THE ROLE AND DECOMPOSITION OF DELIVERY QUALITY IN A FAILING PROJECT: A CASE STUDY

Dirk J. van Ekris
Delta Pi
Postbus 214, 6920 AE Duiven, The Netherlands
J.vanEkris@Delta-Pi.nl

ABSTRACT
Academics and practitioners both recognize that, besides project outcome, the way a project is conducted is an important factor in generating repeat business. This aspect of project management is called Delivery Quality and it focuses on the interactions between project team and customer organization.

When a project delivers the expected product on time and on budget, Delivery Quality is an essential part for turning a merely satisfied customer into a returning one. However, many projects fail to meet customers’ expectations, posing the question if Delivery Quality is still relevant and the underlying decomposition is still valid. The case study shows that Delivery Quality is still relevant to the project when product quality clearly is lacking: Delivery Quality might be a requirement for being able to rescue the project.

The case also shows that the decomposition of Delivery Quality into underlying factors has issues: the factor Transparency is clearly missed, and the factors Responsiveness and Partnership are too narrowly defined. Especially the lack of the factor Transparency is considered critical, since in this case the lack of Transparency led to a quite abrupt abortion of the project.

KEY WORDS
Project management, project success, customer satisfaction, delivery quality

1. Introduction

Successful projects are vital for ICT development organizations to stay economical viable and to provide opportunities to retain customers for future projects. Although the topic of project success is frequently discussed, the underlying definition of when a project is considered successful isn't generally agreed upon [1, 2, 3, 4]. Traditional narrow approaches focus on delivering the right product on time within budget [5], where broader definitions of project success also explicitly include customer satisfaction or rebuy intentions [3, 4, 6, 7, 8], recognizing the commercial value of being able to use past projects as a stepping stone for new projects.

Delivering the right product on time within budget does not suffice for attaining a level of customer satisfaction that triggers rebuy intentions [9, 10, 11, 12]. To achieve that level of customer satisfaction, Delivery Quality becomes relevant [12, 13]: the way the project is conducted and delivers the product to the customer.

Although Delivery Quality is relevant for turning satisfied customers into repeat customers, a significant portion of the ICT projects fails to deliver what is promised on time within budget [14]. This poses the question what the role and composition of Delivery Quality is in the context of a failing project.

We think that Delivery Quality remains relevant for project success, even if the project outcome does not meet the customers’ expectations. Our first hypothesis is that Delivery Quality is relevant in a mission critical ICT project that clearly misses important explicit product goals. This does not mean that Delivery Quality can directly compensate for an unacceptable product, but maintaining high levels of Delivery Quality might be able to prevent project management escalations that might lead to project abortion.

To make Delivery Quality more tangible and manageable, it can be decomposed into several underlying factors. One such decomposition is the extended SERVQUAL model, which has been tested in a context where the product clearly fulfilled customer expectations and was delivered on time and within budget [15]. Leaving this context introduces the question if this decomposition of Delivery Quality remains valid. Our second hypothesis is that this decomposition of Delivery Quality remains valid, despite a clearly failing product. This implies several sub-hypotheses:

a. The factors are sound and complete, i.e. the factors are a meaningful and sufficient way to describe the resulting opinion of the customer;
b. The definitions of factors are stable and remain unchanged by the product failing;
c. There is no “cross contamination” of the lack in product quality, i.e. the lack of product quality is not unjustly projected upon the Delivery Quality.

In this article we describe an empirical test of the effects of product failure on Delivery Quality. First we review the literature regarding the role and decomposition of Delivery Quality. Then we test our hypotheses in a case study. Finally, we analyse the implications of the hypothesis for practitioners and do suggestions for further research.
2. Related Literature

In this section we first look at the definition of Delivery Quality. After this we look at the importance of Delivery Quality when it comes to project success and customer satisfaction. We conclude this chapter with the decomposition of Delivery Quality.

2.1 The concept of Delivery Quality

There are many definitions of project success [1, 2, 3, 4]. Where some have a narrow focus on delivering the right product on time within budget [5], others take a broader scope by including customer satisfaction or rebuy intentions [3, 4, 6, 7, 8]. Industry research does indicate that the narrowly focused definition still dominates among practitioners, despite the broader definitions being available for decades [1]. In a widely cited paper on project success, Baccarini investigated the various definitions used across literature and identified “stakeholder satisfaction with the project management process” as one of the six elements of commonly found in project success definitions [16], noting that the customer is one of the most important stakeholders. Besides identifying this common element among several definitions, there is no explicit definition given, making further research difficult. In project management literature several other authors [17, 18] do recognize satisfaction with the process as a success factor in some way or another, but also don’t define it explicitly.

For the definition of "customer satisfaction with the project management process" we turn to services marketing. In a very widely cited paper on factors affecting customer satisfaction, Grönroos recognizes the way a project outcome was obtained as being relevant for satisfaction and rebuy intentions [19]. Although there is a clear similarity between the two concepts, Grönroos' definition is broader: it explicitly includes many softer factors like appearances, behavior of employees, what was being said and how it was said. Other authors also stressed this importance of inter-human interaction between supplier and customer [12, 20, 21, 22]. In a sense the satisfaction with the process represents the non-monetary costs of a project for the customer: the time, physical and psychic costs he/she had to endure to get the desired project outcome [13, 23].

In this article we adopt the term “Delivery Quality”, being defined as the way the outcome of a project was obtained [12]. It represents the non-monetary costs like the time, physical and psychic costs to get the desired project outcome. Aside from the project management process itself, it also includes the interactions with the suppliers' employees, the effectiveness of these interactions, the flexibility of the project, the appearance of employees and the appearance of project artifacts like documentation.

2.2 The importance of Delivery Quality

The importance of Delivery Quality is dependent on the satisfaction level a developer wants to achieve and the specific role of the stakeholder. In this section we explore both.

The desired satisfaction level of a stakeholder has significant consequences for both rebuy behaviour and the factors that contribute to attain that satisfaction level. Much research treats satisfaction as a binary concept: the customer is either satisfied or dissatisfied, which obscures many relations. When adapting a more continuous approach, Heskett et al. [10] discovered the following relation between satisfaction and retention levels:

Figure 1 shows the relation between the satisfaction level and its effect on rebuy intentions. Heskett et al. recognize three discrete levels of customer satisfaction related to rebuy intentions:
- The "zone of defection" where the customer is unsatisfied and would defect if possible;
- The "zone of indifference" where the customer is satisfied but has no specific affection with the supplier;
- The "zone of affection" where the customer is extremely satisfied and has a strong preference for the supplier or even rebuy intentions.

The distinction between the "zone of indifference" and the "zone of affection" is important since it is the difference between a merely satisfied customer and a returning one: the latter allows the capitalization of satisfaction into new projects. It must be concluded that when investigating customer satisfaction, it is important to distinguish the different levels of customer satisfaction, since it has effects on rebuy intentions.

Aiming for the "zone of affection" has consequences on the activities that are needed to be performed by the developer. The general consensus is that not delivering the right product is an important dissatisfier [3, 13, 16, 24 25, 26, 29, 30]. However, delivering a high quality product is not enough to reach the "zone of affection" [3, 9, 10, 11, 12]. Reaching this level of customer satisfaction requires a developer to deliver the right product and
exceed on the (customer perception of) Delivery Quality [9, 10, 12, 13].

The factors that contribute to customer satisfaction are both time and role dependent [7, 16, 21, 25, 26, 27]. For an end user, the project outcome (in relation to the business objectives) determines the satisfaction [25, 28, 29]. Acquisition/project management team members have a different perspective on projects than end users: where end-users and their management place significant emphasis on the project outcome, an acquisition/project-management team puts more emphasis on the way a project is conducted and how defects are handled [17, 18, 21]. This suggests that for specific roles the Delivery Quality might be an important factor.

Based on the above, we have to conclude that when a developer aims for rebuy intentions by members of an acquisition/project management team, he not only has to deliver the right product, but he has to exceed in the area of Delivery Quality. Empirical evidence supports this; two years after delivering the right product on time and within budget, the customers’ project team members still were not satisfied to the level of rebuy intentions, pointing to the lack of Delivery Quality as their most important reason [12]. So when delivering the right product, for some stakeholders Delivery Quality makes the difference between a merely satisfied customer and a returning one.

Our first hypothesis questions the relevance of Delivery Quality when this product is clearly lacking in quality. Quantitative studies [24, 29, 30] each independently show that when a project does not deliver the right product it can not be compensated by Delivery Quality. However, these studies completely focus on end user satisfaction in a post-project evaluation. They did not include the acquisition/project management teams, especially during the development phase of a software product where such a team may have to decide between allowing repairs or aborting the project. Due to the role and time dependency of factors affecting satisfaction this might lead to different results if the satisfaction level of the acquisition/project management team would have been investigated during the project. Research suggests that for this particular role service recovery [9] and defect handling [21] are significant factors affecting satisfaction. Some research even suggests that a very good service recovery improves the customer relation because it proves the commitment the developer has towards his customer [9, 23]. Research also indicates that the project phase has significant influence on the factors determining satisfaction [25], making post-project studies of factors less relevant for determining in-project factors. This leads us to conclude that the relationship between Delivery Quality and the decision by an acquisition team whether a project/product is salvageable is unknown. Deciding to allow for repairs does have a significant influence on product quality and thus end-user satisfaction, leading to the conclusion that the practical effect of Delivery Quality on a project outcome and customer satisfaction when the product has significant defects is unknown.

2.3 Decomposing Delivery Quality

To make Delivery Quality more manageable, it is necessary to decompose Delivery Quality into underlying factors, similar to ISO/IEC 25010 [31] decomposing software quality into underlying terms. The primary goal of this decomposition is to be able to make Delivery Quality more tangible, and to be able to structure interviews with customers about their perceptions of a developer, allowing a more thorough analysis.

The most commonly used and widely accepted model for service quality is SERVQUAL, developed by Parasuraman, Berry and Zeithaml [32] for business-to-consumer services. In later work, this model has been adapted to the business-to-business services [33]. This business-to-business model recognizes different aspects of service quality:

- **Reliability**, The developer keeps his promises;
- **Empathy**, The developer understands the industry, actively thinks about the solution at the customer;
- **Responsiveness**, The developer responds quickly to questions and change requests;
- **Assurance**, The developer has a good reputation and shows its technical experience;
- **Tangibles**, documents and materials look good.

The SERVQUAL model also has been applied to the ICT industry [37], which has been criticized for several reasons:

- SERVQUAL completely ignores the quality of the product [34];
- Many studies treat an organization as a single individual consumer, disregarding the many individual opinions of employees, leading to a single organization through a political arena [20, 38, 39];
- The relative importance of factors is ignored, depending on the specific context factors might become more important than others [40];
- Factors might not be universally applicable, but might be industry dependent [40].

Based on this critique, an extended SERVQUAL model has been developed, explicitly including the quality of the product and the relative importance of factors [15]. The case study showed that the SERVQUAL factors all had significance in the context of mission critical software development (i.e. soundness of the model), but that one aspect was missing: **Partnership**, where the developer strives for the same goals as the customer [15]. This factor has been recognized by other studies as well for being critical in long-lasting customer-developer relationships [41]. For the remainder of this paper, we adopt this extended SERVQUAL instrument, including the proposed factor **Partnership**.

Looking at the case study in [15], it has been applied in a context where the product quality was meeting expectations, never requiring major repair actions or other technical interventions to rescue an important business
process. There aren't any focused studies that look at the role and decomposition of Delivery Quality when the product fails, aside from predicting that the customer will be unsatisfied. So when it comes to hypothesis 2, the results are quite unpredictable.

3. Case Study

The case presented is one where the product clearly had significant shortcomings. The goal of the case study is to investigate what the effects of a failing product, along with its significant impact on the primary operation of the company, had on the role and decomposition of Delivery Quality. Although the lack of product quality clearly can not be compensated by improving on the Delivery Quality aspects, the impact in such a context is unknown. Also the greater emphasis on recovery operations in such a case might have a significant impact on the importance and structure of the model. First we describe the method used for investigating the case, followed by the context of the case. We conclude this chapter by presenting the case results.

3.1 Research design

Customers of mission critical software are very diverse and the relations are quite complex, making it less suitable for quantitative studies [20, 42]. We therefore use qualitative methods to gain insight in the factors that contributed to customer satisfaction in this specific case. We do this by using structured questionnaires and in-depth follow-up interviews, an approach more fitting with the more complex developer-customer relationship [20]. This questionnaire-approach has the benefit of allowing respondents to reflect without the pressure to rate specific aspects or name specific incidents on the spot. We first elaborate on the specific design of the questionnaire and the structure of the interview, before we discuss how this relates to our hypothesis.

The questionnaire investigates customer satisfaction and aims to inventory the reasons for the (dis)satisfaction level. It has two distinct sections: a closed section and an open section. The closed section aims to score the satisfaction among various factors. The factors considered are a combination of the extended SERVQUAL model [15] and the ISO 25010 [31]. This combination is necessary since the product is an essential part of customer satisfaction [33, 34] and including it prevents attribution of product problems to Delivery Quality (most likely Reliability) [35].

The open section of the questionnaire has three major parts:

- A free format section ranking the relative importance of factors with respect to the respondents (dis)satisfaction level;
- A free format section identifying the most positive and most negative critical incidents, the CIT technique. It is known for being an effective technique in identifying the most dominant causes of (dis)satisfaction in business-to-business relationships [20, 36];
- The actual scoring of the resulting level of customer (dis)satisfaction.

The questionnaire is followed by a two-hour interview that aims to clarify the underlying reasoning for specific scores, as well as discuss the clarity and completeness of the questionnaire. This allows us to make the causal relation between model factors and critical incidents explicit. This is done by discussing the rating of the factors with the respondent, inviting the respondent to describe his underlying reasoning in the scoring of specific factors. The second goal is asking meta-questions: questions about the format and completeness of the questionnaire itself, allowing the identification of factors that the respondent felt were missing or unnecessary. The interview therefore not only deepens the understanding of the scoring but also acts as a safety net for missing or badly defined factors.

The relevance of Delivery Quality in this context, our first hypothesis, can be investigated by looking at the weight of the factors that determined the respondents' satisfaction level and the critical incidents, both part of the open section of the questionnaire.

The soundness and completeness of the decomposition of Delivery Quality, hypothesis 2a, can be investigated by comparing the explanation of the factor scoring with the critical incidents. The model is complete when respondents explicitly relate all incidents in the critical incident section of the questionnaire to some factor. This is double-tested by explicitly asking for completeness in the meta-questions in the interview. The model is sound if none of the respondents indicate during the interview that they felt the questions were irrelevant.

The stability of the factor definitions, hypothesis 2b, can only be tested by comparing the respondents' interpretations of factors (clarified by their deliberation on their scoring during the interview) with the definitions used in previous contexts of the extended SERVQUAL model. It should be noted that the extended SERVQUAL model used is hardly tested, implying that instability might originate from using unclear definitions instead of the change of context.

The unjustly projecting of a failing product upon Delivery Quality, hypothesis 2c, can be identified during the interviews where respondents are required to explain their scoring. Especially the factors Reliability and Assurance are quite prone to this, but in practice all explanations of the scoring of factors that belong to the extended SERVQUAL model have to be checked for an explanation that is related to the lack of product quality.

So in conclusion, the main method of investigation is the questionnaire that covers all product and process factors in a project, the critical incidents that happened and the resulting satisfaction level. The interview focusses on understanding the scores of factors, identifying underlying critical incidents when possible, and allows...
discussing the meta-level questions of completeness and clarity.

3.2 Context

We test our case in a context where the product quality is clearly lacking and beyond discussion. The project clearly failed to deliver the stable product required by the business, and the project was terminated by reverting to its predecessor.

The customer is a large global logistics company, where the company had people involved from both the IT function as well as the management of the primary process. The developer was a smaller company that explicitly specialized in very specific logistic solutions.

For the customer it was an extremely important project: it was to automate a significant portion of a logistics process that couldn't be handled manually and had severe penalties on late delivery. It replaced a legacy system that had to be replaced due to technical reasons as well as new legal requirements that couldn't be fulfilled.

The new systems was a Commercial Of The Shelf (COTS) product that was successfully demonstrated on several other (much smaller) sites, which led to the conclusion by the business-side of the customer that this product was the right product to replace the legacy system.

After several years of implementation the project was halted because several severe disturbances of the primary business process resulted in both a loss of production volume and in many penalties. A technical investigation showed that the software architecture was spaghetti, and wasn't capable of reaching the volume and availability targets the developer and customer agreed, which resulted in an early termination of the project.

3.3 Case results

The case study involved 9 people:

- 6 from the customers' project organization, embedding the product in the existing infrastructure and organization;
- 2 were the project owners on the business side
- 1 contract manager

Out of these 9 people 6 returned the questionnaire, allowing us to investigate the case. The three non-respondents (typically higher management) were also interviewed to gain more insight but aren't included in the analysis unless explicitly stated otherwise, since the information gained proved too shallow.

The general consensus was, unsurprisingly, that all 9 interviewed were extremely dissatisfied. Also none of the 9 interviewed were willing to start another project with the developer in the current context. It should be noted that two respondents indicated during the interview that they would be willing to recommend or use the business consultants and the product for a much simpler context: in their opinion the complexity of their own technical requirements and organization was one of the major underlying reasons for the failing of the project.

First we look at the areas that contributed most to the current level of (dis)satisfaction of the respondents. The following table summarizes the areas of improvement as mentioned by the respondents with their attributed rankings, where the value in the cells indicates the number of respondents that used a specific ranking for the specific area as a reason for (dis)satisfaction:

<table>
<thead>
<tr>
<th>Areas most contributing to respondents' satisfaction</th>
<th>Importance for (dis)satisfaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instability of the product</td>
<td>3rd, 2nd, 1st</td>
</tr>
<tr>
<td>Domain knowledge</td>
<td>1</td>
</tr>
<tr>
<td>Data Integrity</td>
<td>1</td>
</tr>
<tr>
<td>Functionality</td>
<td>1</td>
</tr>
<tr>
<td>Application responsiveness</td>
<td>1</td>
</tr>
<tr>
<td>Usability of user interface</td>
<td>1</td>
</tr>
<tr>
<td>Not honoring commitments</td>
<td>1</td>
</tr>
<tr>
<td>Security of the application</td>
<td>1</td>
</tr>
<tr>
<td>Changeability of application</td>
<td>2</td>
</tr>
<tr>
<td>Defect handling</td>
<td>1</td>
</tr>
<tr>
<td>Integration other systems</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Areas important for current (dis)satisfaction level

Table 1 shows that the dominant reason for dissatisfaction is the instability of the system: 3 out of the 6 respondents ranked it as their most important reason for (dis)satisfaction and 2 ranked it as their second important reason for (dis)satisfaction. Clearly the instability of the product was mentioned by everyone, and it was always in the top 3. It also shows the in-depth domain knowledge as a distant second reason for satisfaction.

Looking at the areas of improvement, the following table summarizes the ranking of areas of improvement to get a higher satisfaction level, where each cell represents the number of respondents that ranked a specific area for improvement:

<table>
<thead>
<tr>
<th>Areas of improvement</th>
<th>Importance for satisfaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stability of the system</td>
<td>1st, 2nd, 3rd, 4th</td>
</tr>
<tr>
<td>Software Architecture</td>
<td>2, 1</td>
</tr>
<tr>
<td>Actively invest in partnership with customer</td>
<td>1</td>
</tr>
<tr>
<td>Honoring commitments</td>
<td>2</td>
</tr>
<tr>
<td>Accuracy of data</td>
<td>1</td>
</tr>
<tr>
<td>Better documentation</td>
<td>1, 1</td>
</tr>
<tr>
<td>Better customer management organization</td>
<td>1</td>
</tr>
<tr>
<td>Application responsiveness</td>
<td>1</td>
</tr>
<tr>
<td>Defect handling</td>
<td>1</td>
</tr>
<tr>
<td>Usability of user interface</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 2: Areas of improvement

Table 2 shows that 3 out of 6 respondents indicated that the most important improvement needed to improve their satisfaction was the improvement of the stability of the system. This table shows a similar pattern to table 1: the improvement of the product stability was deemed very important by all respondents. The lack of software architecture was the root cause of the instability of the product which could be seen as synonym for product stability. During the interviews it became clear that there was a very strong emotional distinction between these
areas: the customers felt betrayed by the developer by selling an unsuitable product with severe hidden defects that frustrated their primary process.

A remark has to be made about "not honoring commitments". In the questionnaire it was mentioned twice as a dissatisfier and twice as the second most important area of improvement by others. However, it is also mentioned during all 9 interviews as a serious problem in the relationship since it concerned major contractual areas as operational procedures, development/deployment strategy and delivering support manuals.

Looking at the critical incidents, the most frequently identified is the inability to deliver a stable product, leading to problems in the primary process:

<table>
<thead>
<tr>
<th>Respondent</th>
<th>Attribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Mentioned in the CIT, attributed to Product stability</td>
</tr>
<tr>
<td>B</td>
<td>Mentioned in the CIT, attributed to Product stability</td>
</tr>
<tr>
<td>C</td>
<td>Mentioned in the CIT, attributed to Product stability</td>
</tr>
<tr>
<td>D</td>
<td>Mentioned in the CIT, attributed to Product stability</td>
</tr>
<tr>
<td>E</td>
<td>Mentioned in the CIT, attributed to Product stability</td>
</tr>
<tr>
<td>G</td>
<td>Mentioned in interview, attributed to Product stability</td>
</tr>
</tbody>
</table>

Table 3: Attribution of product instability

Some respondents also indicated that this was the cause to terminate the project: although the results of the technical audit were the direct reason to terminate the project, the problems the instability caused were the underlying cause to terminate the project. The acquisition team couldn't "sell" the product to the business anymore and the audit only showed that it wouldn't improve in the short term.

A positive critical incident is the strong commitment by the support personnel:

<table>
<thead>
<tr>
<th>Respondent</th>
<th>Attribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Mentioned in the CIT, attributed to Partnership</td>
</tr>
<tr>
<td>B</td>
<td>Mentioned in the CIT, attributed to Partnership</td>
</tr>
<tr>
<td>C</td>
<td>Mentioned in the CIT, attributed to Partnership</td>
</tr>
<tr>
<td>D</td>
<td>Mentioned in the CIT, attributed to Partnership</td>
</tr>
<tr>
<td>E</td>
<td>Not mentioned</td>
</tr>
<tr>
<td>G</td>
<td>Mentioned in the CIT, attributed to Partnership</td>
</tr>
</tbody>
</table>

Table 4: Attribution of commitment support personnel

For 5 of the 6 respondents the commitment of the support employees was without question: the lead developers were on call 24x7 and quickly travelled on-site when production issues arose and only left for home when problems were fixed. Some respondents also explicitly stated that this was a requirement for Partnership: a developer has to take responsibility for his product and how it behaves on a customer site without debating time or costs. During the interview, some indicated that without this extreme commitment from key developers the project would have been terminated much earlier.

Another positive critical incident is the in-depth knowledge and understanding of the business process:

<table>
<thead>
<tr>
<th>Respondent</th>
<th>Attribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Mentioned in the CIT, attributed to Empathy</td>
</tr>
<tr>
<td>B</td>
<td>Mentioned in the CIT, attributed to Empathy</td>
</tr>
<tr>
<td>C</td>
<td>Mentioned in the CIT, attributed to Empathy</td>
</tr>
<tr>
<td>D</td>
<td>Mentioned in the CIT, attributed to Empathy</td>
</tr>
<tr>
<td>E</td>
<td>Mentioned in interview, attributed to Empathy</td>
</tr>
<tr>
<td>G</td>
<td>Not mentioned</td>
</tr>
</tbody>
</table>

Table 5: Attribution of excellent domain knowledge

7 out of the 9 interviewed recognized the domain knowledge, some indicating that the developer sometimes knew the business better then they knew it themselves. This was part of the appeal during the sales phase of the project. However, challenging the customer and using this knowledge to improve the customers' business process did not occur, making the project technically more challenging and less beneficial for the business. This in-depth knowledge was also one of the reasons two respondents indicated they would like to use the business consultants of the developer again.

Another dominant critical incident is the inadequate repairs by the developer, highlighted by 2 respondents in the questionnaire but mentioned in all 9 interviews:

<table>
<thead>
<tr>
<th>Respondent</th>
<th>Attribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Mentioned in the CIT, attributed to Responsiveness</td>
</tr>
<tr>
<td>B</td>
<td>Mentioned in the CIT, attributed to Responsiveness</td>
</tr>
<tr>
<td>C</td>
<td>Mentioned in interview, attributed to Responsiveness</td>
</tr>
<tr>
<td>D</td>
<td>Mentioned in interview, attributed to Responsiveness</td>
</tr>
<tr>
<td>E</td>
<td>Mentioned in interview, attributed to Responsiveness</td>
</tr>
<tr>
<td>G</td>
<td>Mentioned in interview, attributed to Responsiveness</td>
</tr>
</tbody>
</table>

Table 6: Attribution of inadequate repairs

For all 6 respondents of the questionnaire, the inadequate repairs clearly were related to Responsiveness: the response on critical software bugs was quick but ineffective. Fixes were deemed "simple" too quickly and either didn't work or broke something else in the production environment. Some problems needed 20 quick fixes to be repaired, with severe consequences for the primary process. Clearly the effectiveness in the response mattered to the respondents.

The last critical incident reported by multiple respondents is the transparency about architecture problems

<table>
<thead>
<tr>
<th>Respondent</th>
<th>Attribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Mentioned in the CIT, not attributed</td>
</tr>
<tr>
<td>B</td>
<td>Mentioned in interview, not attributed</td>
</tr>
<tr>
<td>C</td>
<td>Mentioned in interview, not attributed</td>
</tr>
<tr>
<td>D</td>
<td>Mentioned in the CIT, not attributed</td>
</tr>
<tr>
<td>E</td>
<td>Suggested at interview, not attributed</td>
</tr>
<tr>
<td>G</td>
<td>Not mentioned</td>
</tr>
</tbody>
</table>

Table 7: Attribution of transparency about architecture problems

According the respondents, the developers bought themselves a lot of time under the guise of "Performance tuning", masking fundamental architecture issues the developer must have been aware of. Several respondents indicated that revealing these issues was an essential turning point in their trust in the developer. Two respondents explicitly stated that if the developer had been transparent about the architecture problems, they would have been given time to refactor their product. They also indicated that, given the high stakes for the customer, they probably would have contributed financially to this improvement. For the respondents it didn't matter if the lack of proactive escalation of the architecture problems originated from incompetence or bad intentions: hiding these problems led to a feeling of betrayal and made all subsequent statements by the developer untrustworthy for these respondents.
4. Discussion of results

In this section we discuss the results; first we start by investigating the hypotheses, followed by a general observation about the research instrument.

In this case, it is quite clear and unsurprising that Product Stability is the most important factor in the customer satisfaction. Without a doubt, the instability of the product, and the problems it caused in the primary process, were the cause for the project termination. The direct causal link found in this case empirically supports the statement from literature that not delivering the right product is a major dissatisfier at the post-project evaluation [3, 13, 16, 24, 25, 26]. Our research does indicate that also extends to the acquisition/project management team during a project as well: there is a matter of professional pride that can not allow the primary business to get hurt by a supplier they are managing.

The domain knowledge of the developer is ranked second as a major factor for satisfaction, which is also consistently recognized in the critical incidents by 4 of the 6 respondents. Although it did not have a direct effect on the course of the project, some respondents would like to involve the developers' business consultants again for future projects, so it has some effects on rebuy intentions.

The commitment of the developers had an important role in the course of the project. Most respondents recognized the strong commitment from the developer and some explicitly indicated that without this strong commitment the project would have been terminated much earlier. Given the impact the failures had on the primary process of the customer this is hardly surprising. The lack of effectiveness of these repair-actions does seem to be separated in the minds of the respondents. Looking at literature, this observation confirms the importance for defect handling [9, 21] for satisfaction of the acquisition team, where apparently the commitment to fix product issues is an important factor.

A major factor that has altered the course of the project is the lack of transparency about fundamental architecture problems in the core product. During the interviews there was a highly emotional distinction by the respondents between the instability of the product and the lack of architecture that caused it. Looking at the critical incidents and the explanations by the respondents we understand why: the acquisition/project management team felt betrayed and lost trust in the developer because of his hiding critical flaws, which generally is terminal for projects [30].

Looking at the first hypothesis, when it comes to a post project evaluation of customer satisfaction it is clear that Delivery Quality can not compensate for a failing product, confirming statements in literature [3, 13, 16, 24, 25, 26]. However, Delivery Quality can alter the course of a project. The presented case highlights this in both a negative and positive way. If the lead developer had been less committed, the project would have been terminated much quicker. If the developer had been transparent about the architecture problems he would have been actively supported by the acquisition/project management team to refactor his product, potentially rescuing the project. So we conclude that a lack of Delivery Quality altered the course of this project. If the developer had been more transparent, the acquisition/project management team would have arranged for more time and it would even have allowed the developer to rescue this troublesome project, potentially even turning it into a successful one.

When looking at soundness and completeness of the decomposition of Delivery Quality, hypothesis 2a, none of the respondents mentioned they missed a factor or found one superfluous when asked during the interview. Also most critical incidents were quite consistently attributed by the respondents to the current factors. However, the outcome of the CITs and the importance-ranking of the factors do indicate that a factor is missing. During the interviews it was indicated that the lack of transparency about the architecture problems altered the course of the project, but isn't attributed to a specific factor by any of the respondents. It is mentioned in the CIT and the factor importance ranking. Looking at the extended SERVQUAL model, there aren't any likely factors that would cover this aspect of the relation. Reflecting on the case described in [15], the need for transparency might be overlooked since that project used a Prince® approach which forces the discussion of (internal) project risks with the customer. In literature other authors also recognize the customers' need for transparency about arising problems and critical events [18]. We think there is an omission in the extended SERVQUAL model and that the factor Transparency should be added to make it suitable for mission critical environments.

Most of the factor definitions are stable, the respondents' attribution of critical incidents to the factors fitted well with most of their original definitions, supporting hypothesis 2b. However, there are two exceptions:

- **Responsiveness**: clearly contained an element of effectiveness. Although hardly surprising given the context, it is not part of the original definition in the extended SERVQUAL model. Marketing literature does suggest an interpretation that does include effectiveness [24], so this is clearly an oversight by the original definition [15];
- **Partnership**: For the respondents of this case, Partnership included a strong sense of a developer taking responsibility for his contribution to the primary process. So aside long-term goal coherence, he is also expected to act accordingly on the short term. Marketing literature also highlights the importance of commitment to solving customers' problems to reassure their importance [9, 13]. So this is also considered an oversight by the original definition.

Both oversights can be explained by the scope of the initial case where the extended SERVQUAL model was tested [15]: due to the stringent quality processes used in
that case there wasn't much emphasis on repairing software, making both these aspects of the definition less visible.

The analysis of the critical incidents reveals that none of the critical incidents tried to attribute product failure to a Delivery Quality factor, supporting hypothesis 2c. The lack of architecture does come close, Assurance would have been a likely candidate. However the respondents explicitly stated that it was the "hiding" of the problem, rather than the architecture problem itself, that was the problem. This is supported by the explicit distinction between the product quality issues and the architecture problems that was made during the interviews. We, as well as the respondents, see the hiding of these problems as a Delivery Quality issue, instead of a product quality issue.

Looking at the results, a note has to be made about the effectiveness of the research method itself and the outcome of the various sections. The setup using a questionnaire followed by an interview does seem to be necessary. The questionnaire provides a solid structure which is hard to extract during an interview. Especially the quality of the importance-ranking of factors and critical incidents deteriorated when asked on the spot during the interview. This might be due to time pressure or other psychological effects, but the information became too anecdotal to be taken into account in the analysis.

Another note has to be made about the effectiveness of the identification of critical incidents. Some authors indicate that critical incidents define the relation [20]. However, we see that specific aspects of the relation (like the not honoring agreements in this case) aren't identified by the CIT section in our questionnaire, despite being ranked as an important factor for dissatisfaction by 4 respondents, and mentioned during all interviews as being important for the relationship. This could be explained by these incidents being overwhelmed by other more important incidents, or that the behavior was so structural that it wasn't considered an incident anymore, but it is important to realize that the critical incidents technique doesn't catch all relevant aspects of a developer-customer relationship.

5. Conclusions and implications

In order to stay competitive, software developers should focus on successful project completion and turning customers into repeat customers. This case illustrates that in the end nothing can compensate for a failing product repeatedly disrupting a business critical process, leading to project failure. The unsatisfactory outcome of a project clearly is dominant when it comes to customer satisfaction for both end users and acquisition/project management team.

The dominant nature of the failing product in the post project satisfaction analysis does not exclude the importance of Delivery Quality during project execution: the case presented illustrates that a high commitment by the developer allowed the project to continue and more transparent communication about fundamental product issues would have resulted in more time from the acquisition/project management team to get product quality under control. Although this is in a very specific organizational context, it does indicate that Delivery Quality has enough power to influence the course of a project to such an extent that it changes the project outcome. Our conclusion therefore is that Delivery Quality has an important role when the project outcome isn't directly satisfactory for the customer.

The case study does highlight several issues with the extended SERVQUAL model as presented in [15]. These issues are most likely caused by its limited testing. Testing this model in the context of a failing product puts a lot of emphasis on new factors, like Transparency. It also highlights new aspects of existing factor definitions, like a strong commitment to get a production process going again and the effectiveness of repairs. Our conclusion is that the extended SERVQUAL model would be improved by extending it further with factor Transparency and improving the definitions of Responsiveness and Partnership. It is unclear if this is sufficient to reach completeness and stability. In a research setting it extracted more useful information than a plain CIT, but there should be a concern about the models completeness and definition stability.

Future research should focus on testing the completeness and stability of the extended SERVQUAL model in different contexts. This case shows that by changing context from a perfect product to a failing product several shortcomings in the extended SERVQUAL model are exposed. This implies that future research should focus on testing this model in other contexts like for example a troubled (but not failed) project.

For practitioners this case shows the need for specific factors of Delivery Quality when the product is failing, allowing them to "buy" enough time to fix project outcome problems. Especially factors like Responsiveness and Partnership (in this case commitment to the customer) are essential. This is a generally recognized concept when looking at service quality: service recovery requires a quick and effective action that clearly shows the commitment of the service provider to satisfy his customers [9, 13].

For practitioners this case also contains a tale of caution. A lack of transparency about fundamental product problems can easily backfire resulting in a customer feeling betrayed by his developer, leading to a radical different relationship and potentially project termination. Although certainly not applicable for all contexts, some customers might rather live in the bliss of ignorance, developers should at least consider being transparent about the need for addressing serious issues in their product proactively.
References


