Managing Requirements Uncertainty in Engine Control Systems Development

Andy J. Nolan¹, Silvia Abrahão² Paul Clements³ Andy Pickard¹

¹Rolls-Royce
SIN C-3, Rolls-Royce plc, PO Box 31
Derby DE24 8BJ, England
{Andy.Nolan, Andrew.C.Pickard}@Rolls-Royce.com
²ISSI Research Group, Department of Computer Science
Universitat Politècnica de València
Camino de Vera s/n, 46022 Valencia, Spain
sabrahao@dsic.upv.es
³ Software Engineering Institute
Carnegie Mellon University
4500 5th Avenue, Pittsburgh, PA 15213-2616, USA
(412) 268-5800
clements@sei.cmu.edu

Abstract—In the development of complex systems the requirements for the system will almost always remain uncertain late into the software development. In gas turbine engine control systems at Rolls-Royce, typically 50% of requirements will change between Critical Design Review and Entry into Service. Ignoring or not planning for requirements uncertainty will cause scrap and rework that will manifest later in the project. This paper evaluates the impact of not managing these uncertainties and describes how Rolls-Royce uses Requirements Uncertainty Analysis to reduce this impact. The paper summarises the findings from an extensive Six Sigma study into requirements uncertainty and provides an overview of the technique now used to identify and monitor uncertainty through a project life. The return on investment of this technique has been between 100:1 and 500:1.

Keywords: Requirements, Uncertainty, Complex Systems.

I. INTRODUCTION

Anything that is complex will tend to evolve during development. However, if a project applies a rigid development approach and fails to recognise uncertainty and the need for evolution, it can actually drive additional costs into a project.

A common reason for project problems is insufficient management of changing requirements during all stages of the project life. One of the root causes for changing requirements is requirements uncertainty [4]. In fact, requirements uncertainty has long been recognized as a major risk factor for software development projects (e.g., [3], [8], [5]). Therefore, it is vital to manage these uncertainties through the project life and to either eliminate the uncertainty or allow for it in cost-effective ways. However, there are few studies that provide a predictive view of requirements uncertainty (e.g., [1] [2]). From studies conducted at Rolls-Royce, we have shown that over 80% of requirements uncertainty can be anticipated at project launch making uncertainty a factor that can be anticipated and managed.

This paper presents an experience report on the application of risk analysis techniques for managing requirements uncertainty. The analyses technique described in this paper adds around 1 minute extra effort for analysing each requirement but has been shown to reduce Scrap & Rework on a project from an average of 50% down to a level below 5%. The return on investment for uncertainty analysis and mitigation can be between 100:1 and 500:1, making it one of the most cost-effective improvements a project can apply.

This paper is organized as follows. Section II presents the context in which the technique for Requirements Uncertainty Analysis presented in this paper has been developed. Section III provides an explanation about the importance of managing requirements uncertainty at Rolls-Royce. Section IV presents the principles of the Requirements Uncertainty Analysis technique as well as some observations gathering from its use. Section V presents four attributes used for predicting the uncertainty of a requirement and its impact to the project. Finally, section VI presents conclusions and future work.

II. CONTEXT: ROLLS-ROYCE ENGINE CONTROL SYSTEMS

Although we believe the results in this paper are broadly applicable, the supporting data for our conclusions was gathered in a specific context, which we will describe in this section.

The Control Systems department at Rolls-Royce is responsible for the Engine Electronic Controllers (EECs) for a range of small and large gas turbine engines, for the aerospace industry. The EEC contains a significant amount of software that is designed to control the engine (see Figure 1), as directed by the pilot, in a way that is safe for the engine, safe for the aircraft, fuel-efficient, component life efficient and environmentally efficient. EEC software is developed to DO-178B Level-A standards [13] for safety-critical software.

Figure 1. Rolls Royce Trent 900 engine used to power the Airbus A380 – the control software is in excess of 200,000 lines of code.
The development of a new engine can take up to 5 years and will be highly evolutionary. The electronics, the engine, and the airframe will evolve and mature through the life of a project causing new functionality and changes to emerge. Historical data shows that between the point of Critical Design Review (a system concept review gate) and Entry into Service, a project will spend approximately 50% of its cost on evolutionary work rather than new product development. The evolutionary work will arise in the form of formal change requests raised either by our customers or by the Control Systems department. Dealing with changing requirements and uncertainty is therefore a critical issue.

III. AN OVERVIEW OF REQUIREMENTS UNCERTAINTY AT ROLLS-ROYCE

A Six Sigma project was conducted at Rolls-Royce to look at improving our estimation capability. The study showed that requirements uncertainty and scope creep were ranked as the two most dominant reasons for poor estimation i.e., overspending relative to the initial estimate. Rolls-Royce is not the only organization to have uncertainty and scope creep.

Data at Rolls-Royce shows that uncertainty changes over time, not just in magnitude (quantity) but also in form. In general, at the start of a project, there is more uncertainty than near the end – but not always. Even late into a project’s life there are residual uncertainties that manifest as surprises and late changes. For example, unforeseen emergent system behaviour has to be addressed; the interface to the aircraft is not quite as specified and so on.

Figure 2 shows the percentage of requirements change at key project phases. The shaded area represents the full measured range of requirements uncertainty a project can expect at each milestone.

Figure 3 shows the average relative cost of a software change at various phases of a project. All costs are shown relative to the cost to find and remove an error during the review process. A similar chart for hardware design shows an order of magnitude change in cost with each key project phase.

By combining Figure 2 and Figure 3 we get Figure 4 which shows an average net impact of 50% Scrap & Rework, meaning that the manifestation of unmitigated uncertainty accounts for half the cost of a project.

A. The Impact of Uncertainty

Any project that proceeds on the basis of certainty will undergo “unexpected” changes later in the life of a project. Planning only on the basis of success and not accommodating risks or uncertainty is likely to lead to failure [12]. Figure 3 is based on data from Rolls-Royce and shows the average relative cost to make a software change at various phases of a project. All costs are shown relative to the cost to find and remove an error during the review process. A similar chart for hardware design shows an order of magnitude change in cost with each key project phase.

B. Managing Uncertainty

It is tempting to believe that the uncertainty is driven by factors outside the project’s control for example; the change is introduced by the customer. An analysis of the source of change for engine control systems (an internal to Rolls-

...
Royce study of all civil and defence projects for the last 16 years) shows that only 16% of change is driven by the customer. Deeper analysis also showed that of the 16% customer driven changes, a proportion could have been anticipated by either us or by our customers.

84% of changes were self generated. Bell Labs and IBM conducted studies that determined that 80% of all product defects were incorporated during the requirement definition phase \[6\]. This is in line with the results obtained by Mawby and Stupples \[10\] who found that project overruns are usually caused by rework generated within the project.

If an organisation has a higher level of unexpected customer changes, that may be a sign that it has not done its systems engineering and requirements gathering correctly. Being close to the customer can help anticipate where requirements are likely to change or new requirements appear.

Technical Risk Management \[11\] is one of the methods that can be used by project managers to identify and potentially mitigate the impact of requirements uncertainty. Given that there is a high level of uncertainty in a Rolls-Royce project and that the effects of this uncertainty causes up to 50% Scrap & Rework, we would expect to see this expressed in the risk logs. An analysis of the risk logs for a range of projects within Rolls-Royce showed that less than 4% of risks were technical in nature, the remaining risks tending to focus on project and business risks.

A further analysis showed that many engineers did not use risk identification and mitigation. The engineers were not encouraged to identify and express their uncertainty – there was an implicit message in the business that engineers must always be certain, and that uncertainty was a sign of failure rather than a normal and expected occurrence to be managed.

A range of projects were studied in the business where the projects all had the same maturity (all in service), they had approximately the same processes, the same team and each project underwent a modification of the same size. In the case where the projects did not apply any Technical Risk analysis to understand the uncertainty, they experienced a 50% Scrap & Rework rate. For those projects that actively applies Technical Risk Management (to requirements and design), they had a Scrap & Rework rate below 5%. The cost benefit from Technical Risk Management was 100:1. A similar study in hardware engineering revealed a return on investment of 500:1.

These studies show that uncertainty can be understood, expressed and mitigated and requirements uncertainty can be minimised or contained.

IV. THE PRINCIPLES OF REQUIREMENTS UNCERTAINTY ANALYSIS AND SOME OBSERVATIONS

A. Uncertainty Identification

The Requirements Uncertainty Analysis technique (described in section V) involves applying Technical Risk Management principles to each individual requirement (or logical group of requirements) to understand the uncertainty in the customer (and supplier) requirements in order to control and minimize the cost of late changes. The technique involves the use of the standard risk principles by assigning \textit{probability} and \textit{impact} values to each requirement (e.g. \textit{L}=Low, \textit{M}=Medium and \textit{H}=High - see section V).

The analysis is performed initially by the systems engineer then reviewed, with independence, by the requirement reviewer. The probability of change is “estimated” by the systems engineer (based on subjective experience, data from past projects and his/her understanding of the system). The impact of change can be “estimated” by the designers and implementers of the system but in practice it tends to be estimated by an experienced systems engineer who does this.

Figure 5 presents an example of probability impact diagram showing the results of the uncertainty analysis. The colour coding implies the impact to a project: Green (bottom left) has a low risk, and red (top right) has the greatest risk.

![Probability impact diagram showing the volatile requirements](image)

Figure 5. Probability impact diagram showing the volatile requirements. The objective is to either reduce the uncertainty or reduce the impact or a combination of the 2.

B. Mitigating the Uncertainty

The objective of a systems engineer is to move off the top right area (red) by reducing the uncertainty (moving horizontally across the matrix), minimising the impact (moving down the matrix) or a combination of both approaches. The selection of a particular approach will depend on the type of risk the requirement is exposed to. For example, the risk of “Immature or late customer requirements” has a number of mitigation options.

- More customer engagement
- Does the customer have a "design style" - can you predict the requirements from previous projects working with this customer?
- Get the customer to do a Technical Risk assessment
- Propose requirements to the customer
- Look at other projects – what did they have
- Build robustness into the architecture
- Delay- wait for the requirements
- Proceed but factor late change into plans & budgets
- Model system and present to the customer
- Find the range outside which it will hurt
It is not always necessary to remove the uncertainty. It is possible to develop products that are “robust” to accommodate a reasonable level of uncertainty. Rolls-Royce has used Product Line principles [7] to develop “configurable” functions. Configurable here means that one can easily change the behaviour of a function through the use of a switch or data change, without having to re-code the logic. For example, the performance data used in the control laws of an engine is separated from the functionality, allowing the performance engineers to evolve the engine parameters without impact to the software.

When placing requirements on suppliers, rather than offering a “single-point” requirement, Rolls-Royce offers a 3-point requirement (e.g., the maximum temperature for a probe will be between X and Y, with most probable value Z).

In those cases where there is uncertainty but no clear mitigation available, the systems and software architecture can bound the requirements (or function) and isolate it from the rest of the system until the requirements become more mature.

Requirement uncertainty analysis can be used to help influence the planning process. In the example of a high uncertainty function, a project can ask for time to help mature the requirements before proceeding with development.

Once the requirements uncertainty attributes (described in section V) are calibrated to a project domain, the information can be used to estimate Scrap & Rework, the impact of late changes, the contingency a project must carry and so on.

C. Lessons Learned

It may seem daunting to apply Requirements Uncertainty Analysis to a project. A project may have thousands of requirements. Four years of using this technique has shown that it will take an engineer between 1 and 2 minutes (but typically around 1 minute) per requirement. However, there is additional effort required to mitigate the uncertainty identified.

When we retrospectively applied the technique to the hardware requirements of an existing engine (a double blind experiment), the results show that the engineer was able to correctly identify 80% of the uncertainty and that the risk identification and mitigation approach would have yielded a return on investment of 500:1. Other studies across Rolls-Royce have shown that when used by experts, 90% of the uncertainty can be identified and mitigated with a 100:1 return on investment.

When performing requirement uncertainty analysis, we found that the high-risk requirements were already known about - this analysis did not help identify these issues. A good engineer will typically know the problem areas and will have flagged these issues up to the project.

Interestingly, the sum of the medium-risk requirements added up to 80% of the uncertainty. That means that overall, the high-risk requirements had an impact on a project but not as much as all the smaller risks added together. However, project management is typically only interested in the high-risk requirements, in which case 80% of the uncertainty (and therefore Scrap & Rework) will go ignored by the project. This is consistent with the findings earlier in this paper i.e., 84% of the Scrap & Rework is self-generated.

This technique of Requirements Uncertainty Analysis helps to identify both high-risk and medium-risk requirements which can then be managed and mitigated using Technical Risk Management techniques.

V. REQUIREMENTS UNCERTAINTY ANALYSIS

This section describes the four attributes developed at Rolls-Royce for predicting the uncertainty of a requirement and its impact to the project. Although we apply these attributes to each requirement, they could be applied to a group of requirements such as for a system function.

A. Requirements Attributes

The following attributes are added to our requirements database.

- **Volatility** represents the engineer’s best judgment as to the probability the requirement will change through the course of the project. This is based on subjective judgement, past data and knowledge of the system under development. This attribute was also suggested in other studies, e.g., Lam et al. [9], who manage volatility by process control, suggest that volatility classification should capture the domain-specific nature of change in order to facilitate change estimation and reuse.

- **Impact** accounts for the degree that a change in the requirement will negatively affect a development program. Rather than the systems engineer selecting this attribute, it may be better for the developers of the system to “estimate” the impact.

- **Precedence** incorporates multiple variables to indicate Rolls-Royce’s heritage in providing solutions that address this attribute for a similar application, environment, and context of use.

<table>
<thead>
<tr>
<th>Selection</th>
<th>Volatility</th>
<th>% Volatility</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very High Volatility</td>
<td>9</td>
<td>&gt;90%</td>
</tr>
<tr>
<td>High Volatility</td>
<td>7</td>
<td>70% - 90%</td>
</tr>
<tr>
<td>Medium Volatility</td>
<td>5</td>
<td>50% - 70%</td>
</tr>
<tr>
<td>Low Volatility</td>
<td>3</td>
<td>20% - 50%</td>
</tr>
<tr>
<td>Very Low Volatility</td>
<td>1</td>
<td>10% - 30%</td>
</tr>
</tbody>
</table>

Figure 6. Volatility measurement scale

<table>
<thead>
<tr>
<th>Selection</th>
<th>Type of impact</th>
<th>Impact</th>
</tr>
</thead>
<tbody>
<tr>
<td>High Impact</td>
<td>Cost</td>
<td>&gt;20% cost impact</td>
</tr>
<tr>
<td></td>
<td>Schedule</td>
<td>&gt;4 weeks slip</td>
</tr>
<tr>
<td>Medium Impact</td>
<td>Cost</td>
<td>10% - 20% cost impact</td>
</tr>
<tr>
<td></td>
<td>Schedule</td>
<td>1 - 4 weeks slip</td>
</tr>
<tr>
<td>Low Impact</td>
<td>Cost</td>
<td>&lt;10% cost impact</td>
</tr>
<tr>
<td></td>
<td>Schedule</td>
<td>&lt;1 week slip</td>
</tr>
</tbody>
</table>

Figure 7. Impact measurement scale
Experience within Rolls-Royce has shown that even if you cannot anticipate the risk, if it is a novel system requirement, there will be a lot of evolution required to mature it.

**Time Criticality** provides sensitivity to time critical requirements. This complements the prioritization of work to allow the engineer to work on items critical for the current phase as opposed to items not required until later phases of the project. This attribute challenges the view that all requirements do (and have to) mature at the same rate. This attribute informs an architect that they need to isolate the impact of this late maturating requirement.

<table>
<thead>
<tr>
<th>Selection</th>
<th>Weight</th>
<th>Precedence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>9</td>
<td>No experience of concept, or environment. Historically volatile</td>
</tr>
<tr>
<td>Medium</td>
<td>3</td>
<td>Some experience in related environments. Some historic volatility</td>
</tr>
<tr>
<td>High</td>
<td>1</td>
<td>Concept already in service. Low historic volatility</td>
</tr>
</tbody>
</table>

**Maturity Index (MI):** The maturity index is a value associated with a system function, not with each individual requirement. A function that is 100% mature is defined as mature and free from any further anticipated changes.

\[
MI = \frac{\sum (V \times TC)}{\text{Total # Requirements} \times 81} \times 100
\]  

where,

- \( V \) = Volatility score (1 to 9)
- \( I \) = Impact score (1 to 9)
- \( P \) = Precedence score (1 to 9)
- \( TC \) = Time Criticality score (1 to 9)
- 81 = 9 * 9 (the maximum score for V and TC giving a value for M in the range 0 – 1)

and the sum is for the product of these factors for all of the requirements in the document.

**Proportional Risk Index (PRI):** For an understanding of the risk of an individual requirement, the proportional risk index is recommended. PRI is intended to prioritize the engineer’s tasks in regard to the document.

\[
PRI = \frac{RI}{\sum RI}
\]  

**Example Analysis Results**

This section illustrates some examples from the application of Requirements Uncertainty Analysis and applying the equations shown in the previous section.

**Plotting Proportional Risk:** Figure 10 shows an example of plotting requirements risk. Requirements system function for a single project are listed across the X-axis of the chart and the uncertainty of each on the Y-axis. The systems engineer could annotate this chart as shown with bands of uncertainty to present their risks to project management.

**Reporting Requirement Maturity Index.** The Maturity Index (Figure 11) was calculated for a range of system functions. It shows that there are
some functions which can be started at little risk and other functions requiring more Technical Risk Management to improve their overall uncertainty. Alternatively, the maturity index can be tracked over time and plotted as shown in Figure 12.

<table>
<thead>
<tr>
<th>System Function</th>
<th>Maturity Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function 1</td>
<td>90%</td>
</tr>
<tr>
<td>Function 2</td>
<td>78%</td>
</tr>
<tr>
<td>Function 3</td>
<td>54%</td>
</tr>
<tr>
<td>Function 4</td>
<td>50%</td>
</tr>
<tr>
<td>Function 5</td>
<td>70%</td>
</tr>
<tr>
<td>Function 6</td>
<td>59%</td>
</tr>
<tr>
<td>Function 7</td>
<td>64%</td>
</tr>
<tr>
<td>Function 8</td>
<td>12%</td>
</tr>
<tr>
<td>Function 9</td>
<td>34%</td>
</tr>
<tr>
<td>Function 10</td>
<td>60%</td>
</tr>
</tbody>
</table>

Figure 11. Maturity index per system function

Figure 12. Maturity index plotted over time

VI. SUMMARY

This paper presented the results from an extensive Six Sigma study into requirements uncertainty conducted at Rolls-Royce. The main findings of the study are as follows:

- Uncertainty is certain but if no effort is made to control uncertainty, then it will manifest later as Scrap & Rework.
- Most Scrap & Rework (or a significant amount of it) can be self-generated by not managing your inherent uncertainty.
- Contrary to expectations, changes in customer requirements are not a major driver of Scrap & Rework - most is internally generated by the development team.
- Systems Engineering and Risk Management are critical in understanding and controlling the sources of Scrap & Rework.
- The return on investment for the use of Requirements Uncertainty Analysis can be between 100:1 and 500:1. As future work we plan to extend this technique for its use with software product lines. This will help architects to better define the scope of product lines by analysing not only the risks associated with the known variation but also the risks associated with the unknown variation in a product line.

ACKNOWLEDGMENT

This research is supported by the MULTIPLE project (with ref. TIN2009-13838) funded by the MICIN, Spain.

REFERENCES