUE were collected for the roots (and the goals that overcome obstacles); these were propagated to the leaf goals using the formulae in Table 4.

There are numerous ways to represent the Risk Profile. Table 5 shows how to present the Profile on a per-goal basis calculated using the formulae in Tables 2 and 4 together with the raw values of the risk factors taken from the annotated goal graph. There may be a number of these tables or spreadsheets for each project, depending on the chosen representation and number of goals.

The metrics Feasibility (F) and Adequacy (A) were calculated using the formulae in Table 3 and are shown in Table 6 together with proportional costs and proportional values. We calculate the metrics for all nodes in the graph because we might want to slice through the graph at any point to obtain an assessment of the goal graph at a number of different levels, depending on the stage of the analysis, the required level of detail, the criticality of the assessment, etc. Only the risk factors, costs and values are included in Figure 3 to improve its readability.

Figures 4 and 5 show the plots of Feasibility against Adequacy for the leaf goal costs and values respectively. The charts show the percentages of the Cost or Value for leaf goals with particular Feasibility and Adequacy scores. The charts can also include the names of the goals, made available automatically through a spreadsheet, for ease of identification of problematic goals. The numbers in the figures may include colour coded advice depending on pre-assigned risk thresholds set by a domain expert (such as red for *do not proceed*, amber for *proceed with caution*, green for *proceed*). The placement of the regions is subjective. Indeed, a manager can vary the sizes of the regions, depending on the business case and the priorities for the project. If the information about the requirements is not adequate (i.e. the mandate is lacking, or the refinements are not sound), but the requirements are feasible, then a manager would know to get more information about the mandate and re-examine the refinements. If the requirements are not feasible, but the mandate exists and the refinements are sound (i.e. the requirements are adequate) then again, the project could be re-planned. The areas of concern depend on the manager's priorities for each particular system, and so will differ from one system to the next.

Looking at Figure 4 it can be seen (for example) that 50% of the leaf goals costs fall into the *Proceed with caution* region, as do 13% of the leaf goal values in Figure 5. A high number of leaf goals in the *Do not proceed* region would indicate that further requirements work was warranted. For an alternative representation we can average over the goals as shown in Table 10 in the Appendix and produce one table of aggregated metrics for the entire project.

Figure 4 and Figure 5 show that luckily none of the goals in this example fell into the *Do not proceed* region, although some did fall in the *Proceed with caution* region mainly due to a problems with the project's adequacy. Even though this is a small example it was useful to check that our technique is viable in that it is not too time-consuming for the experts. This example also showed that the technique is straightforward to apply, that the calculations can be automated and that the results agree with intuitive assessments.

5 Applying the technique

In order to test the proposed technique, we began by retrospectively applying the metrics to a medium-sized project provided by one of University College London's Administrative Divisions. We chose to study the University Personal Identifier or UPI upgrade project as it had the necessary requirements data as well as accompanying historical documentary evidence which we could use to check the results. One of the authors, the project manager and project supervisor acted as expert assessors and reviewed the project approximately 18 months after project initiation using the PRINCE2 project initialisation document, business case and initial project plan to obtain the risk factor metrics. At this point the project was complete, which suited us as we needed to be able to check our findings against what actually happened. Also the project manager and supervisor were both independent of our research and had no vested interest in the results.

The College had been using the UPI system to control access to services over the College intranet. This UPI system links information held in a series of primary person systems (PPS) and a series of secondary person systems (SPS). The former include registration and human resource systems. The latter include library systems, buildings access etc. and were in a state of flux; some of the systems were stable, some were unstable and new ones were being developed. The existing UPI system was deemed unsatisfactory because:-

1. Its design was ad hoc and maintaining it depended upon an understanding held by a small and decreasing number of staff.
2. It had function and performance shortcomings.
3. Extending it to meet future needs was considered to be unduly expensive.

The College wished to remedy these problems by introducing a new, replacement UPI system.

The goal graph, containing 52 goals, was reverse engineered and then inspected by the project manager. After correction it was adopted as the best effort goal graph. Every leaf requirement was operationalised with an associated responsibility. The project manager then answered a questionnaire requesting metrics for the risk factors (RF1, RF2, RF3 and RF4) as well as Value and Cost as appropriate for root goals, leaf goals and sub-goals. The data obtained was then used as input for our metrics toolkit.

It took the project manager and supervisor under 2 hours in total to judge the risk factors, costs and values for the goals, and this was reported as easy and intuitive to do after about 30 minutes of instruction. The assessments (which were in agreement) were not judged as time consuming compared to creating the best effort goal graph. It took about an hour to add costs and values to the goal graph as they were already partially documented. The assessment of the refinement argument (RF3) was the most difficult for the assessors to appreciate but with some guidance became straightforward.

All the goal refinements except for two were confirmed by the project manager and supervisor as correct and so were given a score for RF3 of 1. The goal graph was mainly shallow (with a maximum height of 6 and average height of 4). Figures 6 and 7 represent the results of the automated application of our technique to the whole goal graph. Note that only the leaf goals are shown but from them we can mine the metrics for every associated goal.

In figure 6 we found that 47% of the goals by value are in the *Do not proceed* region whereas only 26% of the goals are in the *Proceed* region. Also, in figure 7 we found that 47% of the expected cost is in the *Do not Proceed* region whereas none of the cost is in the *Proceed* region. The entries marked as 0% represent assumptions on the environment. They may have costs but they are not counted using our technique.

Table 7 shows the leaf goals from the *Do not proceed* region together with their proportional costs and proportional values, and their Feasibility (F) and Adequacy (A) metrics. Note that the Goal numbers are simply used as identifiers: they do not indicate the position of a goal in the graph. Some of the goals in table 7 violate simple principles of good requirements statements, and it could be argued that this should have raised concerns. However, whilst detecting problems in this way is clearly commendable, such a technique cannot be fully automated and does not scale.

Table 8 shows the leaves from the *Proceed* region**.** None of them are operationalised in the system as they are all assumptions. The results show the high confidence in the assumptions underpinning the project (Feasibility, F).

In the event all the goals in Table 8 were implemented successfully but goal 61 became problematic following implementation for various non-technical reasons.

Table 9 shows how our lightweight assessment technique for the root goals compares with their actual outcomes. The 'Threat' ratio *(n/m)* indicates that out of the *m* leaf goals that contribute to a higher-level goal, *n* of them fall in the *Do Not Proceed* category. The ratio is of interest because partial implementation of the goal graph may be possible. The PValue is the proportional value of these goals. The Outcome column indicates whether or not the goal was achieved at the end of the project. The 'Pain' column indicates how difficult it was to complete the goal (the need for extra investment, extra management etc. indicates higher 'pain'). The colour coding shows whether our technique produced an assessment that was correct, 'Ok' with a verbal explanation, too pessimistic, or open to question.

Goal 25 has Goal 3 as its child. Our technique labelled Goal 3 as a *proceed* goal; in fact Goal 25 was very effective and saved a certain amount of staff effort when cleaning the data. Goal 34 was highly threatened, mostly due to lack of feasibility rather than its adequacy. Goal 38 was partially compromised and partially sound. This was resolved but required extra investment. Goal 84 was difficult to achieve due to politics and was set aside pending a different solution.

This project had 52 goals; the development team was small but served a large range of stakeholders. In this application domain we found that typically a project would have between 30 and 100 goals. It is straightforward to provide spreadsheet questionnaires for the assessment, and as the calculations are automated and data entry can also easily be automated one way to approach the assessment would be to provide automated steps through the refinement in an inspection-like manner.

In conclusion, Figures 6 and 7 show that the assessors had low confidence in a dangerously high proportion of the project (47% in both Figures 6 and 7). This alone shows that the project is at risk. Of course the analysis is subjective. Nevertheless the expert assessors agreed that the relative positioning of the goals in the two graphs is reasonable. Further, the assessors could change the subjective values and perform *what-if* scenario testing to produce best, worst and average case analyses. The project was initially expected to last 6 months but was re-scheduled after 1 year, then sought re-budgeting and still required significant re-planning on two further occasions. At the outset the team were optimistic about technical possibilities but feasibility was always a concern. In fact the biggest risks arose from the lack of realism concerning availability of staff and the lack of support for some crucial assumptions in the maintenance of the primary person and secondary person systems. The goals ‘support for external persons’ (67) and ‘easing workload’ (84) were uncertain from the start and were substantially jettisoned.

The nine goals in Table 9 represent 100% of the value assigned to the project. From the table we can see that 80% of our assessment was good (shown by the regular text, italic text and bold text) within which 43% (in regular text) was very good. Unfortunately the assessment for goal 72 (the bold italic text) was questionable as the end result was reported as being "painful". On investigation of the goal graph and expert judgement of feasibility we conclude the judgements were not themselves accurate. They may have been affected by a lack of shared understanding of the system.

During our analysis it became clear that a number of problems stemmed from a serious lack of shared understanding between the stakeholders at the time of project sanction. If our technique been used it is very likely that the project would not have started until greater confidence in the requirements had been achieved and that this would have brought about better shared understanding.

6 Future Work

Our technique depends upon goal analysis. When a project is founded on a goal analysis our technique is straightforward and natural to use. However in the real world of software development the situation is usually more complex. Urgency may tempt developers to use only ad-hoc methods of requirements analysis and project planning. In such cases, as mentioned earlier, a best effort graph, with an approximate sketch of the currently known requirements, needs to be created. As this will require extra resources it is crucial that this graph can be created economically and quickly in order that the momentum of the project is not lost.

This principle is being put to the test in a large venture capital project which is under development using agile processes. The Scrum process [19] was adopted in this project following two years of traditional process modelling. As the system was under development it was uneconomical to perform a retrospective goal analysis of the work done to date. Instead a goal graph was constructed in step with each short development phase, or *sprint*. Thus over time a comprehensive graph emerges and (more importantly) a goal graph becomes available for each sprint. These relatively small goal graphs are then used when evaluating confidence about the risk of failure of the sprint. The goal analysis creates requirements for the specific sprint whereas the work of previous sprints becomes a part of the environment.

We are also applying the technique to a venture capital supported project to develop a product that will solve problems of storing huge amounts of data. This had seen circa 20 person years of development activity and was based on IPR originally developed in government laboratories but had not led to a product. Early in 2006 a decision was taken to develop the product using the Scrum methodology. This allowed a build up of concrete functionality whilst evolving the required functionality through a series of sprints. We are in the early stages of applying our technique to this project whereby the value, adequacy and feasibility of each sprint is to be assessed as part of the gateway to the sprint backlog prior to starting the sprint.

This application to real industrial Scrum cycles has benefited not only the risk evaluation but has also been of considerable assistance to the test team in preparing sprint acceptance tests as it facilitates development of test cases for each requirement at requirements time. The goal analysis technique is surprisingly agile when used in this manner. A more mainstream development process which we believe our work will match well is the spiral development methodology [3] since it hinges on identifying risks and dealing with them early on.

This partitioning of an existing system into environment assumptions and new software with each sprint is akin to developing in iterations and increments as often happens when working with legacy systems. Our industrial experience shows that this situation is increasingly prevalent; indeed, the Scrum method in general is becoming increasingly popular with companies wishing to adopt an ‘agile’ approach. Thus following our initial experiences of applying our technique to agile projects we are encouraged to apply the technique to legacy systems as well as to new developments.

As mentioned earlier, our approach assumes that the requirements can be expressed in such a way that all leaf goals are independent. It also assumes that the requirements can be represented in a hierarchical decomposition. This assumption is somewhat restrictive and it potentially inhibits the treatment of some cross-cutting requirements such as performance and security [1, 16, 18]. In order to address such concerns and provide a generalized solution we are investigating the use of problem frames [9, 11] to structure and encapsulate subsystems and so provide interfaces that facilitate a formal decomposition and re-composition of subsystems.

The use of COTS components would clearly impact the project costings. In this case the metrics could be used to aid decision making when comparing different system’s architectures. We intend to validate our technique through a number of case studies. Clearly the technique may be most applicable to particular application domains that are amenable to subjective analysis, and our future work will investigate the applicability of the technique and the composition of subsystems with nonfunctional requirements further.

7 Related Work

A process and tool called DDP (Defect Detection and Prevention) have been developed at JPL to facilitate risk management over the entire project life cycle [5]. DDP is intended for use in the aerospace industry. It uses a large set of risk elements, weighted requirements, pre-set formulae and information from experts. It is similar to our technique, but the process uses trees of requirements (rather than goal graphs), and trees showing why these requirements may not be achieved. The user can then select preventative measures, analyses, process controls and tests in order to minimize the residual risk. The process begins at the architecture stage, whereas our technique is intended for use during requirements analysis. There is software tool support for the DDP process, which weights risk elements by their impact on weighted requirements. The weights are obtained from the mission success criteria. The DDP process uses a large number of metrics and pre-determined industry-specific knowledge. This contrasts with our technique, which we have deliberately tried to keep as simple and lightweight as possible.

Karlsson, Wohlin and Regnell evaluated six different methods for prioritizing software requirements experimentally [13]. The six methods ranged from the analytic hierarchy process, AHP, through binary search trees to priority groups and found AHP to be the most promising approach. An overview of requirements prioritization [2] explains and critiques a range of methods (and their combination) for prioritization of factors such as importance, cost, risk, volatility etc. An example is given which uses stakeholders' opinions to rank the importance of a set of requirements. This work on prioritization does not examine how the risk assessment is arrived at. In contrast, our work defines requirements risk for individual requirements more precisely and provides a method that integrates with KAOS.

The use of expert opinion in software development in general is not a new idea: Freimut, Briand and Vollei [8] give a thorough account to justify its use during evaluation of the cost-effectiveness of inspections. They suggest that expert opinion may be needed if information regarding a phenomenon cannot be collected by any other affordable means or if information is not being collected within the timeframe that it is needed. They point out that expert data is subject to bias, uncertainty, and incompleteness but say that these problems can be prevented and controlled by means of carefully performed elicitation of expert estimates.

Ruhe et al. have presented an approach to decision support in requirements negotiation called "Quantitative WinWin" [20], which incorporates quantitative methods with Boehm's original WinWin approach. The approach uses the Analytical Hierarchy Process for a stepwise determination of the stakeholders' preferences in quantitative terms. These results are combined with methods for early effort estimation to evaluate the feasibility of alternative requirements subsets in terms of their related implementation effort. This work is particularly concerned with finding those subsets of requirements that can be implemented without exceeding a given maximum effort whereas our work is concerned with assessing the potential for error in systems development.

8 Conclusions

In this paper we have presented an easy-to-use technique for assessing risk in requirements analysis using goal graphs and judgements supplied by either stakeholders or experts. Using this technique, potentially risky projects can be detected at an early stage so that decisions can be taken about different courses of action.

This research is the first report of a risk assessment technique using high-level subjective metrics collected during requirements analysis using goal-graphs. Note that the technique can be used with assessments provided either by stakeholders or by experts. It will thus provide a meta-level risk assessment that could be very valuable to project managers and senior managers. We have shown how the information can be combined with data concerning the value and cost of goals. The technique has been empirically tested using a number of projects, the results from one of which has been described in detail. These tests have shown that we can indeed discover the parts of a project that are most likely to lead to problems using this technique. Our experience suggests that the technique is entirely sympathetic with the real world needs of industrial software development. We have also partially automated this technique by implementing a lightweight tool called KAOS Lite for goal sketching and automated data collection.

9 Acknowledgments

The authors would like to thank their collaborators for providing project data, particularly Kathryn Lewis of MSD, and the anonymous referees who reviewed this paper and provided very helpful comments. We would also like to thank Claes Wohlin and Barbara Kitchenham for their input to early versions of this paper.

10 References

[1] Baniassad, E., Clements, P.C., Araújo, J., Moreira, A., Rashid, A., Tekinerdogan, B.: “Discovering Early Aspects,” IEEE Software, 2006, 23, (1), pp. 61-70.

[2] Bernader, P., Andrews, A.: “Requirements Prioritization” in Aurum, A. Wohlin, C. (Eds.): ‘Engineering and Managing Software Requirements’, (Springer, 2006), pp. 69–94.

[3] Boehm, B.W.: "A Spiral Model of Software Development and Enhancement," Computer, 1988, 21, (5), pp. 61-72.

[4] Boehm, B.W.: "Software Risk Management: Principles and Practices", IEEE Software, 1991, 8, (1), pp. 32-41.

[5] Cornford, S.L., Feather M.S., and Hicks, K.A.: “DDP - A tool for life-cycle risk management,” Proceedings of the IEEE Aerospace Conference, Big Sky, Montana, 2001, pp. 441-451.

[6] Dardenne, A., Fickas, S., and Lamsweerde, A. van: "Goal-directed concept acquisition in requirements elicitation", Proc. 6th IEEE Workshop System Specification and Design, Como, Italy, 1991, pp. 14-21.

[7] Darimont, R.: “Requirements Engineering with Objectiver: from Goal Analysis to Automatically Derived Requirements Documents,” RE’03 Exhibitors’ track, International Conference on Requirements Engineering (RE03), Los Alamitos, California: IEEE Computer Society Press, 2003.

[8] Freimut, B., Briand, L.C., and Vollei, F.: “Determining Inspection Cost-Effectiveness by Combining Project Data and Expert Opinion,” IEEE Transactions on Software Engineering, 2005, 31, (12), pp 1074-1092.

[9] Hall, J.G., Jackson, M., Laney, R.C., Nuseibeh, B., Rapanotti, L.: “Relating Software Requirements and Architectures using Problem Frames,” Tenth International IEEE Conference on Requirements Engineering, Los Alamitos, California: IEEE Computer Society Press, 2002, pp. 137-144.

[10] IEEE Std. 1233-1996, “IEEE Guide for Developing System Requirements Specifications”, IEEE June 1996.

[11] Jackson, M.: Problem Frames: Analysing and Structuring Software Development Problems, Addison Wesley, 2000.

[12] Karlsson, J., Ryan, K.: “A cost-value approach for prioritizing requirements,” IEEE Software, 1997, 14, (5), pp. 67-74.

[13] Karlsson , J., Wohlin, C., Regnell, B.: “An evaluation of methods for prioritizing software requirements,” Journal of Information and Software Technology, 1998, 39, (14-15), pp. 939-947.

[14] Keil, M., Cule, P.E., Lyytinen, K., Schmidt, R.C. “A Framework for Identifying Software Project Risks.” Communications of the ACM, 1998, 41, (11), pp. 76-83.

[15] Lamsweerde, A. van, Dardenne, A., Delcourt, B. Dubisy, F.: “The KAOS Project: Knowledge acquisition in automated specification of software.” Proceedings of the AAAI Spring Symposium Series, Stanford University, AAAI, March 1991, pp. 59-62.

[16] Mylopoulos, J., Chung, L., Nixon, B.: “Representing and using nonfunctional requirements: a process oriented approach,” IEEE Transactions on Software Engineering, 1992, 18, (6), pp. 483-497.

[17] Pinheiro, F.A.C.: “Requirements Honesty” International Workshop on Time-Constrained Requirements Engineering (TCRE'02), Germany Sep. 2002.

[18] Rashid, A., Sawyer, P., Moreira, A., Araujo, J.: “Early aspects: a model for aspect-oriented requirements engineering.” Requirements Engineering 2002 (RE’02), Germany 2002, pp. 199-202.

[19] Rising L., and Janoff, N.: “The Scrum Software Development Process for Small Teams”, IEEE Software, 2000, 17, (4), pp. 26-32.

[20] Ruhe, G., Eberlein, A., and Pfahl, D.: “Quantitative WinWin: a new method for decision support in requirements negotiation.”, Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering (SEKE '02), Ischia, Italy, July 2002, pp. 159-166.

[21] Wallace, L., Keil, M.: “Software project risks and their effect on outcomes.” Communications of the ACM, 2004, 47, (4), pp. 68-73.

[22] Wallace, L., Keil, M., Rai. A.: “How Software Project Risk Affects Project Performance: An Investigation of the Dimensions of Risk and an Exploratory Model,” Decision Sciences, 35, (2), March 2004, pp. 289-321.

APPENDIX

TABLE 10

Calculations of the aggregated metrics

Risk Factor	Metric Aggregation: total for project
RF1	$\frac{1}{G}\sum_{g=1}^{G}\text{Env}(g)$ where G = total no. of goals
RF2	$\frac{1}{G}\sum_{g=1}^{G}\text{Req}(g)$ where G = total no. of goals
RF3	$\frac{1}{G}\sum_{g=1}^{G}\text{Refine}(g)$ where G = total no. of goals
RF4	$\frac{1}{G}\sum_{g=1}^{G}\text{Mandate}(g)$ where G = total no. of goals
Average value	$\frac{1}{G}\sum_{g=1}^{G}\text{Value}(g)$ where G = total no. of goals
Total cost	The estimated total cost of a project p is the sum of the estimated costs of its operationalised requirements: $\text{TOTALCOST}(p) = \sum_{r=1}^{R}\text{Cost}(r)$ where R = total no. of operationalised requirements.

TABLE 1

The 4 risk factors and associated metrics

Risk Factor	Raw data	Description
RF1: Environment	ENV	Probability that the assumptions can be satisfied by the environment
RF2: Achievability	ACHIEVE	Probability that the requirements are achievable
RF3: Refinement	SOUND	Probability that the refinements are sound
RF4: Mandate	MANDATE	Probability that the requirements are mandated

TABLE 2

Calculations of the metrics for the Risk Factors

Risk Factor	Definition of metric for each leaf and goal
RF1	The RF1 metric for an operationalised assumption *a* is calculated as the average of the assessors scores: RF1(a) = $\frac{1}{N}\sum_{n=1}^{N}\mathrm{ENV(n)}$ where N = total no. of assessors, ENV is the probability that an assumption can be fulfilled within project and domain constraints, estimated for leaves by assessors. A goal's RF1 is calculated as the product of its children's RF1 values. If there are no operationalised assumptions for a goal then RF1 = 1
RF2	The RF2 metric for an operationalised requirement *r* is calculated as the average of the assessors scores: RF2(r) = $\frac{1}{N}\sum_{n=1}^{N}\mathrm{ACHIEVE(n)}$ where N = total no. of assessors, ACHIEVE(n) is the probability that a requirement can be achieved within project and domain constraints, estimated for leaves by assessors. A goal's RF2 is calculated as the product of its children's RF2 values. If there are no operationalised requirements for a goal then RF2 = 1
RF3	The RF3 metric for a goal *g* is calculated as the average of the assessors scores: RF3(g) = $\frac{1}{N}\sum_{n=1}^{N}\mathrm{SOUND\ (n)}$ where N = total no. of assessors, SOUND(n) is the probability that the refinement argument for a goal is sound, estimated by assessors. For leaf goals this equates to estimating whether further refinement is needed. The default RF3 metric for goals is 0.5
RF4	The RF4 metric for a goal *g* is calculated as the average of the assessors scores: RF4(g) = $\frac{1}{N}\sum_{n=1}^{N}\mathrm{MANDATE(n)}$ where N = total no. of assessors, MANDATE is the probability that the stakeholders' mandate for a goal is sound, estimated by assessors. The default RF4 metric for goals is 0.5

TABLE 3

Calculations of the indirect metrics

Definition of metric for each leaf goal
Feasibility F(g) = RF1(g)*RF2(g)
Adequacy $A(g) = RF4(g) * \prod_{i=1}^{L} RF3(g_i)$
where L = number of levels from the root goal g_1 to the leaf goal g_L (and where $g = g_L$)

TABLE 4

Calculations of the Cost, Value and Priority metrics

Definition of metric for each goal
Value(g) = $\frac{1}{N}\sum_{n=1}^{N}\text{VALUE(n)}$ where N = total no. of assessors, VALUE is the business value of a goal as rated by assessors (its contribution to the project) on an ordinal scale greater than or equal to 0.   For a goal not rated by assessors Value is inherited from its immediate parent(s) s.t. the children's Values sum to the parent's Value. A goal with 2 or more parents will inherit the sum of its parents' Values.
The TOTALVALUE of a project is the sum of the values of its operationalised requirements: TOTALVALUE (p) = $\sum_{r=1}^{R}\text{Value (r)}$ and R = total no. of operationalised requirements.   The proportional value of a leaf goal g, PValue (g), is the value of the goal divided by the total value of the project p expressed as a percentage: PValue (g) = (Value (g)/TOTALVALUE (p))*100, TOTALVALUE(p) $\neq$ 0   PValue (g) = 0 otherwise   PValues for the leaf goals are normalised such that the total sums to 100%.
The cost of an operationalised requirement r,   Cost(r) = $\frac{1}{N}\sum_{n=1}^{N}\text{COST(n)}$ where N = total no. of assessors, COST is the cost of the operationalised requirement (either in FPs, staff days or LOC) as estimated by assessors.
The cost of a goal is the sum of the costs of all its operationalised requirements: Cost(g) = $\sum_{r=1}^{R'}\text{COST(r)}$ where R' = total no. of operationalised requirements for a goal. The cost of an operationalised assumption is zero.
The TOTALCOST of a project is the sum of the costs of its operationalised requirements: TOTALCOST (p) = $\sum_{r=1}^{R}\text{Cost (r)}$

and R = total no. of operationalised requirements in the graph.

The proportional cost of a leaf goal g, PCost (g), is the cost of the goal divided by the total cost of the project p expressed as a percentage:

$PCost(g) = (Cost(g)/TOTALCOST(p))*100, TOTALCOST(p) \neq 0$

PCost (g) = 0 otherwise

PCosts for the leaf goals are normalised such that the total sums to 100%.

$$Priority1(g) = \frac{PValue(g)}{PCost(g)} *100$$

if $PCost(g) \neq 0$

Priority1(g) = 100% otherwise

where N = total no. of assessors.

$$Priority2(g) = PValue(g) * (1 - PCost(g)) *100$$

if $PCost(g) \neq 100\%$,

Priority2(g) = 100% otherwise

where N = total no. of assessors

TABLE 5

Raw data for WeighCom, root goal *‘Maintain walk-on scales normal operation’*

Name	**RF1**	**RF2**	**RF3**	**RF4**	**Cost**	**Value**
G1	**1**	**1**	**1**	**1**	**0**	17
G2	1	0.64	**0.9**	**1**	100	**100**
G3	**1**	**0.8**	**0.7**	**1**	**50**	18
G4	**1**	**1**	**1**	**1**	**0**	17
G5	1	0.8	**0.7**	**1**	50	35
G6	**1**	**0.8**	**1**	**0.9**	**30**	18
G7	1	0.8	**0.8**	**0.6**	30	35
G8	**0.8**	1	**1**	**1**	**0**	40
G9	0.8	1	**0.9**	**0.9**	0	**40**
G10	**1**	**1**	**0.9**	**1**	**20**	30

Key: bold entries are expert judgements.

TABLE 6

Feasibility and Adequacy metrics for WeighCom leaf goals for the root goal *'Maintain walk-on scales normal operation'* together with proportional cost and proportional values.

Name	**Feasibility**	**Adequacy**	**PCost**	**PValue**
G1	1	0.63	0%	12%
G3	0.8	0.44	50%	13%
G4	1	0.72	0%	12%
G6	0.8	0.65	30%	13%
G8	0.8	0.9	0%	29%
G10	1	0.81	20%	21%
			100%	100%

TABLE 7

Leaves from the *Do not proceed* quadrant

Goal	Name	PValue	PCost	Feasibility	Adequacy
63	Visiting staff and contractors are not allocated a UPI and are thus excluded	0.4%	0.0%	0.00	0.16
67	Allow external persons access (details TBD)	10.8%	8.9%	0.00	0.40
77	Access to each SS services is gated to identify the appropriate UPI which is then maintained throughout the ensuing session	2.1%	4.4%	0.20	0.40
81	For each non-connecting SS the most recent UPI-status pairs are persisted for reference.	4.7%	4.4%	0.30	0.48
90	User initiated extraction of data from PPSs in form suitable for use in producing College directory service	1.7%	4.4%	0.30	0.36
92	A user initiated mechanism providing college directory, "all staff" and telephone directory information from PPSs	1.7%	2.2%	0.50	0.36
94	Office process improvements	3.5%	0.0%	0.50	0.28
101	Real-time API returning status for any submitted UPI	4.7%	4.4%	0.40	0.48
102	Each connecting SS is supported to use real-time status API	4.7%	4.4%	0.20	0.40
103	A scheduled batch-mode service provides updates of all known UPI-Status to each non-connecting SS	2.3%	4.4%	0.50	0.48
104	Every non-connecting SS cooperates with the batch-mode service	2.3%	4.4%	0.30	0.48
122	Data extracted from SSs and PPSs for person centric reports is accurate.	3.5%	4.4%	0.20	0.36
123	The PPs and SSs support the necessary querying based on UPI to satisfy statutory obligations on personal data access	3.1%	0.0%	0.50	0.50
128	Each SS acts on the UPI linked Status to provide the right degree of service to the individual.	2.1%	2.2%	0.30	0.56

TABLE 8

Leaves from the *Proceed* region

Goal	Name	PValue	PCost	Feasibility	Adequacy
3	Maintain Comprehensive technical and user documentation.	9.4%	0.0%	0.80	0.70
4	Re-engineering of the UPI will enable future modification at low cost.	9.4%	0.0%	0.60	0.70
61	The status and data about staff is correctly maintained on the HR system	0.4%	0.0%	0.80	0.72
62	The status and data about students is correctly maintained on the REG system	0.4%	0.0%	0.80	0.72
82	Staff operating the primary systems maintain current status for all UPI's pertinent to each primary system	1.6%	0.0%	0.70	0.80
86	Only the PPSs hold personal data to be matched to a UPI	2.0%	0.0%	0.60	0.80
124	The existence of the UPI allows adequate facility to querying across PPSs and SSs.	3.1%	0.0%	0.70	1.00

TABLE 9

Assessments vs. Outcomes for Root Goals

Goal	Description	Threat	PValue	Outcome	'Pain'
20	Every UPI is maintained with a current status	Low (2/7)	6%	Full	Low
74	Every eligible individual is assigned a UPI	None (0/8)	6%	Full	Low
56	Every individual's access rights to any secondary system is based on their UPI linked status	High (7/7)	6%	Partial	Medium
25	To enhance maintainability compared to the original UPI which depends upon vulnerable knowledge	None (0/2)	19%	Full	Low
31	*Optimally determine external person's eligibility to use College facilities.*	*High (1/1)*	*10%*	*Removed*	*Low*
34	**Operate in a "real time" as opposed to "batch" mode**	**High (5/5)**	**17%**	**Partial**	**Medium**
38	To comply with statutory obligations on personal data.	Medium (1/2)	6%	Partial	Low
72	***To improve the efficiency of daily UPI matching compared with the original UPI***	***None (0/2)***	***19%***	***Full***	***High***
84	*To ease the workload of College Staff*	*High (4/4)*	*10%*	*Uncertain*	*High*

Key: (n/m): *n* leaves of *m* belonging to the goal are in the *Do Not Procee*d zone.

Regular text: Assessment Correct

Italic text: Assessment Ok

Bold text: Assessment Pessimistic

Bold italic text: Assessment Questionable

A Lightweight Technique for Assessing Risks in Requirements Analysis

Kenneth Boness[a], Anthony Finkelstein[b] and Rachel Harrison[c]

[a]School of Systems Engineering, University of Reading, Berks, UK, k.d.boness@reading.ac.uk

[b]Department of Computer Science, UCL, Gower Street, London, WC1E 6BT, UK, a.finkelstein@cs.ucl.ac.uk

[c]Stratton Edge Consulting, Glos., GL7 2LS, UK, Rachel.Harrison@strattonedge.com

Abstract: This paper describes a simple and practical technique for assessing the risks, that is, the potential for error, and consequent loss, in software system development, acquired during a requirements engineering phase. The technique uses a goal-based requirements analysis as a framework to identify and rate a set of key issues in order to arrive at estimates of the feasibility and adequacy of the requirements. We illustrate the technique and show how it has been applied to a real systems development project. We show how problems in this project could have been identified earlier, thereby avoiding costly additional work and unhappy users.

1 Introduction

It is a common experience, regardless of what process is adopted, that requirements gathering is stalled by a premature leap to design and coding. This leap is compelled by project managers, and, strangely, customers who have a poor understanding of the consequences of neglecting requirements. Requirements processes sometimes look like a handy place to save time on projects that must be completed quickly.

Experienced developers usually attempt to prevent this from happening, and to do this they advance the established arguments that faults found late in the development process are exponentially more costly to fix than those found earlier. They are able to use published data [4] to support this argument. Such general and rather abstract arguments are however difficult to sustain in an industrial setting, particularly when some requirements work has already been undertaken and the question boils down to the adequacy of that work [17]. It must also be acknowledged that schedules do matter. A less than adequate system delivered quickly may, on some occasions, be better than a fault-free system delivered late. If external factors have resulted in slippage, the time does need to be made up somewhere.

What we describe below is a simple, and we believe practical, technique for assessing risks in requirements analysis. It provides a framework within which developers and managers can identify occurrences of those factors known to be associated with subsequent development problems and by this means reach a balanced and informed assessment of risk.

The result of applying the approach is a Risk Profile that combines project data and expert estimates. The approach can be used very early in a project life cycle. If used correctly the profile can indicate where effort in requirements analysis should be directed to eliminate issues giving rise to latent high-severity problems. The benefit of the approach is that the risk assessment is based on the current state of the requirements analysis and is directly related to the project and its particular risks.

We begin by discussing the framework for our technique. Section 3 presents the rationale and section 4 gives details of the technique, including the risk factors and an example for illustration. Section 5 discusses how we tested the technique and section 6 discusses future work. Finally section 7 presents related work on metrics for risk assessment.

2 Framework

Our technique uses, as an underlying framework, a requirements goal-graph (see Figure 1). A goal graph represents stakeholders' hopes for a system-to-be, which will operate in an expected environment, in fulfilment of a contract. The graph can represent the rationale and understanding of the problem to be solved along with a set of domain assumptions and the stated requirements for a system. Goal graphs of this type are widely used in goal-based requirements engineering and most notably in the KAOS approach [15] which informs our work. In this paper we use the word 'requirement' to refer to allgoal s in the goal graph, not just leaf goals.

A requirements goal graph is usually composed of a number of root goals, the motivating goals, with a hierarchy of sub-goals connected by refinement relations. Sub-goals may satisfy the super-ordinate goal in conjunction (and) or as alternative realisations (or). The hierarchy has the potential to be cross-cutting although this is beyond the scope of this paper, as is a detailed treatment of risks associated with requirements traceability. Leaf goals have no refinements but can be expressed in operational terms and can be assigned a method by which the goal may be satisfied (that is, implemented); this may be performed by a system component (in which case the leaf is an operationalised requirement) or by the system environment (in which case the leaf is an operationalised assumption).

We will not argue here as to why such an approach is, in fact, a good basis for requirements engineering, though we believe it to be so. It is not required that a project using our technique adopts a goal-based approach as its sole or even principal method of gathering and organising requirements. The technique does however require a 'best effort' goal-graph to be constructed representing the decomposition and refinement of the requirements at the point at which a risk assessment is to be made. If an approach other than a goal-based approach is used, for instance a conventional functionally driven natural language specification, such a goal graph can be derived from it. The better the quality of the requirements work, the easier this is to do. If, of course, a goal-oriented approach has been used the first part of our approach comes for free.

3 Rationale

There are two primary concerns in requirements engineering; to ensure that what is set down is what is ‘actually’ wanted (the *adequacy* of the requirements) and to ensure that this is achievable (the *feasibility* of the requirements). If the requirements prove subsequently to *not* represent the needs of users or prove impossible or too costly to implement then there is a good chance that this will prove a serious problem.

The adequacy of the requirements can only be established indirectly. Firstly by assessing the extent to which the key stakeholders (customers, clients, analysts, designers, developers, managers, sales representatives etc.) have mandated the root goals. Secondly by estimating confidence in the refinement of the goals, that is that the sub-goals preserve the intentions expressed in the top-level goals (following a method such as KAOS may help to ensure this). The feasibility of the requirements as a working basis for subsequent development can also only be established indirectly, firstly by ensuring that all the leaf goals are in fact operationalised, and secondly by ensuring that each of these requirements or assumptions has been checked against project or domain constraints to ensure that they could reasonably be achieved within project resources or that it is reasonable to expect the system environment to satisfy them.

4 Technique

Our technique uses a goal-graph as an armature to record expert judgements which rate the goal graph against a set of factors that are associated with risk. The *cost* and *value* of the goals is also assessed in order to produce an overall profile. The risks of goals which have not been explicitly assessed by experts are calculated using the metrics associated with their parents or children.

We assume here that the risk analysis is performed by a requirements engineer for the benefit of a development manager working closely with customers on bespoke fixed price contracts. Systems that have been delivered but require re-development can also be accommodated by re-analysing the evolving subsystem.

In summary, this technique identifies and rates a set of key issues by providing a minimal set of independent subjective metrics using information from stakeholders or expert opinion. As such it represents a meta-level assessment technique that assesses *the information that is known* about the requirements rather than assessing the requirements themselves by (say) using a formal language representation of the requirements. As the metrics are subjective they can be obtained very early on in the project life cycle, and provide information that would not otherwise be made explicit.

4.1 Risk Factors

From our previous experience we identified four independent risk factors: (1) the environmental assumptions, (2) the achievability of the implementation of the requirements, (3) the integrity of the refinements and (4) the stakeholders' mandate. These are described in more detail below.

RF 1. **Environment:** leaf goals assigned to the environment for satisfaction despite inadequate grounds for believing this is a reliable assignment.

RF 2. **Achievability**: leaf goals assigned to the software for satisfaction despite inadequate grounds for believing an acceptable implementation is achievable.

RF 3. **Refinement:** refinements where the stated refinement is open to question (due to semantic entailment), goals with no justifiable parents or where there is an uncertainty about the degree to which a goal contributes to a root goal, possibly due to errors with the semantic entailment or to 'gold-plating'.

RF 4. **Stakeholders' mandate**: goals where stakeholder endorsement is uncertain, in other words where the stakeholder(s) agreement with the goal is in doubt.

Independent work has referred to these factors as *categories* in the context of project risk [14, 21, 22]. Table 1 lists the risk factors together with the names and descriptions of their raw data. We use the convention that raw data is shown capitalized. The metrics are subjective; appropriate expert assessors (who are often project stakeholders, but may also be managers and requirements engineers who are independent of the project) are asked to express their confidence that the risks have been addressed. Guidelines are given to the assessors to help to ensure uniform interpretation of the risk factors. Since the assessment is subjective relevant information can be obtained very early on in the project life cycle. An in-depth analysis into the reasons for the different values given by the assessors can also be very revealing. Table 2 describes the calculation of the metrics in more detail. These metrics are in fact measures of the probability of *success*: if a risk has a probability p of occurring, our metrics measure (1-p).

The Risk Factors RF1 (ENV) and RF2 (ACHIEVE) are estimated for the leaf goals and then calculated from these estimates for the remaining goals under consideration. Table 2 shows how to do this and how to average over the assessors' estimates (a more detailed data analysis would include consideration of medians and outliers). Note that the probabilities for RF1 and RF2 are multiplied to obtain the root goals' values as this is how probabilities accumulate. Any design alternatives (indicated by an 'Or' in the goal graph) are removed prior to calculation of the metrics. This may be done by the production of additional alternative goal graphs. Admittedly this could lead to a combinatorial explosion of goal graphs. However, we assume that incremental evaluation is being performed, leading to a reduction in the number of 'Ors' as the project proceeds.

The Risk Factors RF3 (SOUND) and RF4 (MANDATE) are estimated for each individual goal. The estimates for RF4 for child goals are considered when estimating a parent's RF4. Conflicting scores for RF4 should stimulate further analysis and negotiation. However, this is outside the scope of the paper: this technique is intended for analyzing the stakeholders' concerns, not for managing them.

Clearly it would be possible to decompose the four risk factors into more fine-grained factors. However we prefer to proceed with the four factors shown above in order to keep the technique as simple and practical as possible for automation. For a top-level summary it may also be necessary to produce a total for the whole goal graph. Table 10 in the Appendix shows how this can be done by averaging over the project goals.

We treat any obstacles in the goal graph as surrogate root goals. This has the advantage of keeping the technique simple and is intuitively obvious for stakeholders.

Note that our technique takes no account of the possible relationships between different goals other than child to parent, and assumes that the leaf goals are independent (i.e. a child node should not be shared by 2 or more higher level goals). Nor does it attempt to rank the risk factors in any way or to weight assessors' judgements. These matters are the subject of our on-going research.

A manager will want to know whether the necessary work is feasible and whether the project will deliver an adequate result. The former (Feasibility, F) can be answered by considering the metrics RF1 and RF2 together and the latter (Adequacy, A) by considering RF3 and RF4 together. As the four risk factors are all probabilities we calculate Feasibility and Adequacy for the leaf nodes by multiplying the appropriate factors together (see Table 3). The formula for Adequacy uses the product of the RF3 metric for all goals from the root goal to the leaf goal in question. This is in order to estimate confidence about all the relevant refinements.

A manager will also be interested in the Proportional Value (PValue) and Proportional Cost (PCost) of the goals, where Value is the business value of a goal and Cost is the development cost of an operationalised requirement relative to the other operationalised requirements. This paper assumes that we are concerned with projects that are currently under development rather than re-development. Value and Cost are converted into PValue and PCost, so that systems with different numbers of goals can be compared easily. This is done by dividing by the total Value or Cost (respectively) of the operationalised requirements. Operationalised requirements are used because it is these that are implemented. PValue and PCost are expressed as percentages (see Table 4). For this Risk Profile experts are asked to assign a relative Value to all the root goals (i.e. the value is relative to the other root goals' values). The Values for goals that are not explicitly given by experts are taken from the Values of the immediate parent goals such that the sum of the child goals is equal to the parents' Values. Clearly there are other ways to distribute Value throughout the graph, but we chose this technique for simplicity. As noted earlier, design alternatives are removed prior to calculation of the metrics so that the computations can be performed in a straightforward and inexpensive manner. Once Costs and Values have been assigned to the leaf goals they are then normalised such that the total for each sums to 100%.

The costs of environmental assumptions are taken to be zero as such costs are regarded here as outside project constraints.

We could also calculate the *priority* of the requirements. Two possible exemplar metrics are given in Table 4. The first, Priority1, is defined as the value of a goal divided by its proportional cost, the second, Priority2, as the value of the goal multiplied by the complement of its proportional cost. These are both obviously simplifications of metrics for prioritization that allow valuable goals that are cheap to be given the go-ahead in favour of valuable expensive goals. This allows requirements to be ranked in terms of priority for development. Also the values of the Risk Factors for each goal could be weighted by the goal's priority in order to produce an overall Risk Profile. A manager may have a number of different priority metrics taking value for money, cost and other factors into account [2, 12, 13, 10]. We prefer not to prescribe a particular prioritisation scheme here but rather to leave it to individual choice, depending on the particular circumstances.

4.2 Example: calculating body mass index

By way of demonstration, we present a small but typical problem involving the calculation of body mass index.

From the requirements definition in Figure 2 the mandated goals for the new software are as follows:-

1) Normal operation of the walk-on scales must be maintained.

2) The scales are for use in public places.

3) WeighCom's good reputation must be maintained.

4) The scales are to be constructed from prescribed components.

We will use the first goal to illustrate our technique. The first step is to annotate the goal graph [6] with the estimated values for the risk factors. This is shown in Figure 3, which extends the model from the tool Objectiver [7] to indicate operationalised requirements or assumptions in ovals and metric annotations in italics. For this example we produced the expert opinion by adopting the role of the product manager for WeighCom. In different situations however it may be more appropriate to approach other assessors for opinions, such as developers when estimating project costs, and customers when estimating the stakeholders' mandate. Clearly the technique benefits from automation when a larger number of assessors is involved.

The four risk factors in Table 1 are used to measure the probability that the project will succeed in each of the four areas (Environment, Achievability, Refinement and Stakeholders' Mandate) for our Risk Profile. Figure 3 also shows an obstacle. An obstacle is something that may prevent a goal from being achieved. Obstacles are not assessed in our technique, but goals that overcome them are.

The Risk Factors were obtained using expert judgement and the formulae in Tables 2 and 4. The scores for ENV and ACHIEVE were collected for the leaves as appropriate. They were then converted to RF1 and RF2 scores for the leaves and then propagated through the graph to the roots as indicated in Table 2. Scores for SOUND were taken for all goals and then RF3 was calculated for each goal from its SOUND score and those of all its descendent goals using the formula in Table 2. RF4 scores were obtained from the MANDATE scores for each goal (if provided) according to the formula in Table 2. There is no propagation of MANDATE scores through the graph. Next, COST was judged for all leaf requirements and then propagated to the roots using the formulae in Table 4. Finally the scores for VALUE were collected for the roots (and the goals that overcome obstacles); these were propagated to the leaf goals using the formulae in Table 4.

There are numerous ways to represent the Risk Profile. Table 5 shows how to present the Profile on a per-goal basis calculated using the formulae in Tables 2 and 4 together with the raw values of the risk factors taken from the annotated goal graph. There may be a number of these tables or spreadsheets for each project, depending on the chosen representation and number of goals.

The metrics Feasibility (F) and Adequacy (A) were calculated using the formulae in Table 3 and are shown in Table 6 together with proportional costs and proportional values. We calculate the metrics for all nodes in the graph because we might want to slice through the graph at any point to obtain an assessment of the goal graph at a number of different levels, depending on the stage of the analysis, the required level of detail, the criticality of the assessment, etc. Only the risk factors, costs and values are included in Figure 3 to improve its readability.

Figures 4 and 5 show the plots of Feasibility against Adequacy for the leaf goal costs and values respectively. The charts show the percentages of the Cost or Value for leaf goals with particular Feasibility and Adequacy scores. The charts can also include the names of the goals, made available automatically through a spreadsheet, for ease of identification of problematic goals. The numbers in the figures may include colour coded advice depending on pre-assigned risk thresholds set by a domain expert (such as red for *do not proceed*, amber for *proceed with caution*, green for *proceed*). The placement of the regions is subjective. Indeed, a manager can vary the sizes of the regions, depending on the business case and the priorities for the project. If the information about the requirements is not adequate (i.e. the mandate is lacking, or the refinements are not sound), but the requirements are feasible, then a manager would know to get more information about the mandate and re-examine the refinements. If the requirements are not feasible, but the mandate exists and the refinements are sound (i.e. the requirements are adequate) then again, the project could be re-planned. The areas of concern depend on the manager's priorities for each particular system, and so will differ from one system to the next.

Looking at Figure 4 it can be seen (for example) that 50% of the leaf goals costs fall into the *Proceed with caution* region, as do 13% of the leaf goal values in Figure 5. A high number of leaf goals in the *Do not proceed* region would indicate that further requirements work was warranted. For an alternative representation we can average over the goals as shown in Table 10 in the Appendix and produce one table of aggregated metrics for the entire project.

Figure 4 and Figure 5 show that luckily none of the goals in this example fell into the *Do not proceed* region, although some did fall in the *Proceed with caution* region mainly due to a problems with the project's adequacy. Even though this is a small example it was useful to check that our technique is viable in that it is not too time-consuming for the experts. This example also showed that the technique is straightforward to apply, that the calculations can be automated and that the results agree with intuitive assessments.

5 Applying the technique

In order to test the proposed technique, we began by retrospectively applying the metrics to a medium-sized project provided by one of University College London's Administrative Divisions. We chose to study the University Personal Identifier or UPI upgrade project as it had the necessary requirements data as well as accompanying historical documentary evidence which we could use to check the results. One of the authors, the project manager and project supervisor acted as expert assessors and reviewed the project approximately 18 months after project initiation using the PRINCE2 project initialisation document, business case and initial project plan to obtain the risk factor metrics. At this point the project was complete, which suited us as we needed to be able to check our findings against what actually happened. Also the project manager and supervisor were both independent of our research and had no vested interest in the results.

The College had been using the UPI system to control access to services over the College intranet. This UPI system links information held in a series of primary person systems (PPS) and a series of secondary person systems (SPS). The former include registration and human resource systems. The latter include library systems, buildings access etc. and were in a state of flux; some of the systems were stable, some were unstable and new ones were being developed. The existing UPI system was deemed unsatisfactory because:-

1. Its design was ad hoc and maintaining it depended upon an understanding held by a small and decreasing number of staff.
2. It had function and performance shortcomings.
3. Extending it to meet future needs was considered to be unduly expensive.

The College wished to remedy these problems by introducing a new, replacement UPI system.

The goal graph, containing 52 goals, was reverse engineered and then inspected by the project manager. After correction it was adopted as the best effort goal graph. Every leaf requirement was operationalised with an associated responsibility. The project manager then answered a questionnaire requesting metrics for the risk factors (RF1, RF2, RF3 and RF4) as well as Value and Cost as appropriate for root goals, leaf goals and sub-goals. The data obtained was then used as input for our metrics toolkit.

It took the project manager and supervisor under 2 hours in total to judge the risk factors, costs and values for the goals, and this was reported as easy and intuitive to do after about 30 minutes of instruction. The assessments (which were in agreement) were not judged as time consuming compared to creating the best effort goal graph. It took about an hour to add costs and values to the goal graph as they were already partially documented. The assessment of the refinement argument (RF3) was the most difficult for the assessors to appreciate but with some guidance became straightforward.

All the goal refinements except for two were confirmed by the project manager and supervisor as correct and so were given a score for RF3 of 1. The goal graph was mainly shallow (with a maximum height of 6 and average height of 4). Figures 6 and 7 represent the results of the automated application of our technique to the whole goal graph. Note that only the leaf goals are shown but from them we can mine the metrics for every associated goal.

In figure 6 we found that 47% of the goals by value are in the *Do not proceed* region whereas only 26% of the goals are in the *Proceed* region. Also, in figure 7 we found that 47% of the expected cost is in the *Do not Proceed* region whereas none of the cost is in the *Proceed* region. The entries marked as 0% represent assumptions on the environment. They may have costs but they are not counted using our technique.

Table 7 shows the leaf goals from the *Do not proceed* region together with their proportional costs and proportional values, and their Feasibility (F) and Adequacy (A) metrics. Note that the Goal numbers are simply used as identifiers: they do not indicate the position of a goal in the graph. Some of the goals in table 7 violate simple principles of good requirements statements, and it could be argued that this should have raised concerns. However, whilst detecting problems in this way is clearly commendable, such a technique cannot be fully automated and does not scale.

Table 8 shows the leaves from the *Proceed* region**.** None of them are operationalised in the system as they are all assumptions. The results show the high confidence in the assumptions underpinning the project (Feasibility, F).

In the event all the goals in Table 8 were implemented successfully but goal 61 became problematic following implementation for various non-technical reasons.

Table 9 shows how our lightweight assessment technique for the root goals compares with their actual outcomes. The 'Threat' ratio *(n/m)* indicates that out of the *m* leaf goals that contribute to a higher-level goal, *n* of them fall in the *Do Not Proceed* category. The ratio is of interest because partial implementation of the goal graph may be possible. The PValue is the proportional value of these goals. The Outcome column indicates whether or not the goal was achieved at the end of the project. The 'Pain' column indicates how difficult it was to complete the goal (the need for extra investment, extra management etc. indicates higher 'pain'). The colour coding shows whether our technique produced an assessment that was correct, 'Ok' with a verbal explanation, too pessimistic, or open to question.

Goal 25 has Goal 3 as its child. Our technique labelled Goal 3 as a *proceed* goal; in fact Goal 25 was very effective and saved a certain amount of staff effort when cleaning the data. Goal 34 was highly threatened, mostly due to lack of feasibility rather than its adequacy. Goal 38 was partially compromised and partially sound. This was resolved but required extra investment. Goal 84 was difficult to achieve due to politics and was set aside pending a different solution.

This project had 52 goals; the development team was small but served a large range of stakeholders. In this application domain we found that typically a project wouldhave between 30 and 100 goals. It is straightforward to provide spreadsheet questionnaires for the assessment, and as the calculations are automated and data entry can also easily be automated one way to approach the assessment would be to provide automated steps through the refinement in an inspection-like manner.

In conclusion, Figures 6 and 7 show that the assessors had low confidence in a dangerously high proportion of the project (47% in both Figures 6 and 7). This alone shows that the project is at risk. Of course the analysis is subjective. Nevertheless the expert assessors agreed that the relative positioning of the goals in the two graphs is reasonable. Further, the assessors could change the subjective values and perform *what-if* scenario testing to produce best, worst and average case analyses. The project was initially expected to last 6 months but was re-scheduled after 1 year, then sought re-budgeting and still required significant re-planning on two further occasions. At the outset the team were optimistic about technical possibilities but feasibility was always a concern. In fact the biggest risks arose from the lack of realism concerning availability of staff and the lack of support for some crucial assumptions in the maintenance of the primary person and secondary person systems. The goals 'support for external persons' (67) and 'easing workload' (84) were uncertain from the start and were substantially jettisoned.

The nine goals in Table 9 represent 100% of the value assigned to the project. From the table we can see that 80% of our assessment was good (shown by the regular text, italic text and bold text) within which 43% (in regular text) was very good. Unfortunately the assessment for goal 72 (the bold italic text) was questionable as the end result was reported as being "painful". On investigation of the goal graph and expert judgement of feasibility we conclude the judgements were not themselves accurate. They may have been affected by a lack of shared understanding of the system.

During our analysis it became clear that a number of problems stemmed from a serious lack of shared understanding between the stakeholders at the time of project sanction. If our technique been used it is very likely that the project would not have started until greater confidence in the requirements had been achieved and that this would have brought about better shared understanding.

6 Future Work

Our technique depends upon goal analysis. When a project is founded on a goal analysis our technique is straightforward and natural to use. However in the real world of software development the situation is usually more complex. Urgency may tempt developers to use only ad-hoc methods of requirements analysis and project planning. In such cases, as mentioned earlier, a best effort graph, with an approximate sketch of the currently known requirements, needs to be created. As this will require extra resources it is crucial that this graph can be created economically and quickly in order that the momentum of the project is not lost.

This principle is being put to the test in a large venture capital project which is under development using agile processes. The Scrum process [19] was adopted in this project following two years of traditional process modelling. As the system was under development it was uneconomical to perform a retrospective goal analysis of the work done to date. Instead a goal graph was constructed in step with each short development phase, or *sprint*. Thus over time a comprehensive graph emerges and (more importantly) a goal graph becomes available for each sprint. These relatively small goal graphs are then used when evaluating confidence about the risk of failure of the sprint. The goal analysis creates requirements for the specific sprint whereas the work of previous sprints becomes a part of the environment.

We are also applying the technique to a venture capital supported project to develop a product that will solve problems of storing huge amounts of data. This had seen circa 20 person years of development activity and was based on IPR originally developed in government laboratories but had not led to a product. Early in 2006 a decision was taken to develop the product using the Scrum methodology. This allowed a build up of concrete functionality whilst evolving the required functionality through a series of sprints. We are in the early stages of applying our technique to this project whereby the value, adequacy and feasibility of each sprint is to be assessed as part of the gateway to the sprint backlog prior to starting the sprint.

This application to real industrial Scrum cycles has benefited not only the risk evaluation but has also been of considerable assistance to the test team in preparing sprint acceptance tests as it facilitates development of test cases for each requirement at requirements time. The goal analysis technique is surprisingly agile when used in this manner. A more mainstream development process which we believe our work will match well is the spiral development methodology [3] since it hinges on identifying risks and dealing with them early on.

This partitioning of an existing system into environment assumptions and new software with each sprint is akin to developing in iterations and increments as often happens when working with legacy systems. Our industrial experience shows that this situation is increasingly prevalent; indeed, the Scrum method in general is becoming increasingly popular with companies wishing to adopt an ‘agile’ approach. Thus following our initial experiences of applying our technique to agile projects we are encouraged to apply the technique to legacy systems as well as to new developments.

As mentioned earlier, our approach assumes that the requirements can be expressed in such a way that all leaf goals are independent. It also assumes that the requirements can be represented in a hierarchical decomposition. This assumption is somewhat restrictive and it potentially inhibits the treatment of some cross-cutting requirements such as performance and security [1, 16, 18]. In order to address such concerns and provide a generalized solution we are investigating the use of problem frames [9, 11] to structure and encapsulate subsystems and so provide interfaces that facilitate a formal decomposition and re-composition of subsystems.

The use of COTS components would clearly impact the project costings. In this case the metrics could be used to aid decision making when comparing different system's architectures. We intend to validate our technique through a number of case studies. Clearly the technique may be most applicable to particular application domains that are amenable to subjective analysis, and our future work will investigate the applicability of the technique and the composition of subsystems with nonfunctional requirements further.

7 Related Work

A process and tool called DDP (Defect Detection and Prevention) have been developed at JPL to facilitate risk management over the entire project life cycle [5]. DDP is intended for use in the aerospace industry. It uses a large set of risk elements, weighted requirements, pre-set formulae and information from experts. It is similar to our technique, but the process uses trees of requirements (rather than goal graphs), and trees showing why these requirements may not be achieved. The user can then select preventative measures, analyses, process controls and tests in order to minimize the residual risk. The process begins at the architecture stage, whereas our technique is intended for use during requirements analysis. There is software tool support for the DDP process, which weights risk elements by their impact on weighted requirements. The weights are obtained from the mission success criteria. The DDP process uses a large number of metrics and pre-determined industry-specific knowledge. This contrasts with our technique, which we have deliberately tried to keep as simple and lightweight as possible.

Karlsson, Wohlin and Regnell evaluated six different methods for prioritizing software requirements experimentally [13]. The six methods ranged from the analytic hierarchy process, AHP, through binary search trees to priority groups and found AHP to be the most promising approach. An overview of requirements prioritization [2] explains and critiques a range of methods (and their combination) for prioritization of factors such as importance, cost, risk, volatility etc. An example is given which uses stakeholders' opinions to rank the importance of a set of requirements. This work on prioritization does not examine how the risk assessment is arrived at. In contrast, our work defines requirements risk for individual requirements more precisely and provides a method that integrates with KAOS.

The use of expert opinion in software development in general is not a new idea: Freimut, Briand and Vollei [8] give a thorough account to justify its use during evaluation of the cost-effectiveness of inspections. They suggest that expert opinion may be needed if information regarding a phenomenon cannot be collected by any other affordable means or if information is not being collected within the timeframe that it is needed. They point out that expert data is subject to bias, uncertainty, and incompleteness but say that these problems can be prevented and controlled by means of carefully performed elicitation of expert estimates.

Ruhe et al. have presented an approach to decision support in requirements negotiation called "Quantitative WinWin" [20], which incorporates quantitative methods with Boehm's original WinWin approach. The approach uses the Analytical Hierarchy Process for a stepwise determination of the stakeholders' preferences in quantitative terms. These results are combined with methods for early effort estimation to evaluate the feasibility of alternative requirements subsets in terms of their related implementation effort. This work is particularly concerned with finding those subsets of requirements that can be implemented without exceeding a given maximum effort whereas our work is concerned with assessing the potential for error in systems development.

8 Conclusions

In this paper we have presented an easy-to-use technique for assessing risk in requirements analysis using goal graphs and judgements supplied by either stakeholders or experts. Using this technique, potentially risky projects can be detected at an early stage so that decisions can be taken about different courses of action.

This research is the first report of a risk assessment technique using high-level subjective metrics collected during requirements analysis using goal-graphs. Note that the technique can be used with assessments provided either by stakeholders or by experts. It will thus provide a meta-level risk assessment that could be very valuable to project managers and senior managers. We have shown how the information can be combined with data concerning the value and cost of goals. The technique has been empirically tested using a number of projects, the results from one of which has been described in detail. These tests have shown that we can indeed discover the parts of a project that are most likely to lead to problems using this technique. Our experience suggests that the technique is entirely sympathetic with the real world needs of industrial software development. We have also partially automated this technique by implementing a lightweight tool called KAOS Lite for goal sketching and automated data collection.

9 Acknowledgments

The authors would like to thank their collaborators for providing project data, particularly Kathryn Lewis of MSD, and the anonymous referees who reviewed this paper and provided very helpful comments. We would also like to thank Claes Wohlin and Barbara Kitchenham for their input to early versions of this paper.

10 References

[1] Baniassad, E., Clements, P.C., Araújo, J., Moreira, A., Rashid, A., Tekinerdogan, B.: "Discovering Early Aspects," IEEE Software, 2006, 23, (1), pp. 61-70.

[2] Bernader, P., Andrews, A.:" Requirements Prioritization" in Aurum, A. Wohlin, C. (Eds.): 'Engineering and Managing Software Requirements', (Springer, 2006), pp. 69–94.

[3] Boehm, B.W.: "A Spiral Model of Software Development and Enhancement," Computer, 1988, 21, (5), pp. 61-72.

[4] Boehm, B.W.: "Software Risk Management: Principles and Practices", IEEE Software, 1991, 8, (1), pp. 32-41.

[5] Cornford, S.L., Feather M.S., and Hicks, K.A.: “DDP - A tool for life-cycle risk management,” Proceedings of the IEEE Aerospace Conference, Big Sky, Montana, 2001, pp. 441-451.

[6] Dardenne, A., Fickas, S., and Lamsweerde, A. van: "Goal-directed concept acquisition in requirements elicitation", Proc. 6th IEEE Workshop System Specification and Design, Como, Italy, 1991, pp. 14-21.

[7] Darimont, R.: “Requirements Engineering with Objectiver: from Goal Analysis to Automatically Derived Requirements Documents,” RE’03 Exhibitors’ track, International Conference on Requirements Engineering (RE03), Los Alamitos, California: IEEE Computer Society Press, 2003.

[8] Freimut, B., Briand, L.C., and Vollei, F.: “Determining Inspection Cost-Effectiveness by Combining Project Data and Expert Opinion,” IEEE Transactions on Software Engineering, 2005, 31, (12), pp 1074-1092.

[9] Hall, J.G., Jackson, M., Laney, R.C., Nuseibeh, B., Rapanotti, L.: “Relating Software Requirements and Architectures using Problem Frames,” Tenth International IEEE Conference on Requirements Engineering, Los Alamitos, California: IEEE Computer Society Press, 2002, pp. 137-144.

[10] IEEE Std. 1233-1996, “IEEE Guide for Developing System Requirements Specifications”, IEEE June 1996.

[11] Jackson, M.: Problem Frames: Analysing and Structuring Software Development Problems, Addison Wesley, 2000.

[12] Karlsson, J., Ryan, K.: “A cost-value approach for prioritizing requirements,” IEEE Software, 1997, 14, (5), pp. 67-74.

[13] Karlsson , J., Wohlin, C., Regnell, B.: “An evaluation of methods for prioritizing software requirements,” Journal of Information and Software Technology, 1998, 39, (14-15), pp. 939-947.

[14] Keil, M., Cule, P.E., Lyytinen, K., Schmidt, R.C. “A Framework for Identifying Software Project Risks.” Communications of the ACM, 1998, 41, (11), pp. 76-83.

[15] Lamsweerde, A. van, Dardenne, A., Delcourt, B. Dubisy, F.: “The KAOS Project: Knowledge acquisition in automated specification of software.” Proceedings of the AAAI Spring Symposium Series, Stanford University, AAAI, March 1991, pp. 59-62.

[16] Mylopoulos, J., Chung, L., Nixon, B.: “Representing and using nonfunctional requirements: a process oriented approach,” IEEE Transactions on Software Engineering, 1992, 18, (6), pp. 483-497.

[17] Pinheiro, F.A.C.: “Requirements Honesty” International Workshop on Time-Constrained Requirements Engineering (TCRE'02), Germany Sep. 2002.

[18] Rashid, A., Sawyer, P., Moreira, A., Araujo, J.: “Early aspects: a model for aspect-oriented requirements engineering.” Requirements Engineering 2002 (RE’02), Germany 2002, pp. 199-202.

[19] Rising L., and Janoff, N.: “The Scrum Software Development Process for Small Teams”, IEEE Software, 2000, 17, (4), pp. 26-32.

[20] Ruhe, G., Eberlein, A., and Pfahl, D.: “Quantitative WinWin: a new method for decision support in requirements negotiation.”, Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering (SEKE '02), Ischia, Italy, July 2002, pp. 159-166.

[21] Wallace, L., Keil, M.: “Software project risks and their effect on outcomes.” Communications of the ACM, 2004, 47, (4), pp. 68-73.

[22] Wallace, L., Keil, M., Rai. A.: “How Software Project Risk Affects Project Performance: An Investigation of the Dimensions of Risk and an Exploratory Model,” Decision Sciences, 35, (2), March 2004, pp. 289-321.

APPENDIX

TABLE 10

Calculations of the aggregated metrics

Risk Factor	Metric Aggregation: total for project
RF1	$\frac{1}{G}\sum_{g=1}^{G}\mathrm{Env}(g)$ where G = total no. of goals
RF2	$\frac{1}{G}\sum_{g=1}^{G}\mathrm{Req}\,(g)$ where G = total no. of goals
RF3	$\frac{1}{G}\sum_{g=1}^{G}\mathrm{Refine}(g)$ where G = total no. of goals
RF4	$\frac{1}{G}\sum_{g=1}^{G}\mathrm{Mandate}(g)$ where G = total no. of goals
Average value	$\frac{1}{G}\sum_{g=1}^{G}\mathrm{Value}(g)$ where G = total no. of goals
Total cost	The estimated total cost of a project p is the sum of the estimated costs of its operationalised requirements: $\mathrm{TOTALCOST}\ (p) = \sum_{r=1}^{R}\mathrm{Cost}\ (r)$ where R = total no. of operationalised requirements.

TABLE 1

The 4 risk factors and associated metrics

Risk Factor	Raw data	Description
RF1: Environment	ENV	Probability that the assumptions can be satisfied by the environment
RF2: Achievability	ACHIEVE	Probability that the requirements are achievable
RF3: Refinement	SOUND	Probability that the refinements are sound
RF4: Mandate	MANDATE	Probability that the requirements are mandated

TABLE 2

Calculations of the metrics for the Risk Factors

Risk Factor	Definition of metric for each leaf and goal
RF1	The RF1 metric for an operationalised assumption *a* is calculated as the average of the assessors scores: RF1(a) = $\frac{1}{N}\sum_{n=1}^{N}\mathrm{ENV(n)}$ where N = total no. of assessors, ENV is the probability that an assumption can be fulfilled within project and domain constraints, estimated for leaves by assessors. A goal's RF1 is calculated as the product of its children's RF1 values. If there are no operationalised assumptions for a goal then RF1 = 1
RF2	The RF2 metric for an operationalised requirement *r* is calculated as the average of the assessors scores: RF2(r) = $\frac{1}{N}\sum_{n=1}^{N}\mathrm{ACHIEVE(n)}$ where N = total no. of assessors, ACHIEVE(n) is the probability that a requirement can be achieved within project and domain constraints, estimated for leaves by assessors. A goal's RF2 is calculated as the product of its children's RF2 values. If there are no operationalised requirements for a goal then RF2 = 1
RF3	The RF3 metric for a goal *g* is calculated as the average of the assessors scores: RF3(g) = $\frac{1}{N}\sum_{n=1}^{N}\mathrm{SOUND}\,(n)$ where N = total no. of assessors, SOUND(n) is the probability that the refinement argument for a goal is sound, estimated by assessors. For leaf goals this equates to estimating whether further refinement is needed. The default RF3 metric for goals is 0.5
RF4	The RF4 metric for a goal *g* is calculated as the average of the assessors scores: RF4(g) = $\frac{1}{N}\sum_{\mathrm{n}=1}^{\mathrm{N}}\mathrm{MANDATE(n)}$ where N = total no. of assessors, MANDATE is the probability that the stakeholders' mandate for a goal is sound, estimated by assessors. The default RF4 metric for goals is 0.5

TABLE 3

Calculations of the indirect metrics

Definition of metric for each leaf goal
Feasibility F(g) = RF1(g)*RF2(g)
Adequacy A(g) = RF4(g)* $\prod_{i=1}^{L}$ RF3(g_i)
where L = number of levels from the root goal g_1 to the leaf goal g_L (and where g = g_L)

TABLE 4

Calculations of the Cost, Value and Priority metrics

Definition of metric for each goal
$\text{Value}(g) = \frac{1}{N}\sum_{n=1}^{N}\text{VALUE}(n)$ where N = total no. of assessors, VALUE is the business value of a goal as rated by assessors (its contribution to the project) on an ordinal scale greater than or equal to 0. For a goal not rated by assessors Value is inherited from its immediate parent(s) s.t. the children's Values sum to the parent's Value. A goal with 2 or more parents will inherit the sum of its parents' Values.
The TOTALVALUE of a project is the sum of the values of its operationalised requirements: $\text{TOTALVALUE}(p) = \sum_{r=1}^{R}\text{Value}(r)$ and R = total no. of operationalised requirements. The proportional value of a leaf goal g, PValue (g), is the value of the goal divided by the total value of the project p expressed as a percentage: PValue (g) = (Value (g)/TOTALVALUE (p))*100, TOTALVALUE(p) $\neq$ 0 PValue (g) = 0 otherwise PValues for the leaf goals are normalised such that the total sums to 100%.
The cost of an operationalised requirement r, $\text{Cost}(r) = \frac{1}{N}\sum_{n=1}^{N}\text{COST}(n)$ where N = total no. of assessors, COST is the cost of the operationalised requirement (either in FPs, staff days or LOC) as estimated by assessors.
The cost of a goal is the sum of the costs of all its operationalised requirements: $\text{Cost}(g) = \sum_{r=1}^{R'}\text{COST}(r)$ where R' = total no. of operationalised requirements for a goal. The cost of an operationalised assumption is zero.
The TOTALCOST of a project is the sum of the costs of its operationalised requirements: $\text{TOTALCOST}(p) = \sum_{r=1}^{R}\text{Cost}(r)$

and R = total no. of operationalised requirements in the graph.

The proportional cost of a leaf goal g, PCost (g), is the cost of the goal divided by the total cost of the project p expressed as a percentage:

PCost(g) = (Cost(g)/TOTALCOST(p))*100, TOTALCOST(p) $\neq$ 0

PCost (g) = 0 otherwise

PCosts for the leaf goals are normalised such that the total sums to 100%.

Priority1(g) = $\frac{\text{PValue(g)}}{\text{PCost(g)}}$ *100

if PCost(g)$\neq$ 0

Priority1(g) = 100% otherwise

where N = total no. of assessors.

Priority2(g) = $\text{PValue(g)} * (1 - \text{PCost(g)})$ *100

if PCost(g) $\neq$ 100%,

Priority2(g) = 100% otherwise

where N = total no. of assessors

TABLE 5

Raw data for WeighCom, root goal *'Maintain walk-on scales normal operation'*

Name	**RF1**	**RF2**	**RF3**	**RF4**	**Cost**	**Value**
G1	**1**	**1**	**1**	**1**	**0**	17
G2	1	0.64	**0.9**	**1**	100	**100**
G3	**1**	**0.8**	**0.7**	**1**	**50**	18
G4	**1**	**1**	**1**	**1**	**0**	17
G5	1	0.8	**0.7**	**1**	50	35
G6	**1**	**0.8**	**1**	**0.9**	**30**	18
G7	1	0.8	**0.8**	**0.6**	30	35
G8	**0.8**	1	**1**	**1**	**0**	40
G9	0.8	1	**0.9**	**0.9**	0	**40**
G10	**1**	**1**	**0.9**	**1**	**20**	30

Key: bold entries are expert judgements.

TABLE 6

Feasibility and Adequacy metrics for WeighCom leaf goals for the root goal *'Maintain walk-on scales normal operation'* together with proportional cost and proportional values.

Name	**Feasibility**	**Adequacy**	**PCost**	**PValue**
G1	1	0.63	0%	12%
G3	0.8	0.44	50%	13%
G4	1	0.72	0%	12%
G6	0.8	0.65	30%	13%
G8	0.8	0.9	0%	29%
G10	1	0.81	20%	21%
			100%	100%

TABLE 7

Leaves from the *Do not proceed* quadrant

Goal	Name	PValue	PCost	Feasibility	Adequacy
63	Visiting staff and contractors are not allocated a UPI and are thus excluded	0.4%	0.0%	0.00	0.16
67	Allow external persons access (details TBD)	10.8%	8.9%	0.00	0.40
77	Access to each SS services is gated to identify the appropriate UPI which is then maintained throughout the ensuing session	2.1%	4.4%	0.20	0.40
81	For each non-connecting SS the most recent UPI-status pairs are persisted for reference.	4.7%	4.4%	0.30	0.48
90	User initiated extraction of data from PPSs in form suitable for use in producing College directory service	1.7%	4.4%	0.30	0.36
92	A user initiated mechanism providing college directory, "all staff" and telephone directory information from PPSs	1.7%	2.2%	0.50	0.36
94	Office process improvements	3.5%	0.0%	0.50	0.28
101	Real-time API returning status for any submitted UPI	4.7%	4.4%	0.40	0.48
102	Each connecting SS is supported to use real-time status API	4.7%	4.4%	0.20	0.40
103	A scheduled batch-mode service provides updates of all known UPI-Status to each non-connecting SS	2.3%	4.4%	0.50	0.48
104	Every non-connecting SS cooperates with the batch-mode service	2.3%	4.4%	0.30	0.48
122	Data extracted from SSs and PPSs for person centric reports is accurate.	3.5%	4.4%	0.20	0.36
123	The PPs and SSs support the necessary querying based on UPI to satisfy statutory obligations on personal data access	3.1%	0.0%	0.50	0.50
128	Each SS acts on the UPI linked Status to provide the right degree of service to the individual.	2.1%	2.2%	0.30	0.56

TABLE 8

Leaves from the *Proceed* region

Goal	Name	PValue	PCost	Feasibility	Adequacy
3	Maintain Comprehensive technical and user documentation.	9.4%	0.0%	0.80	0.70
4	Re-engineering of the UPI will enable future modification at low cost.	9.4%	0.0%	0.60	0.70
61	The status and data about staff is correctly maintained on the HR system	0.4%	0.0%	0.80	0.72
62	The status and data about students is correctly maintained on the REG system	0.4%	0.0%	0.80	0.72
82	Staff operating the primary systems maintain current status for all UPI's pertinent to each primary system	1.6%	0.0%	0.70	0.80
86	Only the PPSs hold personal data to be matched to a UPI	2.0%	0.0%	0.60	0.80
124	The existence of the UPI allows adequate facility to querying across PPSs and SSs.	3.1%	0.0%	0.70	1.00

TABLE 9

Assessments vs. Outcomes for Root Goals

Goal	Description	Threat	PValue	Outcome	'Pain'
20	Every UPI is maintained with a current status	Low (2/7)	6%	Full	Low
74	Every eligible individual is assigned a UPI	None (0/8)	6%	Full	Low
56	Every individual's access rights to any secondary system is based on their UPI linked status	High (7/7)	6%	Partial	Medium
25	To enhance maintainability compared to the original UPI which depends upon vulnerable knowledge	None (0/2)	19%	Full	Low
31	*Optimally determine external person's eligibility to use College facilities.*	*High (1/1)*	*10%*	*Removed*	*Low*
34	**Operate in a "real time" as opposed to "batch" mode**	**High (5/5)**	**17%**	**Partial**	**Medium**
38	To comply with statutory obligations on personal data.	Medium (1/2)	6%	Partial	Low
72	***To improve the efficiency of daily UPI matching compared with the original UPI***	***None (0/2)***	***19%***	***Full***	***High***
84	*To ease the workload of College Staff*	*High (4/4)*	*10%*	*Uncertain*	*High*

Key: (n/m): *n* leaves of *m* belonging to the goal are in the *Do Not Procee*d zone.

Regular text: Assessment Correct

Italic text: Assessment Ok

Bold text: Assessment Pessimistic

Bold italic text: Assessment Questionable

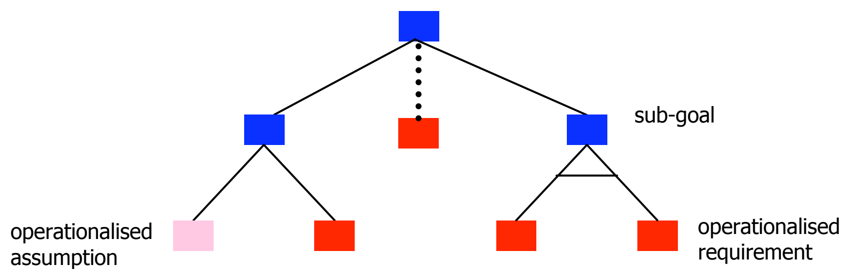
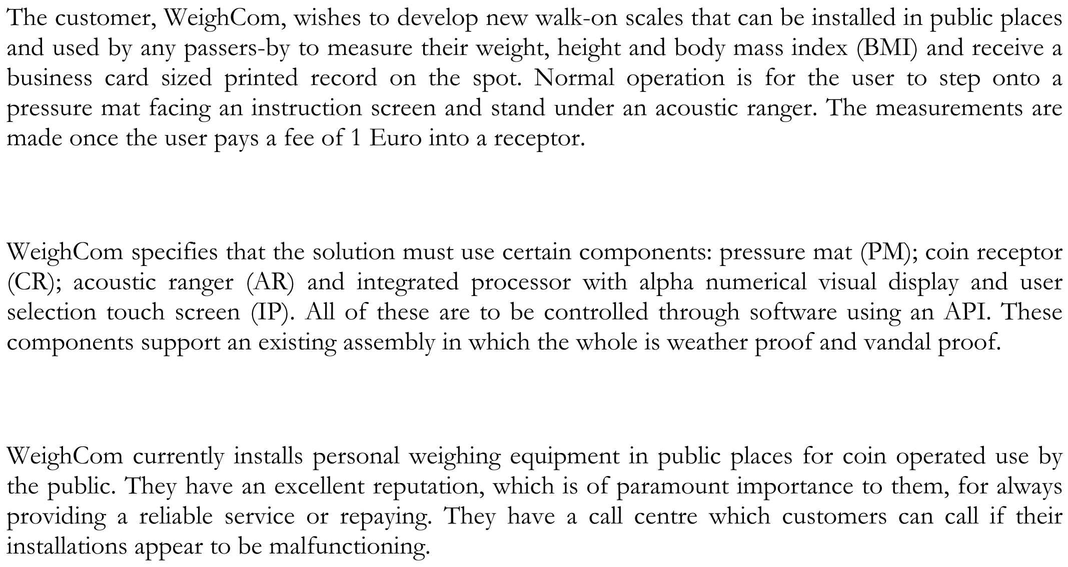


Fig. 1: An example of a goal graph

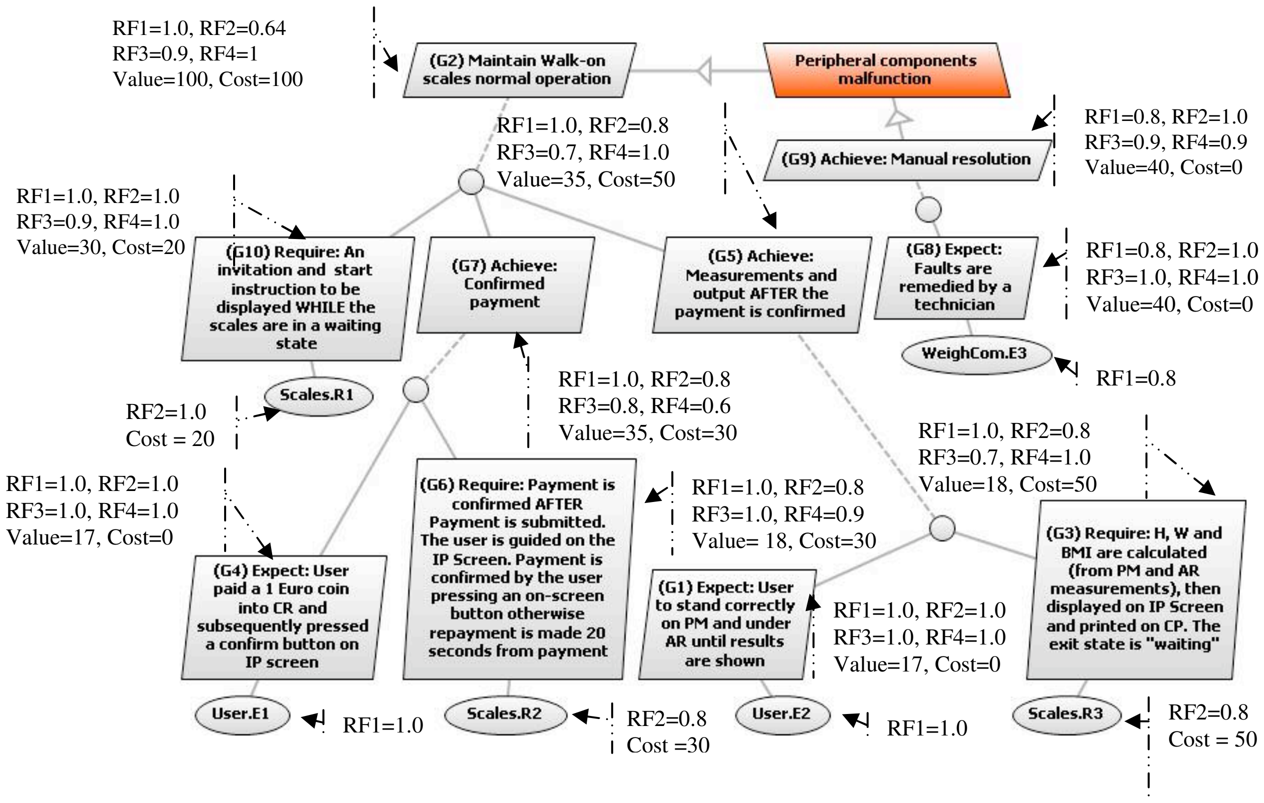


The customer, WeighCom, wishes to develop new walk-on scales that can be installed in public places and used by any passers-by to measure their weight, height and body mass index (BMI) and receive a business card sized printed record on the spot. Normal operation is for the user to step onto a pressure mat facing an instruction screen and stand under an acoustic ranger. The measurements are made once the user pays a fee of 1 Euro into a receptor.

WeighCom specifies that the solution must use certain components: pressure mat (PM); coin receptor (CR); acoustic ranger (AR) and integrated processor with alpha numerical visual display and user selection touch screen (IP). All of these are to be controlled through software using an API. These components support an existing assembly in which the whole is weather proof and vandal proof.

WeighCom currently installs personal weighing equipment in public places for coin operated use by the public. They have an excellent reputation, which is of paramount importance to them, for always providing a reliable service or repaying. They have a call centre which customers can call if their installations appear to be malfunctioning.

Fig. 2. A requirements definition for the calculation of body mass index.


RF1=1.0, RF2=0.64
RF3=0.9, RF4=1
Value=100, Cost=100
(G2) Maintain Walk-on scales normal operation
Peripheral components malfunction
RF1=1.0, RF2=0.8
RF3=0.7, RF4=1.0
Value=35, Cost=50
RF1=0.8, RF2=1.0
RF3=0.9, RF4=0.9
Value=40, Cost=0
(G9) Achieve: Manual resolution
RF1=1.0, RF2=1.0
RF3=0.9, RF4=1.0
Value=30, Cost=20
(G10) Require: An invitation and start instruction to be displayed WHILE the scales are in a waiting state
(G7) Achieve: Confirmed payment
(G5) Achieve: Measurements and output AFTER the payment is confirmed
(G8) Expect: Faults are remedied by a technician
RF1=0.8, RF2=1.0
RF3=1.0, RF4=1.0
Value=40, Cost=0
WeighCom.E3
RF1=0.8
Scales.R1
RF2=1.0
Cost = 20
RF1=1.0, RF2=0.8
RF3=0.8, RF4=0.6
Value=35, Cost=30
RF1=1.0, RF2=0.8
RF3=0.7, RF4=1.0
Value=18, Cost=50
RF1=1.0, RF2=1.0
RF3=1.0, RF4=1.0
Value=17, Cost=0
(G6) Require: Payment is confirmed AFTER Payment is submitted. The user is guided on the IP Screen. Payment is confirmed by the user pressing an on-screen button otherwise repayment is made 20 seconds from payment
RF1=1.0, RF2=0.8
RF3=1.0, RF4=0.9
Value= 18, Cost=30
(G4) Expect: User paid a 1 Euro coin into CR and subsequently pressed a confirm button on IP screen
(G1) Expect: User to stand correctly on PM and under AR until results are shown
RF1=1.0, RF2=1.0
RF3=1.0, RF4=1.0
Value=17, Cost=0
(G3) Require: H, W and BMI are calculated (from PM and AR measurements), then displayed on IP Screen and printed on CP. The exit state is "waiting"
User.E1
RF1=1.0
Scales.R2
RF2=0.8
Cost =30
User.E2
RF1=1.0
Scales.R3
RF2=0.8
Cost = 50

Key: Each right-sloping parallelogram represents a goal

Each left-sloping parallelogram represents an obstacle

Each oval represents an operationalised requirement or operationalised assumption

Fig. 3. WeighCom: Annotated goal graph for root goal *'Maintain walk-on scales normal operation'*

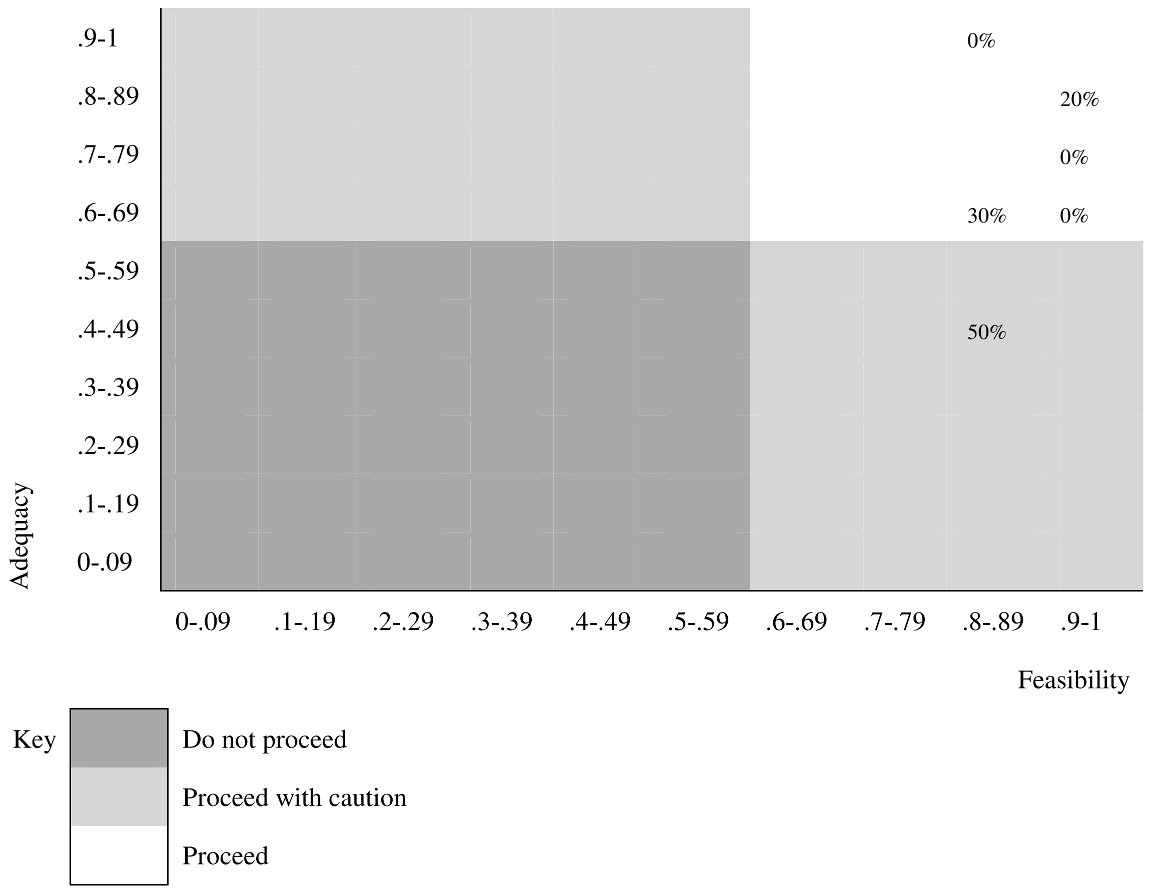


Fig. 4. Proportion of leaf goal costs (PCosts) for Feasibility versus Adequacy

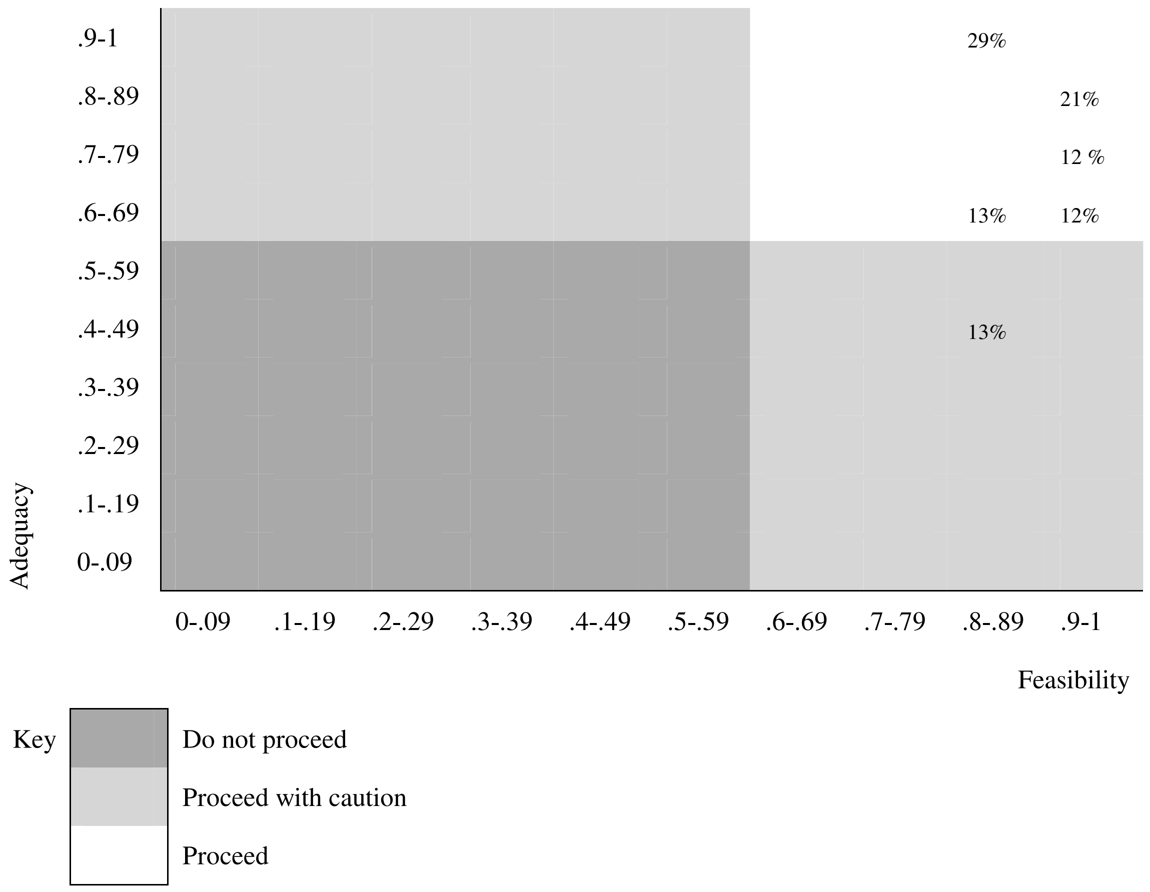


Fig. 5. Proportion of leaf goal values (PValues) for Feasibility versus Adequacy

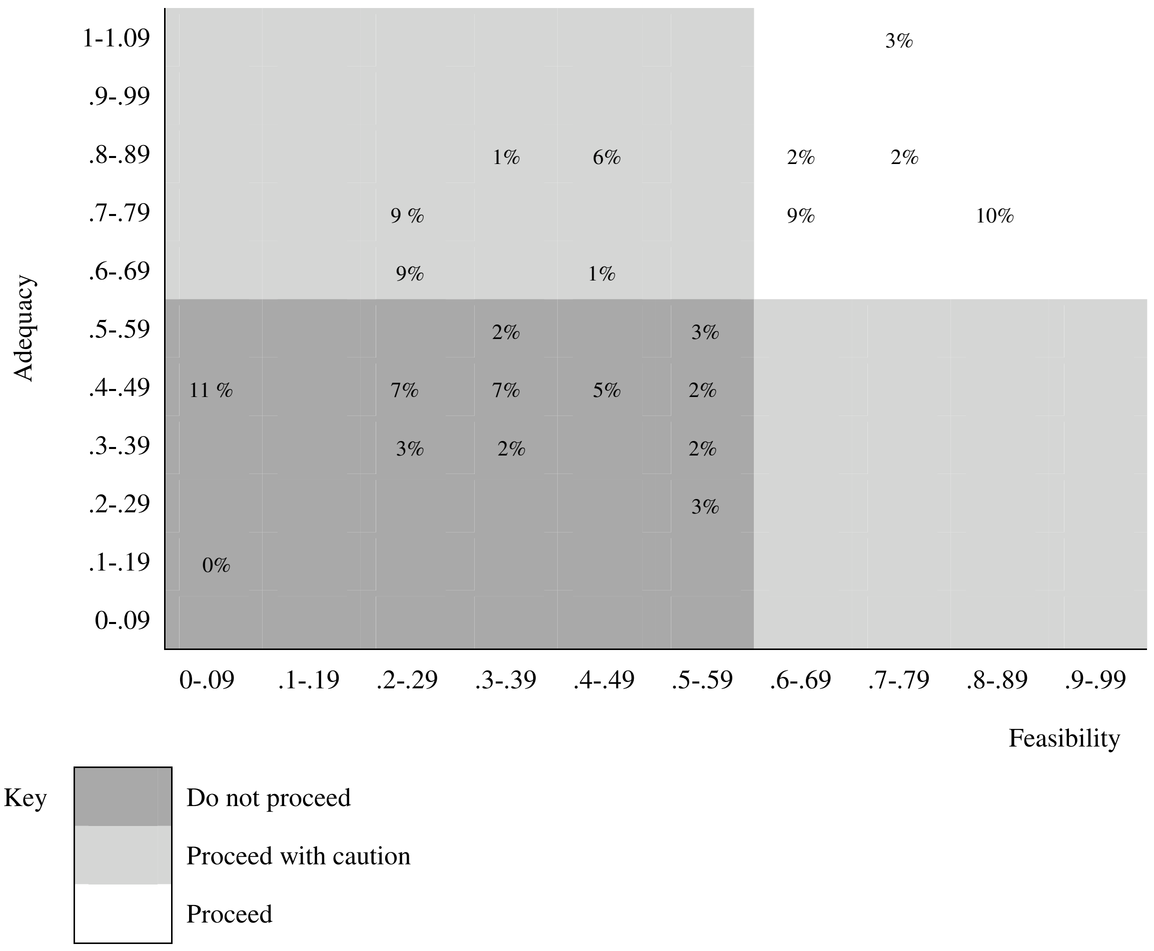
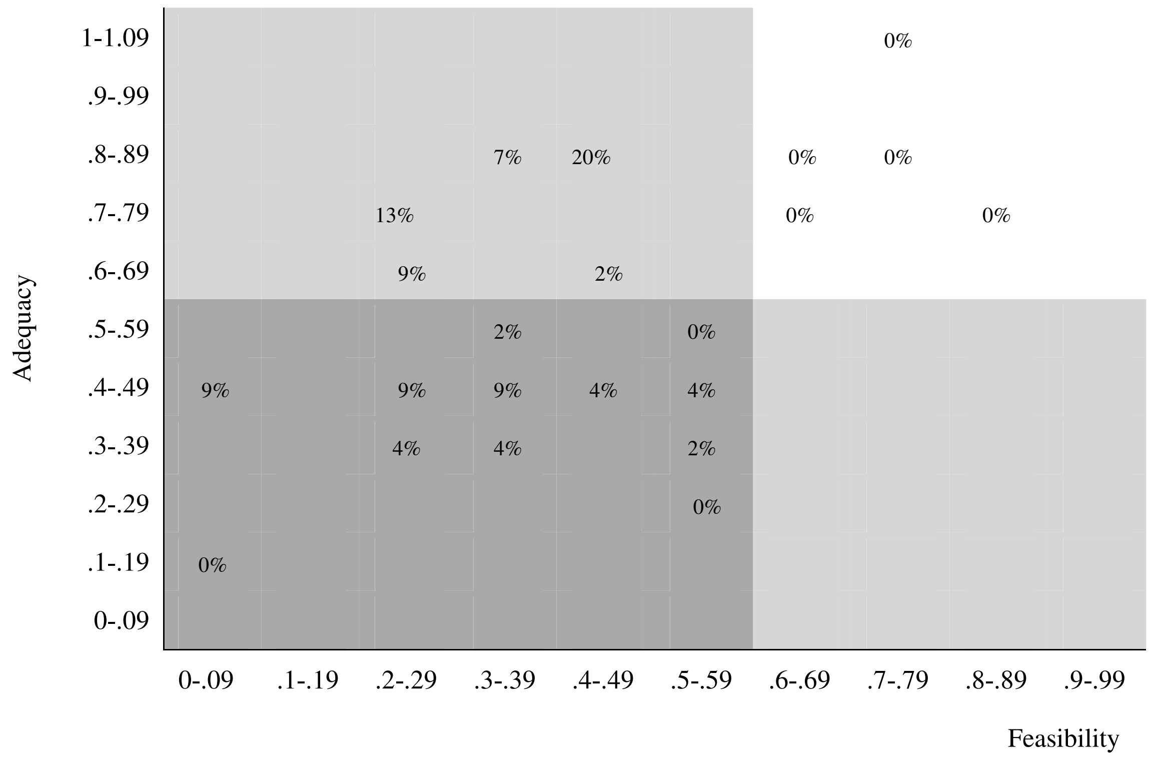


Fig. 6. Proportion of leaf goal values (PValues) for Feasibility versus Adequacy



Key		
	▓	Do not proceed
	▒	Proceed with caution
	□	Proceed

Fig. 7. Proportion of leaf goal costs (PCosts) for Feasibility versus Adequacy