Communication Factors for Speed and Reuse in Large-Scale Agile Software Development

Antonio Martini
Chalmers | University of Gothenburg
SE 412 96 Gothenburg, Sweden
+46-31-772 5710
antonio.martini@chalmers.se

Lars Pareto
Chalmers | University of Gothenburg
SE 412 96 Gothenburg, Sweden
+46-31-772 6040
pareto@chalmers.se

Jan Bosch
Chalmers | University of Gothenburg
SE 412 96 Gothenburg, Sweden
+46-31-772 5716
jan.bosch@chalmers.se

ABSTRACT
An open issue in industry is the combination of software reuse in the context of large scale Agile Software Development. The speed offered by Agile Software Development is needed for short time to market, while reuse strategies such as Software Product Line Engineering are needed for long-term productivity, efficiency, and profit. The paper investigates, through a survey, communication factors affecting both speed and reuse in 3 large companies developing embedded systems and employing Agile Software Development and Software Product Line Engineering. Our results include a prioritized list of communication related factors obtained by statistical analysis and the recognition and spread of the factors in the companies. We have recognized 5 interfaces with the Agile development team that need to be improved: system engineers (architects), product management, distributed teams, inter-project teams and sales unit. Few factors (including inter-project communication) depend on the business drivers for the company. We also reveal that Agile teams need strategic and architectural inputs in order to be implanted in a large company employing Software Product Line Engineering. Academic and industrial training as well as different tactics for co-location would improve the communication skills of engineers. There is also a need for solutions, in the reference architecture, for fostering Agile Software Development: the goal is the combination of the focus on customer value of the teams, reusability, system requirements and avoidance of organizational dependencies.

Categories and Subject Descriptors
D.2.9 [Management], D.2.13 [Reusable Software], H.5.3 [Group and Organization Interfaces]

General Terms
Management, Economics, Human Factors, Theory.

Keywords
Agile software development, software reuse, development speed, embedded systems, speed, communication, factors, software process improvement (SPI).

1. INTRODUCTION
Recently Agile Software Development (ASD) has gained significant attention in large software companies striving to reduce time-to-market. By parallelizing the development of features or components, using teams, such companies increase speed, and their ability of continuous delivery of customer value [4].

Another well-known approach to reduce time-to-market is Software Product Line Engineering (SPLE), which aims at reusing software components or platforms across a line of products [5]. This fosters the development of high-quality mass-customized products at significantly reduced costs and increased speed in product replication and maintenance.

An open issue in research and industry is the compatibility of ASD (reactivity to customer needs and the speed to hit the market) and the SPL (speed in rolling out product innovation across a product line) strategies. Several theoretical studies argue that the two approaches are compatible in principle [3].

However several imperative studies in companies trying to combine ASD practices with SPL or SPL-like practices have shown engineering communication to stand out as major challenge. Past research by the authors found communication related challenges to outnumber the challenges in any other areas of software engineering [12]. Pawar and Sharifi found that teams in large organizations require coordination and communication with many stakeholders, both horizontally (other teams) and vertically (organizational hierarchy) [14]. Some studies highlighted how communication factors often cause inefficiencies [1][4][15][12], and hinder the speed gained by the parallelization of the development [15]. Communication factors have also been found in distributed ASD, e.g. Global Software Development [6].

Understanding what communication factors in software development are the most frequent and critical, would ease the introduction and management of ASD in large-scale software companies employing or seeking to employ SPL. Moreover, as the cost to set up communication channels might be onerous, in terms of time and resources, guidelines for selecting such the critical ones would reduce the efforts in addressing the communication aspect for large software organizations introducing ASD, as resources could be directed to the most critical interfaces.

The paper presents a ranking and investigation of 23 factors hindering speed and reuse in three large product developing organizations seeking to implement ASD with due attention to SPL principles. The research design includes a survey collecting quantitative and qualitative data from 36 respondents. The data
were analyzed using statistical methods in combination with grounded theory and thematic analysis [17].

The precise research questions are:

**RQ1** Which are the most recognized and wide-spread communication factors hindering speed and reuse when employing large scale ASD?

**RQ2** Which communication factors (in RQ1) are context dependent? What are the relevant context parameters?

**RQ3** What communication factors (in RQ1) are recognized only by some roles in the companies? Why?

**RQ4** How are these factors related to large scale ASD and what can be done to manage them?

The contributions of the paper are

- a prioritized list of communication factors hindering speed and reuse related to ASD, based on their recognition (strong, controversial, weak) and their spread (bar display) through the companies (confirmatory results);
- a set of factors dependent on the companies’ contexts, and their connection to context parameters (exploratory hypothesis);
- a set of critical communication interfaces between different roles in large software companies (exploratory hypothesis);
- a set of improvement suggestions based on qualitative analysis (exploratory hypothesis);

in the context of large scale ASD combined with SPL in embedded systems development.

The paper continues with the explanation of our theoretical framework (section 2) and the research design (section 3). Section 4 outlines our results, discussed in section 5. Section 6 relates our findings with other scientific works and we draw our conclusions in section 7.

### 2. BACKGROUND

#### 2.1 Theoretical Framework

![Figure 1. We study the communication factors influencing reuse and speed, important for Return of Investments](image)

Our theoretical framework is outlined in Fig. 1. Several factors in an organization that employs SPL might hinder two business goals: replication speed and 1st deployment speed, which in turn affect the Return of Investment of R&D [11]. Replication speed is usually reached by reusing software components, which is a key strategy in SPL, and it’s important to keep the products cheap and to minimize maintenance work. 1st deployment speed is important for short time to market. Some of the factors are context dependent (such as market, geographical, legal constraints etc.) and therefore are not manageable. Other factors are manageable, which means that organizations can invest resources to modify the factors. Communication factors are manageable factors, and, according to a previous study [12], are important to match ASD with reuse practices (SPLE), and therefore to combine 1st deployment speed and replication speed. We have selected communication factors (the grey set) that inhibit reuse or speed (“-“ arrows in the picture).

The contexts are the companies that we have studied, all of which develop embedded software through a line of products. They are described in section 3.1. Some of the communication factors might depend on other non-manageable factors that are embedded in the companies, and we call them context-related factors.

#### 2.2 Roles description

The roles whose communication is most important to improve while combing ASD and SPL practices are:

- **Managers**: responsible for the performance of a group of development people, either at a project or a product level. The successful achievement of software reuse (SPLE) is very dependent on strategic choices made at a managerial level.
- **System engineers/Architects**: responsible for breaking down the system requirements to sub-systems and for the architecture of the (sub-) systems. (referred in the paper as system engineers) Such role is fundamental in SPL for domain engineering, i.e. for the preparation of the components that need to be reused.
- **Designers**: responsible for designing solutions with the implementation of components and algorithms as well as some architectural views. Such developers might be responsible for reused or product-specific components.
- **Testers**: responsible for validating and verifying that a program works as expected and meets the requirements that guided its design and development.

After consultation with the studied companies we needed to choose these roles, as they were common to all the companies.

#### 2.3 Agile Software Development

**Agile Software Development** is an umbrella for several practices and principles. In our context, ASD is employed by the establishment of Agile development teams implementing sets of practices such as Scrum and dedicated to the development of features or components. The principles of ASD have been strategically chosen by the studied companies, with the goal of developing software continuously and iteratively. However, parts of the organizations still follow a waterfall model, as embedded systems need to follow an overall product development bound to the final physical product.

Communication in ASD is central for the achievement of reactivity to changes in requirements. More in general, the ASD manifesto stresses the value of the iterative and continuous feedback among different team members, for continuous improvement and sharing of knowledge.

#### 2.4 Software Product Line Engineering

**Software Product Line Engineering** is an approach for the planned and systematic reuse of software components or platforms for a line of products. In our contexts, all the companies develop at least a line of products, in which SPL is applied but without the approach “of the book”, i.e. without using all the roles and specific process found in the SPL literature. However, it was pretty clear during the study that the companies employed (different degrees of) SPL. All of the companies used a component-based architecture, and shared components among the different projects and products. In two cases out of three the
products shared a platform on top of which different features were developed and sold for specific customers. In the third case the reuse was less organized. All the companies used a reference architecture maintained by system engineers/architects.

Communication in SPLE assures the flow of information between different parts of the organization, necessary to guarantee compatibility between reused and product-specific components (domain-application coordination) and to make the evolution of reused assets effective throughout the product line [9].

3. RESEARCH DESIGN

We planned a multiple-case case study by conducting a questionnaire in three large software development companies, all with extensive in-house embedded software development. The companies studied were to have several years of experience of ASD. The companies chosen were situated in the same geographical area (Sweden), but were active on different international markets. For confidentiality reasons, we will call the companies A, B and C. This study followed another interview study [12] from which the communication factors were previously extracted and operationalized in questions for the survey.

3.1 Case Descriptions

Company A is involved in the automotive industry. Part of the development is carried out by suppliers, some by in-house teams following Scrum. The surrounding organization follows a stage-gate release model for product development. Business is driven by products for mass customization, and SPLE is employed in the form of the maintenance of an explicit reference architecture, reused components and the management of variability with in-house mechanisms.

Company B is a manufacturer of product lines of utility vehicles. In this environment, the teams are partially implementing ASD (Scrum). Some competences are separated, e.g. System Engineers sit separately. Special customers requesting special features drive the business, and 1st deployment speed is the most important business goal. SPLE is less present than in case A: projects are less cooperating and reuse is not always effective and successful, for a lack of strategic decisions in the management towards replication speed.

Company C is a manufacturer of telecommunication systems product lines. Their customers receive a platform and pay to unlock new features. The organization is split in cross-functional teams, most of which with feature development roles. Some of the teams have special supporting roles (technology, knowledge, architecture, etc.). Most of the teams use their preferred variant of ASD (often Scrum). Features were developed on top of a reference architecture, and the main process consisted of a pre-study followed by few (ca. 3) sprint iterations.

3.2 Questionnaire

We created an on-line questionnaire that was designed using a mix of closed- and open-ended questions [2]. The first version of the survey was tested by two PhD students to have feedback from an academic point of view and by at least one contact in each company to make the questionnaire fitting the jargon and being understandable for all the respondents. During the test we also monitored time and caught misunderstandings due to question formulation.

The motivation for using an on-line questionnaire was to maximize coverage and participation without having to set up a time-consuming interview system. A survey is a suitable method for data collection when asking structured questions [2]. The participants could access the survey whenever they wished. The answers were bound to a web-link and a mail address, so the respondents could also interrupt the survey and continue it in another moment. The questionnaire was made accessible on-line at surveymonkey.com between April and June of 2012.

The questionnaire was structured in 3 parts:

- 2 questions about the role of the respondent and the organizational unit of work. We merged in the end the roles in “manager”, “system engineer”, “designer” and “tester”.
- a set of 23 communication factors. In the survey we called them challenges. For each one, the respondents were asked to give an exclusive answer between 4 options: “No, I don’t recognize the challenge”, “Yes, I recognize the challenge in other units”, “Yes, I recognize the challenge in my own unit”, “Yes I recognize the challenge in my unit and other units”. When appropriate, we added a fifth answer, in which the respondents could express their impossibility to answer due to a lack of experience in the described situation. The 23 factors were divided in 5 categories (explained below). For each question, we asked also for suggestions for the improvement of the factor.
- A final question in which the respondents ranked the areas of improvement (Architecture, People, Processes, Organization, Technology) by importance when addressing the communication challenges related to the factors.

The 23 factors are the results from the previous exploratory interview study conducted in the same companies [12], in which we categorized them as communication-related. We operationalized the factors in closed-questions for the survey. In some cases we decided to split the factor in two questions. For example, the inhibitor “different tool and programming languages” was split in two questions, Q17 and Q18. For most of them, however, there is a one-to-one correspondence between the factors and the questions in the survey.

We defined 5 categories using the analysis of a different study [12]. First, we defined hard and soft areas. Hard areas are the ones more related to artifacts rather than to personal interaction, and included 2 sub-areas (Architecture and Technology):

- **Architecture**: the challenges in this category are related to misunderstanding or an unnecessary flow of communication due to the definition of a system and software structure.
- **Technology**: the different tools and programming languages used by the team for development may cause conflicts on the preferred technology or delays due to incompatibility of artifacts.

Soft areas, involving prevalently social interaction, included Processes, Organization and People:

- **Processes**: the definition of who is responsible for delivering what, to whom and when may be underspecified causing unnecessary communication, or may not match the other processes, causing lack of communication opportunities.
- **Organization**: the definition of the structure of task allocation, coordination and supervision may create communication barriers.
- **People**: communication problems may arise due to personal (or group) attitudes, mindset or knowledge.

Such categories were formulated in order to ask the respondents for suggestions.

We wrote a 22 pages survey guide with detailed explanation for most of the factors, complemented with concrete examples. The guide was accessible through the web and linked in the survey, so
it was always available for the respondents while they were answering the survey. The challenges are outlined in Table 1:

| Q01 | The processes/ways of working that you have to follow are not suited for the kind of product that you are developing |
| Q02 | The processes/ways of working that you have to follow are not suited for the project management |
| Q03 | Project-related bugs or defects |
| Q04 | Developers and system engineers are not co-located, which causes communication problems in requirement agreement |
| Q05 | There is an upcoming product. Erroneous assumptions have been made on what part of the existing software can be reused and/or adjusted, causing inaccurate budget or resources allocation |
| Q06 | Evaluation of costs and feasibility of new features don’t take into account implementations and constraints, causing inaccurate budget and resources (for example time or workload) allocation |
| Q07 | There is no time to improve parts of shared technology among projects. |
| Q08 | A satellite unit is “invisible”: for example, it’s difficult to consult them, there is no clear information on their work or cannot be guided properly (the kind of problem may depend on your role) |
| Q09 | A development unit has to build a component to be integrated in other units’ (projects’) system, but the communication with them is not sufficient. |
| Q10 | A development unit was forced to integrate a common component. This caused communication problems, and now the unit is not willing to integrate new common components. |
| Q11 | Reuse is not supported by the Product Line Management, it’s an individual initiative. |
| Q12 | Different attitudes and values of distributed (not co-located) teams (or units, projects) caused communication problems. |
| Q13 | Team’s (or unit, project) lack of will to integrate a common component. |
| Q14 | Lack of understandable documentation or of proper communication causing long warming up periods for consultancy or for new employees to understand the system |
| Q15 | Documentation is abundant but it doesn’t help to understand the code |
| Q16 | Leaders’ mindset is not open to listen and they are not able to recognize strengths and weaknesses. This hinders the development of improvements. |
| Q17 | Different favorite programming languages in the same team create communication. |
| Q18 | Different favorite tools in the same team create communication problems. |
| Q19 | Artifacts received from the system engineers are not clear enough because they were created with inappropriate tools |
| Q20 | Developers are too constrained by system engineers on design (for example, developers receive white box/very detailed specification) |
| Q21 | An internal interface had to be exposed to other units to allow their development. The interface documentation provided didn’t help to understand the code. |
| Q22 | Disagreement between different development units (or projects) about what set of functionalities a common component should provide. This creates communication problems |

### 3.3 Data Analysis

We performed statistical analysis on answers to the closed-ended questions complemented with qualitative analysis of answers to the open-ended ones.

#### 3.4 Factor Recognition

The recognition of a factor is based on how many respondents have personally experienced the factor. This is used for confirmation of previous results and for prioritization purposes. We have grouped the factors in strongly recognized and weakly recognized, depending on the ratio of the participants answering “yes” or “no”. However, when the number of yes and no is similar, we consider the factors as controversial, which means that even if the factors are strongly or weakly recognized, there is contrast in the answers. Combining these two categorizations, we have 4 categories: strongly recognized, strongly but controversial, weakly but controversial and weakly recognized factors.

We have analyzed the frequencies of the sample about the roles and the companies. Frequency analysis was also used to assess the recognition of the factors (strongly, strongly but controversial, weakly but controversial, weakly). The same procedure was used to rank the areas of improvement. For this one we could show the distribution of the answers in a cross-tabulation. This would help answering RQ1.

### 3.5 Spread perception

The perceived spread of a factor is based on the respondents’ experience that the factor is not only local (for example present in the respondent’s team) but it affects also other parts of the organization (Fig. 2). In fact, the respondents might be in contact with other units or might have worked in other units: for example, a system engineer, a manager or a Scrum master is contact with other teams at the same time and might find the same problem described for his/her current team. By answering “yes, in my and in other teams”, the respondents provide direct experience about one team and indirect experience about other teams as well. The information about the surrounding environment of the respondents expands the coverage of our investigation of the factors in the organization. This information is useful for further prioritizing the factors, and to mitigate the threat that the factor is present only in the unit of the respondents.

In order to evaluate the spread of the factors through the organization, we weighted the 4 answers provided by the respondents. We obtained ordinal data similar to a Likert scale without a middle-point. We have weighted the answers with the following weights: 0 → no, 1 → yes in other units/teams, 3 → yes in my unit and 4 → yes in my unit and in other units (1+3). We have weighted the answer yes in my unit as 3 times yes in other units: this way we “trusted” the direct experience more than the participants’ perception about the presence of the factor in other (close) contexts. We have accentuated the polarity between the “no” (0) and the “yes” (3): means around 3 would show strong recognition, means around 0 strong negation, and means around 2 would show high polarity in answers (not many mild answers). The 1 point assigned to in other units weakly influences the means and is useful to give us an idea of how much the factor is perceived as spread throughout the organization. We could have assigned a higher weight to in other units and the results would have given a stronger result on the spread of the factors, but we have chosen to be cautious.

---

**Table 1. Challenges (factors) studied**

| Q23 | Loss of knowledge about a reused framework’s variation points, for example a framework created some years. |

---
4. RESULTS

4.1 Sample Frequencies

We obtained 36 complete answers. The web tool helped us selecting only the answers that came from respondents who finished the whole survey, i.e., incomplete answers has been discarded. We counted 4 respondents for company A, 22 for company B and 10 for company C.

Fig. 3 shows the distributions of answers across roles. We see that all roles were covered (“system” stands for system engineers). Company B and C respondents cover all roles, whereas company C respondents covers all roles except testers. We notice a major presence of designers, especially for company C.

4.2 Factors Recognition and Spread

Table 2 shows the overall recognition of the factors, coded by Q01–Q23 in the leftmost column.

Table 2. Factors ordered by their recognition and spread, annotated if dependent on context or role.

<table>
<thead>
<tr>
<th>FACTOR</th>
<th>RECOGNITION</th>
<th>RANK WITH SPREAD</th>
<th>DEPENDS ON CONTEXT</th>
<th>DEPENDS ON ROLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q04</td>
<td>93.9</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q01</td>
<td>83.3</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q00</td>
<td>84.2</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q07</td>
<td>85.3</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q14</td>
<td>86.1</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q09</td>
<td>84.8</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q12</td>
<td>60.6</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q05</td>
<td>77.8</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q03</td>
<td>75.0</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q15</td>
<td>64.7</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q11</td>
<td>61.1</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q19</td>
<td>55.6</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q22</td>
<td>60.0</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q23</td>
<td>54.5</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q18</td>
<td>54.3</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q06</td>
<td>52.8</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q02</td>
<td>54.3</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q16</td>
<td>50.0</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q20</td>
<td>52.8</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q10</td>
<td>50.0</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q17</td>
<td>38.1</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q13</td>
<td>39.2</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q10</td>
<td>20.0</td>
<td>Strong</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The second column from the left (Recognition) displays the % of respondents (calculated on the valid answers provided) that have recognized that factor. All the factors are recognized at least by 29% of the participants. In the middle column we highlighted the level of recognition (strongly, strongly but controversial, weakly but controversial and weakly recognized), and a bold border groups the rows of the factors included in each level. 9 factors, with more than 75%, are strongly recognized. The factors with more than 50%, other 11, are strongly recognized but

We also wanted to derive properties on the factors, according to RQ2 and RQ3: were the answers related to the role or the company of the respondents? To discover if the answers on the factors and on the areas of improvement depended on the company, we have used cross-tabulation, and performed a Pearson chi-squared test to check if the distribution of the answers changed with respect to the companies B and C (we couldn’t use A because of the few respondents). For factors that gave a significant response (p-value < 0.1), we know that there were differences in the expected distribution. Then, we analyzed the tabulations to recognize the mismatch. We performed the same test using the roles as categories. Finally, we combined the categories, roles and companies, in another cross-tabulation, but the single cells contained too few entries to perform a useful analysis. We also ran, in a cautious way (not all the assumptions were covered by the data), the one-way ANOVA test of variance between the means of the answers. This step mainly confirmed the results from the chi-squared test. However, for the relationship between factors and roles, this test highlighted some factors that we have decided to manually analyze. We could use all these results to further validate the ranks or to warn about those factors for which it would be better to consider context-related factors, or that are not recognized by some roles.

3.6 Recommendations

We analyzed the results with the help of the qualitative data provided through the open-ended questions, which allowed us to infer relationships between the factors and the categories, in order to define hypothesis of solutions. We also considered the factors with the ASD perspective in order to understand the influence of such practices and principles on the factors, or vice-versa if the factors seem to affect ASD.

3.7 Validation

We ran a validation workshop with other employees, at least one for each company, where we showed a first prioritization of the factors and some of the recognized practices. The response was positive. One of the practices that we found during the qualitative analysis was questioned, and therefore we have left it out from the results.

Furthermore, we occasionally showed the results during other interviews (for other studies) in the same companies (when relevant). The feedback has always been positive and confirming.

![Figure 2. Using the respondents experience to extend the sample.](image)

![Figure 3. Sample of the respondents, divided by roles and companies.](image)
controversial, which means that even if there are more “yes” than “no”, there is a clear contrast in the answers. The last 3 factors are weakly recognized but controversial, which means that the number of “no” is more than the number of “yes”, but the percentage of the latter ones is close to the former ones. We don’t have any weakly recognized factor.

In the middle column (Rank with Spread) we have showed the spread calculated using the means of the answers weighted from 0 to 4 as explained in section 3.3: such results show, together with the recognition, how much the respondents perceive the spread of the factor around them. This changes partially the order of the factors, as we can see for Q14 and Q01: the former one has a higher recognition rate, but the latter one is evaluated to be more spread through the company, and the two aspects are summarized in the means. Notice that the spread doesn’t influence the previous categorization of the factors (strongly, weakly, etc.).

In the last two columns from the left (Depends on Context and Depends on Role) we have marked the factors with a “x” when we have statistical evidences that the factor has the property, with “!” when we have evidences that have to be further confirmed.

4.3 Do Factors Depend on Context?
The distribution of the answers for the factors rarely changes with respect to the respondents’ company. This means that we couldn’t reject the hypothesis that the recognition of most of the factors depended on the company of the respondents but for three of them (Q03 Q07 Q23). The identified factors are reported in Table 2 with “x” on the column Depends on Context. In the following we describe the differences among the companies.

Q03: Project-related bugs or defects

For factor Q03 there is a great difference between the responses of company B and C: the respondents in company C don’t recognize the factor, while the ones in company B strongly recognize it.

Q07: There is no time to improve parts of software shared among projects

For factor Q07 the respondents in company B strongly recognize it, while the other ones stand in the middle. This shows that this factor is very present in company B while is controversial in the other companies. With respect to Table 2, this partially weakens the strongly recognition of Q07 since the mean is influenced by company B.

Q23 - Loss of knowledge about a reused framework’s variation points, for example a framework created some years before.

Company B recognizes this factor more than company C.

The results from ANOVA gave the same results.

4.4 Do Factors Depend on Roles?

We found that only few of the answers for the factors changed when given by different respondents with different roles. These are marked with “x” in Table 2 in the column Depends on Roles.

Q05 - There is an upcoming product. Erroneous assumptions have been made on what part of the existing software can be reused and/or adjusted, causing inaccurate budget or resources allocation (for example time or workload)

For this factor, managers and system engineers show a clear propensity to recognize the factor, while designers and testers gave more controversial answers.

Q10 - A development unit was forced to integrate a common component (shared with other units). This caused communication problems and now the unit is not willing to integrate new common components.

Some results from the ANOVA tests have highlighted interesting differences among the means of the respondents on the following factors. These evidences, though, have to be read with care: they can be considered hints for further research more than strong evidences. For this reason we have marked these factors with “!” in Table 2.

Q06 - Evaluation of costs and feasibility of new features don't take into account implementation issues and constraints, causing inaccuracy in budget and resources (for example time or workload) allocation

The recognition of this factor changed greatly between System Engineers and Designers: the difference between the means was of 2.29, which was significant at 0.036 (<0.05). Designers didn’t recognize the factors. The other roles stand in the middle.

Q12 - Different attitudes and values of distributed (not colocated) teams (or units, projects) caused communication problems (tensions, synchronization problems, overheads, delays, misunderstandings, etc.)

For this factor, some designers seem to not recognize the challenge (1.77), while the other roles show agreement in its recognition (3, 3.29 and 3.3).

Q13 - Team’s (or unit, project) lack of will to integrate a common component (a component shared with other development units)

For this factor, designers and testers don’t recognize the challenge while managers and especially system engineers have experienced the problem (even if there are controversial opinions).

4.5 Improvement Areas

In the table below we show a prioritized list of the improvement areas: each participant ranked the areas from the less important to the most important, and we valued the positions respectively 1-5. The values in the Fig. 4 show, for each area, the mean of the importance given to it by the respondents. The ranking is sharply defined by the following means: People, Architecture, Organization, Processes, and Technology.

Figure 4. Prioritized list of the areas of improvement

The distribution of the answers is given in Fig. 5. It’s clear from the results that in order to improve communication challenges related to the mentioned factors, the respondents suggest to invest into resources related to people (mean 4.11 in the Fig. 4, clear distribution on most important in Fig. 5). This encompasses developing attitudes, values, mindset, knowledge and social skills. The second in the ranking is Architecture, whose mean is greater than 3 and the distribution of the answers are clustered around 3 and towards 5. As for Organization and Processes we cannot sharply recognize the most important one among them, given also the quite homogeneous spread of the answers across the values.
The last in the rank is Technology, with the answers clustered on values 1 and 2, which denote a clear suggestions from the respondents to not consider this area as important for solving the mentioned communication factors.

![Figure 5. Distribution of the answers](image)

**4.5.1 Improvement Areas and Companies**

In the previous section we have analyzed the importance given by all the respondents to the different areas of improvement. However, we want to see if this choice varies depending on the respondents’ company. This way, we could relate the choice to different contexts. However, from Fig. 6, it’s quite clear that the answers are aligned among the companies, i.e. the ranking is homogeneous.

![Figure 6. Distribution of the answers](image)

**4.5.2 Improvement Areas and Roles**

The results for improvement areas and roles are aligned as well. The only difference is in the Organization answers: Managers put this area as less important, while for most of the testers, the area is the second most important one. Designers and system engineers gave mixed results.

5. DISCUSSION

In this paper we have studied the relative importance of communication factors influencing reuse and speed when employing ASD in large organizations. In a previous paper (SPLC'12) an explorative multiple-case study recognized an extensive list of such factors (114). The factors were categorized using a hierarchy of areas of improvement: one of these, “Communication”, contained a substantial amount of factors that were also common to all three organizations. In this study we have applied an in-depth study on 23 of those factors, in the same organizations. Our findings strengthen the results and lead suggestions for how the challenges should be managed. Highlighting the most recognized and spread factors should help practitioners selecting the target for resource investments and provides guidelines for further research. The relationship between factors and context should serve as a warning to practitioners aboutknown issues in applying ASD and SPLS in determined contexts. The different views among the roles on some factors reveal existing conflicts or lack of awareness of some issues.

**5.1 Strongly recognized factors**

The analysis of the frequencies provides a strong validation for 9 factors. These ones have shown to be quite generalizable for the studied cases (with the exception of Q07 and Q03, that might depend on the context). This means that practitioners could consider those factors as very likely to hinder the combination of reuse and speed, and therefore they could act in advance to avoid them or counteract. As for research purposes, some of the factors represent an open issue.

The strongly recognized factors (visible in Fig. 7) highlight different communication problems affecting the interfaces between the development team and other parts of the organization. ASD requires frequent feedback. In order to keep the development iteration, SPLS needs coordination for maintaining reference architecture and consistent reuse across projects and products. The results show that, in order to combine these two strategies, the following interfaces between Agile teams and other organizational entities need particular attention:

- **Team - System engineers/Architects.** Q04 is the most recognized factor, almost by all the respondents (93%, which include many system engineers): system requirements should be continuously discussed between system engineers and the Agile team. Software reuse has to be recognized at a system level to be implemented also in software components. Variability management starts from requirements. According to the qualitative data, (partial) co-location between system engineers and the Agile team would mitigate or solve this factor: the Agile team needs explanation of the reference architecture and agreement on the requirements, for example in understanding which ones are fixed and which ones are not (and could be changed by system engineers to ease software development). Since we have studied only companies producing embedded systems, we can infer that this factor is generalizable in this context, but we can’t generalize it for pure software developing companies;

- **Team - Product development and management:** the respondents’ answers about factor Q01 and Q02 (processes-related communication) contrast visibly. It seems that ASD suffers particularly when has to interface to the product development rather than with the project one. This aspect might be connected with the kind of developed product (i.e. embedded systems): hardware’s design and development process follow a more waterfall-oriented model, which conflicts with the iterative one claimed by ASD. In general, software development might be bound to the surrounding product development process, which hinders the quickness that would be gained by employing ASD. However, we cannot say if this is a real constraint or a factor related to the legacy of the product development in place. In the latter case, we hypothesize (supported by the qualitative answers) that a change in the product development process to make it more ASD-friendly would foster the teams, e.g. by providing frequent interfaces for strategic input. On the other hand, Agile teams should coordinate with the overall strategy (e.g. SPLS), and the local process (as well as the attitude of the team) should be adjusted to consider frequent checkpoints for strategic input (as also suggested in [7]).

- **Team - Distributed teams** (Q08): dependencies with not co-located teams might hinder the speed of the Agile one. This includes outsourced components but also teams distributed in different buildings. Especially, when the quantity and the channels of communication are constrained by contract terms such as time intervals or approval requirement, strong delays occur hindering speed. The same, but less drastic problems, occur by mismatches in processes, practices and attitudes and values. Frequent meetings on-site and social network
activities would ease this interface. The SPLE strategy for developing loosely coupled components seems to decrease this factor. However, the focus of the Agile teams on features (customer value) instead of components brings another kind of dependency: two teams might work on dependent features (which requires interaction and communication between the involved teams) even if the components have been architected as loosely coupled (see “Architectural rules for Agile”).

- Team - Other projects’ teams: Q07, Q03 and Q09 are strongly recognized, but the results have been stronger for a specific context (Q07, Q03 for case B). The isolation of the projects, especially in terms of budget and resources, create silos that hinder the reuse across the projects. The exclusive focus on one project by the team leads also to hindering the communication (and speed) among teams in different projects: this creates inefficiencies, especially if there are dependencies among the projects, and strategic reuse always brings such dependencies. Solutions such as relocation in different projects have been proposed in the qualitative answers, to spread knowledge of the system and to gain acquaintance among the employees. The relation with the context will be better discussed in section 5.3.

- Team - Sales unit: factor Q05 shows that software reuse can be erroneously considered during the selling process (or the marketing scope). In the qualitative answers, respondents specified that no resource allocation was estimated when reusing similar components. However, they specified, the value of identical components is higher than of the similar ones, because the modification of a component usually brings many side effects. This, on a SPLE perspective, suggests that full reusability of components should be checked by the sales unit. This would encompass communication with the involved teams responsible for the components. The strong recognition of this factor mainly by management and system engineers shows that designers and testers are not fully aware of strategic and business related goals. This suggests that if the Agile team is not guided continuously by strategic inputs to coordinate with the overall strategy, it might incur in not optimal decisions.

The remaining factors, and the prioritized Areas of Improvement suggest also the following improvement targets.

Architectural rules for Agile: the factor Q09 shows that Agile teams developing different components (or features) need to be decoupled as much as possible. This suggests, in the reference architecture, the need of architectural design rules to match the organizational (Agile) structure with the architectural components. With the introduction of feature-oriented teams, the component-oriented architecture might cause loss of responsibility for components, causing loss of architectural care. For component teams, the spread of features across different teams would increase the number of interactions among the teams for requirements and design agreement. In summary, there is a clear conflict between feature-oriented and component-oriented focus. Changing the conception of reference architecture might be one possible solution, as also suggested by the respondents with their preference given to Architecture as second most important Area of Improvement. In particular, new architectural solutions should aim at combining the following goals:

- Avoid organizational dependencies caused by the architectural ones (loose coupled Agile teams);
- Satisfy system requirements but adapt and take into account information about software design.
- Include reusability as important quality (through SPLE).

The implementation of a suitable overall architecture (such as a reference architecture in SPLE) compatible with ASD remains an open issue in research [4].

Face-to-face training. Q14 also highlights that the reduction of documentation in favor of face-to-face communication claimed by ASD needs to be complemented with an institutionalized educational training period for external employees (newcomers or consultants), which has to involve the Agile teams themselves.

Communication system to feed Agile teams. The factors mentioned above clearly point out that large organizations developing product lines and wanting to use ASD need to put in place (or need to have already) a communication system that feeds continuously the Agile teams with architectural and strategic information to keep them running. Reuse is involved in both architectural and strategic choices: thus, the reuse of software, especially with ASD, is currently suffering from the lack of communication (flow of information) between the development team and the system engineers and the product management.

Consider the time for transition. A related issue, mentioned by the respondents, would be the amount of time required to set up a working Agile environment. In some cases it could take a lot of time (years) to create efficient teams and to create communication bridges (links) with other teams, projects, system engineers and product management if not already in place.

Communication in education. It’s important to notice that the respondents strongly agree that the solutions have to involve employees’ mindset, communication skills, attitudes and values. This leads to consider communication as an important educational objective, which involves training internal to organizations and practices to connect employees that are going to communicate (suggested in the qualitative answers). Communication should be considered in academic environments as well. In fact, these results show that even though software is a very technology-oriented product, it needs social skills to be developed in large contexts. Therefore, such skills have to be developed during the academic training with courses or cross-course learning objectives.

5.2 Controversial factors

Other 11 factors were validated by an amount between 50% and 75% of “Yes” answers, which can be considered a good validation result (strongly recognized factors), although the respondents showed mixed opinions (controversial). The recognition frequency for the remaining 3 factors was still greater than 29% (weakly recognized, controversial): we could consider them as uncommon factors that may occur but in special circumstances or may not be recognized by some kind of employees. It’s worth mentioning that some discordant views seem to depend on the role of the respondents.

Strategic input for Agile teams. For factors Q06, designers strongly disagree (not recognizing the factor) with respect to managers and especially to system engineers. One possible explanation would be that designers do not see the mismatch between the estimated and the real costs and resources allocation, which would be in alignment with the conclusion drawn about
Q05 (see 5.1, sales unit interface). This is also in line with the hypothesis that Agile teams suffer from lack of strategic information or objectives, which hinder their awareness and therefore the appropriateness of their strategic decisions.

Especially for Q10 and Q13, which are about the frictions generated by the integration of a reused component in another system, there are differences according to the role. In both cases it was the designers and the testers who strongly negated the factors. A possible explanation is that the surveyed units didn’t have the problem, while system engineers and managers, being in contact with more teams, have experienced the factors in a different situation.

5.3 Generalization and contextualization of factors

The results show an overall recognition and spread of the factors. 20 factors have been strongly recognized of which 11 controversial, and only 3 were weakly recognized (but still controversial), while no weakly recognized factors have been found. The low amount of weakly recognized factors strengthen the results and the conclusions highlighted in [12] for the communication factors. These results allow us to make a further step toward the generalization of the factors. In fact, there are convincing evidences that companies B and C equally agreed on the recognition of the factors. With less evidences, we can consider company A on the same line. This means that Table 2 is equally informative for the companies.

As for Q03, Q07 and Q23 the recognition varies according to the context, and Table 2 shows this together with the ranking of the factors. These factors are very related to context B. In this company, in fact, projects were isolated among them hindering the employment of an overall strategy for reuse. The different parameter is business: individual customers with different needs and different time constraints may strongly hinder the investment needed for the creation of a software product line. This is shown also by the recognition of Q23 mainly by case B: according to the qualitative data, the business value of the ad-hoc reuse was overestimated. If the component is not reused exactly as originally implemented, even with slight changes, it would prove inefficient. If the component is considered as equal by the sales department, this would cause a mismatch between the estimated budget and the return of investments.

5.4 Limitations and Threats to validity

We are aware of the following limitations to the study: the ordinal data gained by the closed-ended questions is based on the employees’ perception about the presence of the factors in their unit or in other units, which cannot be considered a precise measurement. However, we have used appropriate statistical tools such as cross-tabulation and chi-squared analysis of distribution. ANOVA has been used only for explorative purposes and we haven’t based strong conclusions on the results, since the required assumptions were not covered by our data (although the method is robust against violations). Randomization of the sample cannot be completely assured since the respondents have been conveniently selected by the companies themselves. However, we have explicitly recommended our contacts in the companies to not influence their choice.

The qualitative information is based on employees’ statements and may be biased. However, the statements don’t usually conflict with each other. On the contrary, the codified concepts are often supported by different respondents and from different cases. Moreover, we have mainly discussed and drawn conclusions on factors that were strongly recognized, minimizing the possibility errors. The hypothesis and suggestions for improvements have been selected among the most cited improvements coded during qualitative analysis. Another possible threat to validity is research bias due to predefined categories, Architecture, Technology, Processes, Organization, and People. However, during the qualitative analysis of the open-ended questions we have intentionally chosen to follow a different categorization (theme analysis) and afterwards we have connected to

We consider the threats to validity as presented in [16] and [20]. Possible threats to construct validity are: mono-operation biased and background influence. We have handled these by collecting data from three companies and making sure that the respondents were distributed in various roles, from managers to programmers. We limited the threats to conclusion validity such as instrumental flaws and influence posed on the subjects by running pre-tests with colleagues from academia and contacts in industries. As for internal validity, each respondent has taken the survey just once, while boredom was avoided by the possibility of interruption and continuation in a later moment. As for external validity (generalizability), the chosen cases allow us to generalize some of the factors with respect to the commonalities in their contexts: we have further discussed this aspect in section 5.4. By studying three cases we have partially limited external validity.

6. Related Work

Part of the motivation for this study has its origin in a previous study [12] (see introduction and section 3).

Lindvall et al. [10] have collected experiences from large companies (similar setting) in order to study the integration of ASD with standard processes already in place. The focus has been in tailoring XP to suite the standard process and quality management, while “Cross-team communication support” mentions the presence of challenges in inter-team communication as an open issue. We have studied more in depth some of those challenges.

Karlstrom and Runeson have studied the application of XP in stage-gate project management [7]. They conclude that such combination is feasible but the Agile team should interface with the gates, while at the same time the project management should adapt to support informal communication and documentation with the XP team. This is in line with our suggestions for improvements of interface between ASD with product management and system engineers.

Kettunen and Laanti [8] studied the large-scale organizational agility and proposed a framework for a company to embed an agile software project in a successful way. The need for an external process adaptable to the Agile team is stressed, which confirms our findings. However, the paper is not focused on communications.

In [18] Strode et al. provide a theoretical model for evaluating and applying coordination between co-located Agile Projects. An important part of coordination is provided by communication, which in the paper is not explicitly mentioned.

On the basis of a literature review on ASD, Turk, France and Rumpe, give their perception about what the limitations of the ASD are [19]. They mention Limited support for building reusable artifacts, describing the importance of reuse practices, which bring especially long-term benefits. However, the authors don’t address the issue. Our work focuses on this problem and analyzing the communication factors related to it.
In [15] the focus is similar to ours, but the study has been carried out in one organization with different context from ours (embedded systems), performing qualitative data analysis, while we have surveyed the factors in a quantitative way in order to have a prioritized list of the most important ones. Moreover, we have studied specific communication factors related to speed and reuse, which is not included in the scope of [15].

In [13] Paasivaara and Lassenius present a set of communication practices for Global Software Development. However, the study is not related to ASD. Some practices may be used to ease the factors that we have studied, but have to be adapted to an ASD environment.

7. CONCLUSIONS

We have studied 23 communication related factors in a multiple-case study in large companies developing embedded systems. We have provided a generalization of 9 factors, strongly recognized and perceived as spread through the organization. These show how mainly inter-organizational communication factors hinder speed and reuse (RQ1). We have recognized 5 interfaces with the Agile development team that need to be improved (RQ4): system engineers (architects), product management, distributed teams, inter-project teams and sales unit.

The most important improvements include:

• Agile teams need strategic and architectural inputs, which requires the (time consuming) setting up of links with system engineers and product management to support reuse and to fast (RQ3).

• Academic and industrial training as well as different tactics for co-location. Solutions related to social science need to be further explored.

• Architectural solutions to combine focus on customer value of the teams, reusability, system requirements and avoidance of organizational dependencies.

Inter-project communication factors are connected with the business strategy of the company: the time gained to serve a single customer will cause losing reuse opportunities among projects (RQ2). Finally, we have provided a prioritizing display (RQ1) showing strong, controversial and weak factors ordered by their spread and annotated as context dependent or recognized by selected roles, which makes the results more applicable for practitioners and provides guidelines for further research.

ACKNOWLEDGMENTS

This research has been carried out in the Software Centre, by Chalmers and University of Gothenburg and Industry Partners. We thank Ali Shahrokhni for suggestions on the data analysis.

REFERENCES


