Requirements Uncertainty: Influencing Factors and Concrete Improvements

Christof Ebert
Alcatel
54, rue la Boetie
F-75008 Paris
christof.ebert@alcatel.com

Jozef De Man
Alcatel
Francis Wellesplein 1, B-2018 Antwerp
Also: Ghent University
jozef.de_man@alcatel.be

ABSTRACT

Practically all industry studies on software project results conclude that good requirements engineering plays a pivotal role for successful projects. A key reason for project failures is insufficient management of changing requirements during all stages of the project life cycle. This article investigates one of the root causes for changing requirements, namely requirements uncertainty. In an experimental field study we looked into four underlying drivers for requirements uncertainty. We found several techniques must be used simultaneously to see tangible success. Using only one such technique in isolation doesn't make a difference. The field study is supported by extensive data from well over 200 projects stemming from very different business areas of Alcatel over a period of two years. Results are presented with practical experiences to allow effective transfer.

Categories and Subject Descriptors

D.2.1[Requirements/Specifications]: Elicitation methods, Methodologies (e.g., object-oriented, structured). K.6.1[Project and People Management]: Life cycle, Management techniques, Systems analysis and design

General Terms

Management

Keywords

product management, product life cycle, process improvement, requirements uncertainty, requirements engineering

1. INTRODUCTION

Requirements are uncertain. That's almost by definition, which is captured by an old requirements analyst slogan, stating "I know it when I see it". These uncertainties are increasing in fast changing markets, as we observe in various industries. Requirements uncertainties originate from various causes, but yield similar results. Projects are delayed and do not fulfill the original expectations.

ICSE'05, May 15–21, 2005, St. Louis, Missouri, USA.

While requirements uncertainty initially is only one risk amongst others, it is worth looking into how it is handled in real projects and what contributions from software engineering research and practice could contribute towards managing requirements uncertainty, thus improving predictability and reducing time to market.

We evaluate in this article what are the root causes of requirements uncertainty, and how we can manage a project in light that we don't know all we would like to know. We found that requirements uncertainties exist independent of project size, scope and environment. In order to identify concrete solutions, we break down the problem of requirements uncertainty into several aspects that we address separately. From this root cause investigation we derive a process that mitigates the major origins or impacts of requirements uncertainties.

The underlying reference process of this experimental field study covers requirements definition (i.e., elicitation, inception) and requirements management (i.e., monitoring, construction). It consists of four elements, namely

1. installing a core team of empowered stakeholders with shared responsibility for the product release or project,
2. reinforcing a standardized product life cycle with early gate reviews,
3. ensuring online monitoring and traceability of all requirements,
4. executing a periodic evaluation of each project from a project and portfolio perspective.

All these techniques are since long considered product development best practices[3,13,17]. The novelty of our approach is that we underline that these techniques MUST be used in combination to offer best results. Used in isolation only adds to process complexity without tangible benefits.

These four process elements have been introduced in Alcatel's business areas at different speed with different intensity over the past years, thus allowing to correlate the degree of process change (i.e., application of each of the four process elements) with observed project success as recorded in our company-wide project history database. When analyzing the (business) impacts of requirements uncertainty on project success, we found that the major impact is on predictability of handover date versus the commitments at project start. Customers and markets do not tolerate delays. We therefore relate in this empirical study success with predictability of delivery date.

An observational field study has been performed covering 246 projects of Alcatel in various domains with different size and scope. We observed in this empirical study that using one or two of the mentioned four reference process elements has no significant impact on improving predictability. However, combining 3 or 4 of these elements in a project reduces delays significantly!

It's this comprehensive focus on several process elements impacting requirements elicitation, definition and management, which effectively copes with requirements uncertainty. This explains why so many concepts from literature for handling requirements uncertainty (e.g., prototyping or risk management) had no impact. They were mostly used in isolation, and not combined with other mentioned techniques. For instance, imagine risk management is performed *inside* a project and it suggests a certain buffer for mitigating requirements uncertainty. If not supported by the – *external* – product manager, this mitigation has no impact, because he would actually come with more changes because engineering is seemingly flexible.

Let's briefly recapture the key terminology. A *requirement* is a "condition or capability needed by a user to solve a problem or achieve an objective"[15]. The on-line dictionary of computing is more to the point, claiming that "a common feature of nearly all software is that the requirements change during its lifetime" [10]. *Requirements uncertainty* means that within a software system, requirements are not known until it is practically used [19]. This uncertainty results from software requirements creating an ill-defined problem [2]. Requirements uncertainty results in requirements instability [2,6], which means project delays [16,24,5,22]. Dealing with changing requirements and uncertainty therefore has been outlined as a major research arena facing the software engineering community [6].

The paper is organized as follows. Section 2 outlines the problem statement and presents the state of research *and* state of practice in dealing with requirements uncertainty. We specifically try to explain why the well-known answers, such as prototyping, did not make it into actual software engineering. Section 3 describes the setting of this observational case study. It investigates how the very early phases in the product life cycle, namely inception and elaboration contribute to requirements uncertainty and how the underlying root causes could be resolved. We provide a formalized root cause relationship drawn from project experiences. Section 4 provides concrete results from this empirical study. We present the results in a format that facilitates direct transfer. Lessons learned are captured in section 5. We provide concrete experiences with the proposed four process elements and show what to observe when transferring the results. We have tried to abstract as much as possible from any specific observation related to the products used in this study to allow easy generalization and transfer to other industries. Finally, section 6 summarizes this case study and provides an outlook on new or remaining questions that we must address in future studies.

2. REQUIREMENTS UNCERTAINTY

A clear relationship between requirements management and project success has been reported since the seventies from various empirical research and surveys, typically putting insufficient requirements management (covering both development and change management of requirements) on top of the list of factors contributing to project failures [16,24,5,22,19]. Typical results from requirements uncertainty are insufficient project planning, continuous changes in the project, delays, configuration problems, defects, and overall customer dissatisfaction due to not keeping commitments or not getting the product they expect.

As a recent example, the 2003 Chaos report of IT project failure shows that, based on data from over 13,000 cases, only 34% of IT projects were considered successful [24]. 15% of projects were complete failures and the balance of 51% was what is referred to as challenged. Challenged in this context means a project overrun on time and/or cost. Data from the evaluated projects showed that only 52% of the originally allocated requirements appear in the final released version.

The traditional rule of thumb indicates a change rate 1-3% per month in terms of effort related to the allocated requirements [21,16,9]. This translates into more than 30% overall requirements change rate in terms of total project effort for a project duration of two years. As a consequence, many contractors and also clients strongly urged to reduce project duration towards one year maximum. With this reduction of project duration, projects indeed performed better [24]. However, this number seems not yet to be robust and stable – being a universally useful software engineering factor. We faced in the past years an increase of change rates to even double that amount, i.e., 30% in a one-year project. This cannot anymore be fixed with only shortening project duration and working with iterations. A common denominator of requirements changes is that they practically always result in project delays.

With product managers, analysts and engineers becoming paranoid on these ever-lasting requirements changes and delays, we observed in parallel an increasing duration of the analysis (or elaboration) phase of a project, that is, before the actual project start. This syndrome, often called "paralysis by analysis" contributed to project duration and cost, but did not really improve much on the side of reducing or coping with requirements changes. We thus face two vicious circles caused by requirements uncertainty that both contribute to delays (figure 1).

Software engineering research has focused on two major questions, namely how to extract the "right" requirements and how to deal with changing requirements. We find several approaches how to extract the right requirements, such as

- elicitation and analysis techniques (e.g., creating scenarios and use cases, interviewing different stakeholders, extracting requirements from an existing system, synthesizing requirements from user needs and behaviors, uncovering requirements by experiments or prototypes, determining problem frames) [5,11,18,12,14]
- psychological techniques for identifying weaknesses (e.g., context-free questioning, workshops, analyzing different viewpoints and interaction schemes, interaction theory, protocol analysis) [12,25,13].

For dealing with continuous requirements changes, the key techniques are

- evolutionary life cycles and prototyping (e.g., JAD, incremental development, various agile methods) [1,16,21,17]
- reduced cycle time from inception to delivery (e.g., Feature-driven development) [17,21]

- sensitivity analysis (i.e., determining the localization, scope and impacts of changes, portfolio management) [9,2,23,13]
- practical risk management (e.g., traceability, impact analysis, improved maintainability, modularity, isolating features that are subject to changes) [1,20,22,7,19,3,13].

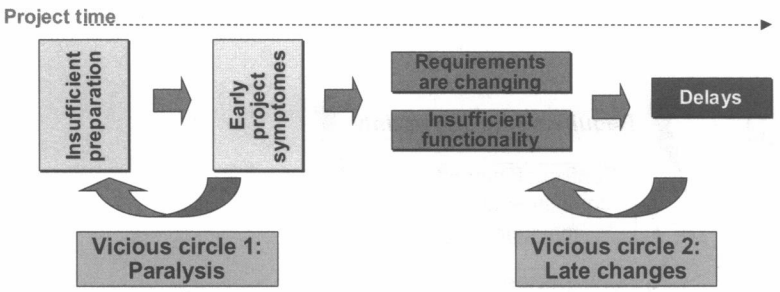

Figure 1: Focus of this study

Seemingly the problem description is clear and several different techniques are available to cope in due time with requirements uncertainty and requirements changes. So we asked ourselves why these techniques are not (more) widely used in industry. In fact, we realized in our own projects but also in discussions during previous requirements engineering conferences that except of iterative development, none of above mentioned techniques really made it to mainstream usage and university education. Using a specific elicitation technique from the broad range described before is insufficient due to the inherent weaknesses of each single technique [11,14]. In fact experienced analysts use them rather mixed, but without specific rules one could pass on to practitioners [14]. Evolutionary techniques combined with repetitive risk management, specifically prototyping, which was always claimed as the best way to deal with requirements uncertainty, are rarely used in practice because product managers fear such "vague results" [24]. Changes are addressed primarily ex post [2]. This is increasingly done with good tools support to visualize and manage requirements changes [7,12,14].

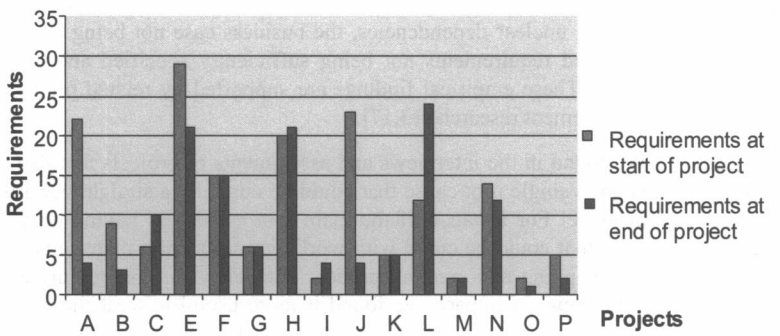

Figure 2: A project survey of 15 projects with respective requirements evolution during project duration

In line with [11,18] we claim that poor usage of mentioned techniques for dealing with requirements uncertainty is primarily a problem of technology transfer. Our hypothesis, which we examine in this article, is that a single ("silver bullet") technique as it's often suggested in literature does not help because too many environmental characteristics impact effective usage.

3. THE STUDY

The field study is built on communications products and solutions within Alcatel's three business groups. Alcatel develops leading-edge technologies and provides innovative communication solutions for telecom operators, internet access providers, and companies, enabling them to deliver value-added voice, data and video communication services. With sales of Euro 12.5 Billion in 2003 and a workforce of 60,000 employees, Alcatel operates in 130 countries. Specific and dedicated solutions for customers have been provided in this market for many years. The challenge today is to build upon technology platforms, while offering tailored solutions.

We have chosen a set of 246 projects for this empirical study. They were finished in the timeframe of January 2002 until October 2003. The end date was always time-stamped (with respect to this study) at the gating review that starts deployment or delivery to customers, thus ensuring comparability. The projects are representative for Alcatel's overall business in such heterogeneous market segments like mobile and fixed communication networks, mobile handsets, fixed and mobile solutions, enterprise communication or transport and satellite infrastructure. All projects that had recorded valid data in the history database were selected. No "outliers" have been thrown out, thus ensuring that we do not oversee influences from other parameters than the four process elements that we observed in detail.

The projects evaluated in this study, though primarily in the domains of embedded systems or software applications cover a wide range of the entire world of software dominated projects. Results therefore should apply to other industries than communication, which is also supported by benchmarks we have been doing in the field of automotive, defense and information systems (for published records see also: [5,16,17,21]).

From a business perspective we evaluate in this study how to reduce project delays caused by requirements uncertainty (figure 1). We did not investigate effects such as on sales, revenues or customer satisfaction because they have much more influences from the market situation and are thus difficult to compare. Delays have a clear root cause relationship to requirements uncertainty, as we show in the next section.

Requirements uncertainties have different causes. We focus on what we call early project symptoms, that is "handles" inside the scope of the project that can be effectively managed by the project team. Looking towards the very early insufficiencies won't help much because it's not yet tangible and certainly in the midst of too many stakeholders that do not yet see what they really want. Looking towards the effects of requirements uncertainty, namely changing requirements and insufficient functionality, is too late to be managed by the project team. We therefore evaluate within this field study two related hypotheses on the combined effects of selected best practices.

4. OBSERVATIONS FROM THE FIELD STUDY

The practical dilemma is that requirements are uncertain and known techniques for coping with uncertainty are rarely used. As

a first step we investigated by interviews and lessons learned reports what might be the root causes. We observed:

- Requirements management and specifically formal elicitation and analysis techniques are perceived as overly "technical". Stakeholders, especially outside the engineering domain, have little understanding for the process steps and intermediate results.
- Uncertainties tend to "vanish" at the interface of marketing, product management and engineering. This is typically caused by the need to achieve initial consensus over defined and accessible contents. Often the prevailing attitude is to start and fix it later, resulting in delays.
- Customers have not much time and resources to actively contribute to projects beyond what is necessary for contracting and monitoring. Agile and evolutionary techniques thus have their limits in getting sufficient customer support.
- Due to not getting all requirements specified in sufficient detail, engineers and project managers tend to guess actual needs and thus answer to uncertainties.
- Product managers and project managers focus on those risks that are meaningful to them. A certain type of risks (e.g., marketing) is not treated as it cannot be approached with product or project management "language".
- Prototyping and evolutionary development is not often used – except the domains of user interfaces and hardware innovations. It's considered difficult to plan (i.e., when will the prototyping cycles be closed?) and it creates configuration risks. What we increasingly see is plain iterative and feature-driven development and reduced cycle time as a form to mitigate risks of delays.

As a practical example let us look into a concrete snapshot of 15 projects from one of our product groups which were developed in 2002 timeframe (figure 2). We found that 73% of a project's requirements are changing in average (median: 50%). Typically one third of the changes is of technical background (e.g., a spec was infeasible for design), while two thirds are of commercial background (covering the majority of requirements uncertainties).

As a first step we qualitatively evaluated the impacts of requirements uncertainty on project success. From interviews with project managers, project teams and product managers we found that the key business impact resulting from requirements uncertainty is project delays. Delays can have various effects, such as not keeping a contracted deadline in a specific customer solution, or being late in the market versus competition with a new product release. We then went backwards towards identifying root causes that could be attacked (figure 3). Delays are mostly caused by ever-changing requirements or having the right functionality not in place. There is a vicious circle, as many product managers pointed out, between delays and content changes. In a highly competitive market, a sales person that is confronted with a delay of a project would try to achieve a win-win situation with supplier and customer by offering additional functionality to compensate for the delay. Unfortunately this additional functionality bears two risks, namely it could cause additional delays and the customers might get used to this perceived flexibility. Being highly flexible almost always results in more delays.

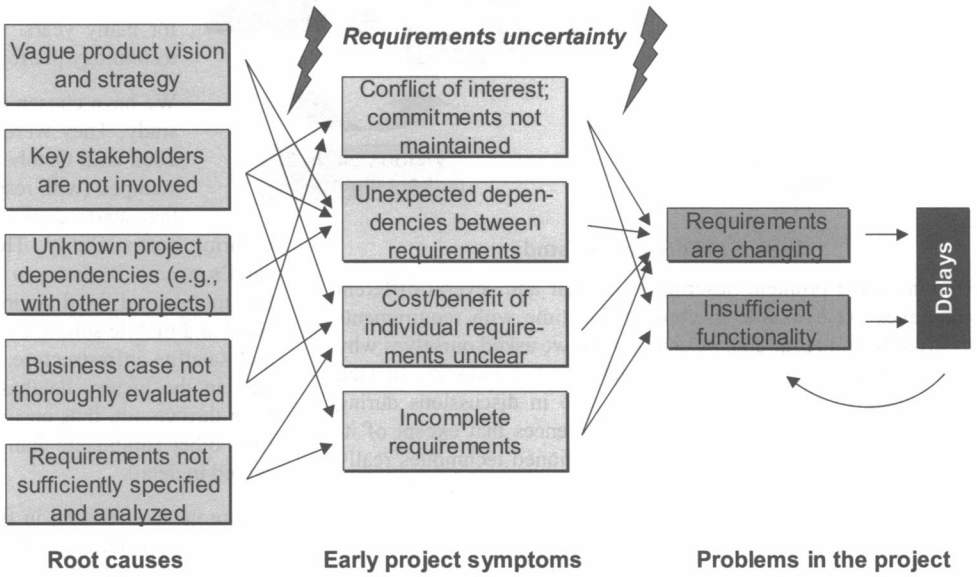

Figure 3: Identifying the root causes of delays from requirements uncertainty

Going one step further to the left in figure 3 (i.e., to the root causes for delays) we found early project symptoms, such as conflict of interest, not keeping commitments, dependencies between requirements, unclear cost-benefit assessment of individual requirements or incomplete requirements. Finally we extracted the common root causes that create requirements uncertainty, namely unknown vision and strategy, not involving stakeholders or customers, unclear dependencies, the business case not being evaluated and requirements not being sufficiently specified and analyzed. These empirical findings are supported by related product management research [13,17].

We found in the interviews and assessments of projects that there is not a single root cause that could be cured by a straightforward measure! For instance, if the main root cause was having no vision, that could be cured with working on portfolio management, roadmapping and creating product strategies. But as they are almost equal in impact, we found it more useful to treat the early project symptoms. The advantage is that unlike the root causes *before* start of a project, the early project symptoms can be treated by strengthening entry conditions *within* a single project. It's a pragmatic approach to take the individual project (and then the set of all projects in an organization) as the vehicle to implement a concrete change, rather than the "organization" or the "management".

Following our own experiences over past years and published best practices [3,4,8,13,17], we selected four concrete techniques to

treat one or more of the early project symptoms (figure 4), which we coin the "reference process" for this study:

1. Core team with empowered stakeholders
2. Product life-cycle with early gate reviews
3. Requirements are online accessible and traceable
4. Requirements evaluation from external perspective

Can we claim completeness of the four techniques? Are they the "correct" techniques? Both are hardly possible to prove in an industrial setting. What makes us sure of being on the right track is the related evidences from product management body of knowledge and the good results in our own product and solution development.

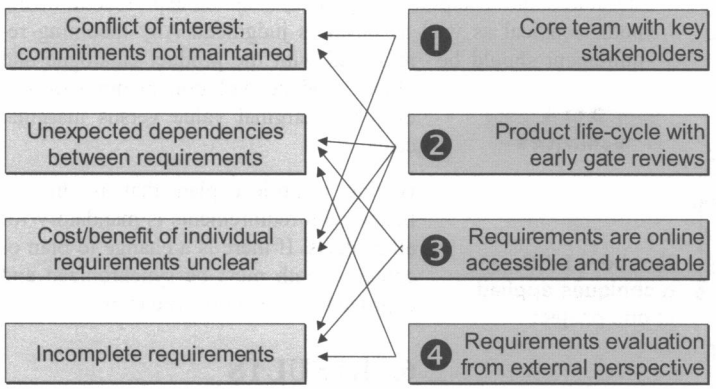

Figure 4: Early project symptoms are addressed with dedicated measures

Around this research and experiences we coined the following two hypotheses:

H1: Using only one or two techniques (from the set of four techniques above) for dealing with requirements uncertainty has no impact on project delays.

H2: A combination of three or four such techniques (from the set of four techniques above) reduces delays significantly.

We evaluated these two hypotheses by looking into 246 projects finished between Jan. 2002 and Oct. 2003 from all business areas of Alcatel. We looked specifically on delays and found that 35% of those projects were on time, 56% were in time or up to 10% late, 33% were over 20% late and 13% were over 50% late.

Often it's claimed that project size heavily influences project delays. We found that the observed results are independent of project size – within the range of project size that we had in our set. A Pearson rank correlation showed r=31% which indicates no relationship between size and delays. A double-sided Chi-square test evaluated whether usage of the four techniques would relate to the project size. We could accept the homogeneity hypothesis on a significance level of α=1%. Usage is not depending on project size. We also evaluated whether some of the four techniques would be used in correlation with each other and found that they were used independently at the time when the projects were executed. A double-sided Chi-square test looked into relationship of the four techniques. We could accept the homogeneity hypothesis on a significance level of α=5% (or below) for each single pair of the 4 techniques, indicating that they are not correlated in usage.

We now evaluate the two hypotheses.

H1: Using only one or two techniques (from the set of four techniques above) for dealing with requirements uncertainty has no impact on project delays. A double-sided Chi-square test accepts the homogeneity hypothesis on a significance level of α=1%. There is no impact on delays when using only one or two of the four techniques.

H2: A combination of three or four such techniques (from the set of four techniques above) reduces delays significantly. A double-sided Chi-square test rejects the homogeneity hypothesis on a significance level of α=1%. There is a significant impact on – reducing – delays when using three or four of the techniques! The amount of projects delivered in due time is improved by twenty percentage points when using three or four of the mentioned techniques (projects being ≤5% late: from 45% to 63%; projects being ≤10% late: from 56% to 77%). Figure 5 shows the overall distribution of projects in our study and how the delays relate to using none, one or several of the techniques.

5. PRACTICAL EXPERIENCES

When it became obvious that using several of mentioned techniques has so heavy impact on reducing delays, we looked into concrete implementation and deployment of these changes. Introducing the changes to a regular project flow is not difficult. Figure 6 shows how the changes are integrated into an existing project flow. We have simplified the activity diagram to primarily show the early project stages between inception and elaboration. Project start is at the bottom of the picture. All four techniques are included between the original business case (or vision statement) and project launch. They are tangible and can easily be checked in each single project. Introduction is not difficult, however needs some training. Let us summarize for each of the techniques some lessons learned.

5.1 Core team with empowered stakeholders

Three roles must be present in this core team, namely a product manager, a marketing manager and a technical project manager. They represent not only the major internal stakeholders in product or solution development, but also sufficiently represent the sales and customer perspective. This core team must have a clear mandate to "own" the project. We found that if such core team is available but not baselining underlying commitments and later measured on achieving them, it is of no value.

5.2 Product life-cycle with early gate reviews

The product life cycle must be mandatory for all projects. This implies that it's sufficiently agile to handle different types of projects. Standardized tailoring of the life cycle to different project types with predefined templates or intranet web pages simplifies usage and reduces overheads. Its mandatory elements must be explicit and auditable. Some online workflow support facilitates ease of implementation and correctness of information. Gate reviews (decision reviews) must be well prepared. They must not result in lengthy meeting, but are rather prepared with online checklists so all attendees are prepared and can decide in short

time the go/no go for next phase. Project information should generally be available online.

5.3 Requirements and project information are online accessible and traceable

By definition all requirements must be accessible online. Different tools can be used, starting with simple spreadsheets. We have seen agile project using one spreadsheet to manage the entire project. Such a spreadsheet has all requirements, their status, effort, responsibles and mapping to increments, test cases and work products. Reporting can be generated directly from such spreadsheet. For bigger projects we recommend online accessible vaulting systems to trace requirements to work products. A requirements database helps with this effort. Contents include requirements, implementation status of each requirement, priorities, estimated cost, value assessment, mapping to releases – especially future releases to communicate the roadmap –, relationships between requirements, and links to related implementation and test details.

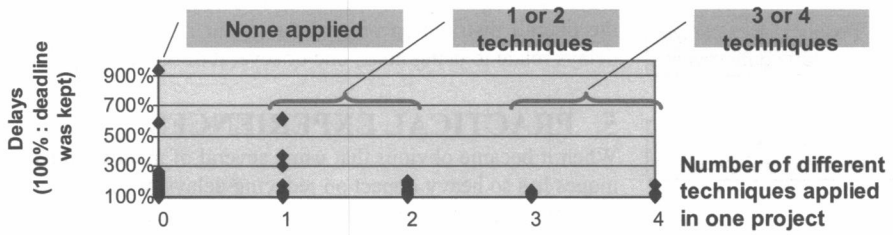

Figure 5: Impact on delays when using none, one or several of mentioned techniques

Data quality of project information and requirements lists is key. A minimum quality assurance is necessary to check for completeness and consistency of requirements and the traceability to work products. Inconsistencies and errors in requirements are often found best by testers because they think in terms of testability. If there are inconsistent or vague requirements, they should be corrected on the spot. If detected during the project, it's a requirements change that has to be approved by the core team. Project information builds an online accessible history database to base further impact analysis and project planning upon.

Properly expressed requirements form a high-level abstraction of the functional and nonfunctional behavior of the product. Formalizing such a description helps in identifying reusable aspects of systems at a level independent of any particular solution or component structure.

In product-line scenarios with many variants, requirements must be explicit per release and still linked with each other across the entire family of domain and application projects. To provide access to what solutions already exist for a given need, it's mandatory to keep all requirements and their variants in one database. For managing requirements, features and portfolio, we use interacting tools. A tailored commercial tool should be used in such scenario for the feature catalogue, traceability and release mapping.

5.4 Requirements evaluation from external perspective

Requirements must be evaluated by the entire core team to ensure that different perspectives are considered. While each single requirement must be justified to support the business case and to allow managing changes and priorities, a Pareto evaluation is recommended to focus on where it makes sense. Underlying financial figures must be correct. This holds for both sides of the equation, cost and value. Often business cases are flawed on the value side and never followed through to see if a single requirement actually contributed as much to value creation as was expected by those who asked for it.

Impact analysis is based on requirements, as well as priority setting and portfolio management. What are the requirements? How do they relate between markets and correlate with each other? What is their impact? What markets have asked for it and for what reason? Are they necessary for a solution or just inherited from an incumbent approach perhaps becoming obsolete in meantime? To address these questions requirements must be documented in a structured and disciplined way. They must be expressed allowing both technical as well as business judgment. Any incoming requirement should be reviewed with the product catalogue and global product evolution in mind to also evaluate marginal value versus marginal costs.

Having a project plan that is directly linked with requirements is mandatory for all projects. If there is a change to plan or contents, both must be synchronized and approved by the entire core team.

6. RESULTS

We have analyzed in this paper how to best cope with requirements uncertainties. A certain degree of uncertainty and thus change is part of any project risk, which determines the value proposition (after all, who would pay for zero risk). We looked into techniques that address early project symptoms indicating that the amount of uncertainties drives the project into heavy delays. Four process elements have been selected and were evaluated with an experimental field study:

1. Core team with empowered stakeholders
2. Product life-cycle with early gate reviews
3. Requirements are online accessible and traceable
4. Requirements evaluation from external perspective

These four process elements are easy to implement and more readily accepted than, for instance, prototyping. They can be introduced stepwise with little marginal effort. They apply to early project symptoms, thus allowing to see the benefits early in the project. They are tangible and can be formally introduced to projects during the launch period, thus reducing the change impact (i.e., no big bang). Practitioners in engineering, product management and marketing accept them because they yield results and stimulate empowered project teams.

We found that a combination of those techniques effectively mitigate the risk of delays resulting from requirements uncertainties. Using three or four from the set of four techniques significantly reduces the delays of a project. Using only one or two techniques, as was often propagated in previous studies, has no influence on delays. The techniques go well beyond engineering and address a

major impact of requirements management, namely that it is a concerted activity of an empowered team covering product management, engineering and marketing throughout the entire project or product life cycle.

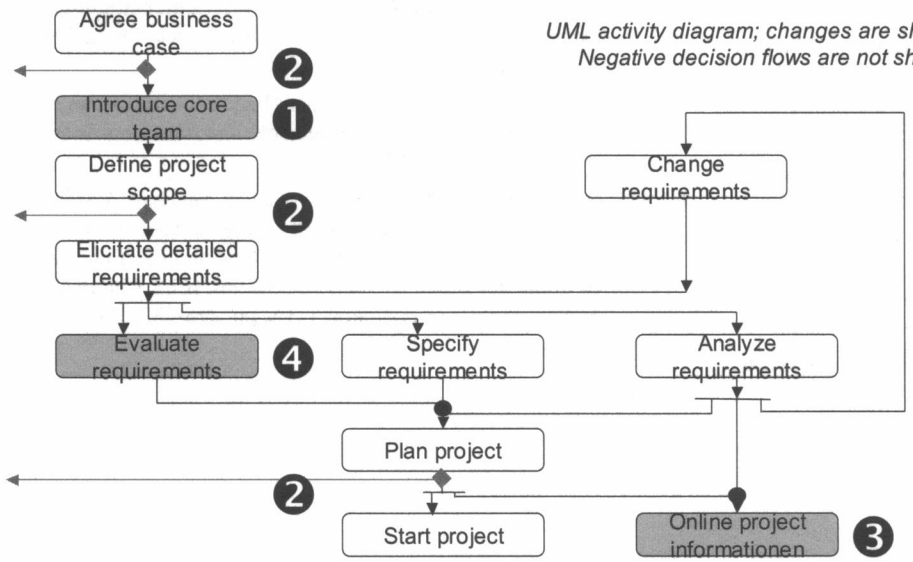

Figure 6: Practically introducing the changes into the project flow

Next steps in better managing requirements uncertainty will address the following topics:

- Better predict changes to requirements on an individual level.
- Improving the evaluation of requirements within a business case from a portfolio management perspective.
- Fostering product management best practices by a more formalized qualification of all product managers.
- Reducing the time to project start (e.g., what is good enough for requirements analysis? Which level of change is feasible to cope with in given scenarios/markets?).
- Modelling the influences from training, product age, architecture, variability, technology or markets/customers on requirements uncertainty.
- Introducing knowledge management techniques for simulative collection, evaluation, modeling and retrieval of requirements and underlying decisions.

7. ACKNOWLEDGMENTS

Our thanks to the many Alcatel product management and engineering teams that shared their best practices and that helped in setting up product life-cycle management.

8. REFERENCES

[1] Boehm, B.W.: A Spiral Model of Software Development and Enhancement. IEEE Computer, Vol. 21, No. 5, 1988.

[2] Bush, D. and A.Finkelstein: Requirements Stability Assessments Using Scenarios. Proc. 11. Int. Conf. On Req. Engineering RE03, IEEE Computer Society Press, Los Alamitos, USA, pp. 23-32, 2003.

[3] Chrissis, M.B., M.Konrad and S.Shrum: CMMI. Guidelines for Process Integration and Product Im-provement. Addison-Wesley, Boston, USA, 2003.

[4] Costello, R. and D. Liu: Metrics for Requirements Engineering. Journal of Systems and Software, Vol. 29, 1995.

[5] Davis, G.B.: Strategies for information requirements determination. IBM Systems Journal, Vol. 21, No. 1, pp. 3-30, 1982.

[6] DeMichelis, G., E. Dubois, M. Jarke, F. Matthes, J. Mylopoulos, M. Papazoglou, K. Pohl, J. Schmidt, C. Woo, E. Yu: Cooperative Information Systems: A Manifesto. In: Cooperative Information Systems: Trends and Directions, M. P. Papazoglou and G. Schlageter (eds), Academic Press, 1997. pp. 315-363.

[7] Doernemann, H.: Tool-Based Risk Management Made Practical. Proc. Int. Conf. On Req. Engineering RE02, IEEE Computer Society Press, Los Alamitos, USA, 2002.

[8] Doerr, J., B. Paech and M. Koehler: Requirements Engineering Process Improvement Based on an Information Model. Proc. Int. Conf. On Req. Engineering RE04, IEEE Computer Society Press, Los Alamitos, USA, 2004.

[9] Ebert, C., R. Dumke, M. Bundschuh and A. Schmietendorf: Best Practices in Software Measurement. Springer, New York, Heidelberg, 2004.

[10] Free On-Line Dictionary of Computing. http://foldoc.doc.ic.ac.uk/foldoc/foldoc.cgi?query=requirement&action=Search. Queried on 07.July 2004.

[11] Galetta, D.F. and M. El Loudadi: L'effet de l'incertidue et des strategies de determination des besoins de l'utilisateur sur les projets d'informatisation. Canadian Journal of Administrative Systems. Vol. 12, No. 1, pp. 56-76, 1995.

[12] Giesen, J. and A.Voelker: Requirements Interdependencies and Stakeholder Preferences. Proc. Int. Conf. On Req. Engineering RE02, IEEE Computer Society Press, Los Alamitos, USA, 2002.

[13] Gorchels, L.: The Product Manager's Handbook : The Complete Product Management Resource. McGraw-Hill, second edition, New York, 2000.

[14] Hickey, A.M. and A.M. Davis: Elicitation Technique Selection: How Do Experts Do It? Proc. 11. Int. Conf. On Req. Engineering RE03, IEEE Computer Society Press, Los Alamitos, USA, pp. 169-178, 2003.

[15] IEEE Standard 610.12-1990. IEEE Standard Glossary of Software Engineering Terminology. IEEE, New York, NY, USA. ISBN 1-55937-067-X (1990).

[16] Jones, C.: Software Assessments, Benchmarks, and Best Practice. Addison Wesley, Reading, 2001.

[17] McGrath, M.E.: Next Generation Product Development: How to Increase Productivity, Cut Costs, and Reduce Cycle Times. McGraw-Hill, New York, 2004.

[18] Naumann, J.D., A.M. Jenkins, and J.C. Wetherbe: The Information Requirements Determination Contingency Model: An Empirical Investigation. University of Minnesota, Minneapolis, MIS Research Center, Report 83-15, 1983.

[19] Parnas, D.L.: Designing Software for Ease of Extension and Contraction. IEEE Trans. On Software Engineering. Vol. 5, No. 2, 1979.

[20] Ramesh, B. and M.Jarke: Toward Reference Models for Requirements Traceability. IEEE Transactions on Software Engineering, Vol. 27, No. 1, Jan. 2001

[21] Royce, W.: Software Project Management, Addison Wesley, Reading, 1999.

[22] Saarinen, T. and A.Vepsalainen: Managing the Risks of Information Systems Implementation. European Journal of Information Systems. Vol. 2, No. 4, pp. 283-295, 1993.

[23] Schmid, K.: A Comprehensive Product Line Scoping Approach and its Validation. Proc. 24. Int. Conf. On Software Engineering ICSE'02, IEEE Computer Society Press, Los Alamitos, USA, pp. 593-603, 2002.

[24] The Standish Group International Inc.: CHAOS Chronicles v3.0. http://www.standishgroup.com/chaos/toc.php. West Yarmouth, USA, 2003.

[25] Strens, M.R. and R.C. Sugden: Change Analysis: A Step towards Meeting the Challenge of Changing Requirements. Proc. of the IEEE Symposium and Workshop on Engineering of Computer Based Systems (ECBS'96), IEEE Computer Society Press, Los Alamitos, USA, pp. 278-283, 1996.