Challenges in Enterprise Software Integration: An Industrial Study using Repertory Grids

Heidi J. Rognerud and Jo E. Hannay
Simula Research Laboratory, Dept. of Software Engineering, Pb. 134, 1325 Lysaker, Norway
Univ. of Oslo, Dept. of Informatics, Pb. 1080 Blindern, 0316 Oslo, Norway
heidijr@ifi.uio.no, johannay@simula.no

Abstract

To identify and systematize software practitioners' perceptions of a problem is an important first step toward analyzing and searching for a solution to the problem. This paper reports on an industrial study, in which the repertory grid technique was used to elicit practitioners' perceptions of key challenges in the company's software integration practices. The perceptions of a total of nine practitioners from three organizational groups (Developer, QA Manager, Project Manager) were elicited and analyzed. We found that perceptions differ markedly between groups, but that on some issues, there is a consensus across all groups. Three types of challenges were identified as critical, causes, and easy to handle, namely responsibility, requirements, and knowledge. The elicited information may be used to plan process improvement for integration projects in the company, and may also be used in building a general ontology for integration challenges and their solutions.

1. Introduction

Quite often, when discussing issues in an organization, people start out by speaking at cross purposes, and it may take several meetings before a common understanding of an issue and its key components has been reached. Our experience is that the process of arriving at such a common understanding is often tacit [2], and it might be unclear to the participants that such a process has taken place, or even that there were diverging conceptions in the first place.

Software professionals discussing process improvement issues are no different. In a comprehensive study [3], Baddoo and Hall found that people in different organizational groups in software development organizations have diverging views of the concepts and importance of software process improvement. They also found differences between groups in how each of the other groups' roles in software process improvement were perceived.

Arriving quickly at a common understanding of the salient concepts of a problem is a key point in problem-solving techniques, such as the Productive Thinking Model [21], the Plan-Do-Check-Act cycle [39, 40, 6], and the Goal-Question-Metric approach [4]. It seems reasonable to assert that early effort spent on clarifying concepts will lead to greater overall efficiency in problem solving.

In this paper, we report from a study conducted at a major software vendor, Company X. Company X develops large-scale enterprise software systems. Often, situations arise where their in-house products need to be integrated with each other, or where third-party software needs to be integrated with their in-house products. These integration efforts were perceived as problematic from several perspectives within the company, but it was not clear what the exact pain points or challenges were, what causal relations existed between the challenges, or which challenges one should address prior to other challenges. It was also suspected that people with different responsibilities in the company would have different perceptions on the challenges of integration.

We therefore conducted concept elicitation sessions on professionals involved in the company's integration work. We used the repertory grid technique for concept elicitation [10, 23, 41] on a total of nine professionals from three different organizational groups: Developer, QA Manager, and Project Manager. The responses were analyzed both individually and on an aggregated basis within and across groups. This gave a picture of the main challenges, their relative importance, and their causal relationships.

Section 2 summarizes notions for the integration of enterprise software, Section 3 describes our research method for the study, Section 4 summarizes the findings of the study, and Section 5 discusses impacts and concludes.

2. Enterprise Software Integration

The IEEE Standard Computer Dictionary defines integration as "the process of combining software components, hardware components, or both into an overall system" [22]. The type of integration of interest to Company X is that of "making applications work together that were never intended to work together by passing information through some form of interface" [16], i.e., making two or more independent systems able to communicate and share information. Hardware integration challenges fall outside the present scope of interest.

2.1. The Need to Integrate

In general, the demands of shorter time to market and faster processing of business requests exert a constant pressure to improve the workflows in an organization. Improved workflows require, as a prerequisite, at least the following:

- that information is available whenever it is needed
- that information is always correct and consistent

Various departments in the organization might need to access the same information, but might not have access to the same systems. As a consequence, information may get stored in several systems with the risk of generating inconsistent data. If these systems are not integrated properly, further consequences may be poor business processes and unforeseen costs when reconciling the data [34]. Users of a system require quick access to data, regardless of whether the data is spread out in several systems. For example, bank customers expect diverse information about savings, stocks, and payments to be presented on one screen [35].

Software vendors that develop enterprise systems therefore try to design their systems so as to minimize the need for post-installment integration in the customer organization. For example, in the 1990s the Enterprise Resource Planning (ERP) system was introduced, whose purpose was to support activities such as finance, human resources, production, supply and distribution in one and the same system.

Enterprise systems such as ERP quickly became popular, (between 1996 and 1997, there was a 40% increase in the ERP market [32]), and enterprise systems have to some degree succeeded in meeting the needs for shared data across an organzation. However, the "one system for all" solution introduces integration problems of its own. It turns out to be very hard to meet all the needs of the different departments of a company within a single enterprise system, and an organization may have several legacy systems that are difficult to replace. Moreover, in a survey carried out to identify the problems with ERP systems, several of the companies reported that they "faced serious problems with their business strategies, as the ERP system imposed its own way of doing business" [42]. Thus, an enterprise system will normally exist alongside other systems of the organization, which again, may cause overlap in functionality and duplication of data. The aforementioned study states that "The most serious problems focus on the integration of the ERP solution with existing applications such as legacy systems (82%), or with new business software (e.g. supply chain management, e-commerce applications etc) (46%)" [42].

As a result, focus has shifted to integrating the smaller systems of the organizations with enterprise software [17]. Several software vendors try to pre-integrate, e.g., third-party software with their enterprise systems before shipping the integrated package to customers. This is the type of integration that was addressed in Company X in our study. In Company X, the enterprise system with which other systems need to be integrated, is most often the Customer Relationship Management (CRM) system.

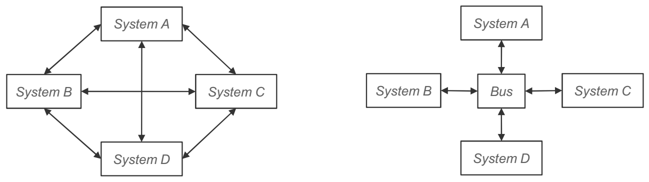


Figure 1. One-to-one versus bus integration

2.2. Integration Techniques

There are two archetypical modes of integrating systems (Figure 1). The first is to integrate all the systems directly with each other, i.e., one-to-one. This will result in n^2 connections for n systems.

For one-to-one integration, communication between two applications is commonly handled via shared files. Before communicating, the applications need to agree on file format, file location, locking mechanisms, and ownership of the file. Currently, XML is used as a standard language for file-based integration [17]. Most of Company X's integrations are one-to-one based on shared files.

An advantage of this approach is that it minimizes what the integration developer needs to know about the applications. File sharing also allows the applications to be altered as long as they produce consistent file formats. This gives loose coupling, which for integrations means to "reduce the assumptions two parties...make about each other when they exchange information" [17].

A major issue with file transfer is that, in practice, the data in the files might not always be up-to-date. There is a large cost of administrating the files if new files are produced every time the data in one of the application is altered. Therefore, new files are usually produced at certain intervals. However, if one of the applications alters its data and does not write the new data to a file immediately, the other system might read and process obsolete data.

The other mode of integration is to develop an intermediator (integration bus) to which all systems are connected. All systems will only communicate with the bus and the bus will make sure that a message will reach the system it is intended for, and in the correct format [17].

Advantages of using an integration bus include that the systems are ensured to be loosely coupled and that the risk of reading obsolete data is eliminated. Messaging requires less administration, and it is possible to update the data whenever it is needed. Further, an integration bus ensures that integration is dealt with in a uniform manner and at a designated location. This also makes the task of integration an explicit activity with clearer responsibilities.

A main disadvantage is that integration bus downtime may entail downtime for all systems at once, i.e., massive systems failure. It is also costly to set up an integration bus, and resources need to be allocated specifically for the purpose of developing the bus. It may be a challenge to convince senior management to allocate resources for such efforts since they are not tied specifically to the development of a sales product. This relates to the general problem of defending capital expenditures (CAPEX) when operational expenditures (OPEX) are the main priority. The integration bus solution is, however, growing more and more popular, and Company X in this study is currently developing an integration bus for their purposes.

Combinations of these two modes are common, i.e., some systems/applications are integrated through an integration bus, while others are integrated one-to-one.

3. Research Method

The research method used for this study followed the principles below. These principles are intended to apply relevant scientific research methods to highly relevant industrial challenges in a more flexible manner than in traditional large-scale research programs. These principles are complementary to the principles of *evidence-based software engineering* [7, 30].

3.1. Research Principles

We adhered to the following principles:

1. *Relevant, concrete and general interest*: The research objective is initiated by a desire to solve an important concrete challenge in a software development company. The challenge is perceived to be of general interest to the software industry in general.

2. *Practitioners' theories*: The research questions are defined and refined by consulting (possibly by eliciting the mental models of) practitioners working on the problem.

3. *Scientific-strength research methodology*: The research questions are answered by applying sound scientific research methods, if necessary, with the help of researchers' expertise on such methods.

Principle 1 is reflected in work by Houdek [20] and Mathiassen [33]. In our setting, Principle 1 was implemented as follows: The topic of integration challenges arose in a meeting between the second author and a senior manager in Company X. In two further meetings with two additional representatives from Company X the following representatives' perceptions were recorded.

1: A disproportionate amount of time is spent on integration, both in absolute terms and relative to the amount of resources allocated to integration.

2: The causes of the large amount of resources spent is probably due to managerial, rather than technical, reasons.

3: Vague responsibilities may be the reason for delays and the large amount of time spent.

4: A lot of things are not specified well enough.

5: There is little time for specification.

6: Standardization is lacking.

7: Too much time is spent on discussing interfaces, and perhaps the data semantics.

8: Little emphasis is put on project management in the integration projects.

9: Very little relevant research into integration challenges from the above points of view seem to be available.

These perceptions motivated our research objective.

> *Research objective*: To lay out and systematize knowledge and perceptions within Company X on what the main challenges of the company's software integration activities are. The inquiry should be conducted on several organizational levels, since it was postulated that major challenges are managerial rather than technical. Results should be fed back into the company, since the information gained might help the company to solve some of the challenges. Results should also be disseminated to a wider audience, since the topic should be of general interest.

Note that the research objective does not address the above nine points directly (e.g., in a hypothesis-testing manner). Instead it pertains to the elicitation of perceptions in a more systematic and stratified manner.

The impetus of Principle 2 comes from work by, among others, Argyris, Gigerenzer, Jarvis, and Schön [1, 2, 12, 24], and is based on the view that practitioners hold valuable implicit and explicit knowledge of their work domain that should give input to scientific knowledge. Since practitioner knowledge is often tacit, various methods for eliciting such tacit knowledge may be employed. The concept of mental models [9, 11, 25, 26, 38] gives a framework to define and elicit tacit knowledge. In our study, Principle 2 not only underlies the research objective itself, but also determined in what way the research objective was addressed.

Principle 3 was implemented by using the repertory grid technique (see Section 3.2 below). This technique is a semi-structured interview technique which originates from psychology, but which has had numerous applications in other domains of research [41], such as marketing, quality control, work training, and software engineering [3, 37]. Principle 2 permeates the repertory grid technique. The technique is based on Kelly's personal construct theory [28, 29] which is founded on the axiom: "Man is a scientist", by which is implied that people try to understand the world and their place in it by constantly testing hypotheses about the world. Though these hypotheses will affect the way we see the world, they will also change over time, given enough information or experiences that contradicts the present construct systems. The repertory grid technique seeks to elicit a person's present construct system. In our context, this translates to eliciting a software professional's construct system regarding software integration.

3.2. The Repertory Grid Technique

The repertory grid technique is a semi-structured interview technique. Its main methodological objective is to let the interviewee determine the salient concepts of the subject matter of the interview, and then to ask the interviewee to rank these concepts in a systematic manner, but according to the interviewee's own value system. We followed the guidelines for the technique described in [10, 23]. In order to facilitate the concept elicitation as well as the subsequent ranking, we asked interviewees to write concepts on 7cm x 10cm laminated cue cards with a white-board pen. Ranking was then performed by organizing the cards relatively to each other. This was inspired by the mental model elicitation approach in [27]. Prior to the concept elicitation, the interviewees were asked to give a 5 minute introduction to the topic of enterprise software integration that should be understandable to non-experts.

The interviews took place during the period from October 2008 to January 2009, and were conducted by the first author. Nine software professionals from Company X were interviewed on the premises of Company X. The interviews lasted from 1.5 to 2 hours. All interviews were audio recorded for the purpose of subsequent clarifications.

3.2.1 Elements

The salient concepts, which are the first things to be elicited in a repertory grid session, are called *elements*. In our context, these are the perceived pain points or challenges of integration efforts in Company X. Each interviewee was provided with a set of blank cue cards and was asked by the interviewer to think of what the interviewee considered to be the most central pain points or challenges regarding integration in Company X, and to write each such element down on a separate cue card. The interviewee was free to write down as many elements as he or she liked. The elements of one of the interviewees (with explanations) are presented in the first half of Table 1.

After the interviewee seemed to have no further elements to present, he or she was asked to pick out the nine most salient elements that he/she had written down. This was intended as a quick quality check. It often turns out that elements are overlapping or hierarchical, and the interviewee was thus given the opportunity to merge, split, or rethink elements. Although hierarchical or overlapping elements are not wrong, it is recommended to avoid this, since the subsequent ranking becomes more difficult for the interviewee [41]. Choosing the most salient elements also limits the number of elements. (The recommended number of elements is suggested to be between five and 12 in most cases [23].) It would be possible to limit the number of elements at the outset by simply asking for, say, the seven most salient elements. However, this would require the interviewee to hold a large number of elements in working memory simultaneously, which is cognitively taxing, e.g., [36]. Writing an unspecified number of elements on cue cards, and then reducing elements afterward, is one solution to this issue.

3.2.2 Constructs

After the elements have been established, the interviewer pools three elements and ask the interviewee to give a characteristic of two of them that distinguishes them from a characteristic of the third one. This generates a bipolar pair of characteristics, which goes under the name of a *construct*. For example, given the elements *Lack of knowledge*, *Who does what*, and *Documentation*, the interviewee might group the first two and contrast them to the third, by characterizing the first two as *Resource related*, and the third as *Information related*. This gives the bipolar construct *Resource related – Information related*. Direct opposites, such as *Resource related – Not resource related* are to be discouraged, since such constructs lack meaning at one of the poles. This process of construct elicitation is repeated with a new set of three elements until a desired number of constructs have been elicited. Note that the same set of three elements may give rise to several constructs. This triadic elicitation method giving bipolar constructs is based on the essence of Kelly's personal construct theory.

The purpose of eliciting bipolar constructs is subsequently to rank all elements on each construct (regardless of whether an element was involved in eliciting a particular construct). Thus, the focus is not on eliciting all possible constructs, but on eliciting sufficiently salient constructs. (In any case, comparing all possible triads would will usually not be practically possible if the number of elements

Table 1. Elements and constructs of Developer 1

Elements

1. *Missing flow of information*: Developers in the integration project work on their own areas and do not discuss their solutions with each other.
2. *Meaning*: It is a challenge to find out what a concept is called and means in the system they are integrating with.
3. *Waiting for technology*: Currently, an integration bus is in development, and a lot of challenges would have been easier if they could have used the integration bus instead.
4. *Low amount of recycling*: Several integration projects require some of the same work, and this is done all over again for every integration.
5. *What information*: It is usually not possible or necessary/desirable to exchange all the information in the systems. Therefore one has to agree on what information to exchange in order to fulfill the requirements.
6. *Errors in external components*: Unexpected errors caused by external components might cause a lot of problems because one does not understand what has happened and what the problem is.
7. *Lack of knowledge*: When integrating with another system the project participants have to acquire knowledge about the other system. This is knowledge you will only use in the current integration. This usually ends with a lot of nagging on the people with knowledge about the system. It might also be that these people are not accessible.
8. *Who does what*: When integrating with another system the developers have to acquire knowledge about the other system. This is time consuming and developers from the other system might have done the same tasks a lot faster because they have more knowledge about the system. It all comes down to the availability and use of resources.
9. *Documentation*: The documentation of systems is not good enough. This is due to the routines of documentation not working as they should and the documentation is hard to find when needed.

Constructs

1. *Usage of resources – Agreement*: Some of the challenges require resources. The reason why they are challenges is because it takes time and resources to solve them. They might be internal or external resources. Other challenges might require discussions to be resolved.
2. *Something we can solve/influence – Something we cannot solve/influence*: There are different levels of how easy it is to overcome a challenge. Some we can influence, while others we just have to evade. The things we can affect are often internal, while the things we cannot affect are external.
3. *Resource-related – Information-related*: Resource-related challenges concern what people are told to do and what they really should do. Information-related challenges concern the structure and maintenance of documents with information about the system and how easy accessible these documents are.
4. *Information spread – Information gain*: On the one hand knowledge is made visible for others. On the other hand, information is obtained for one's own purposes.
5. *Technology – Information*: Technological solutions versus documentation and knowledge about the system (not information in the system).
6. *Information about the system – Information in the system*: Knowledge about the system and how to integrate with it versus what information to exchange and what information is stored in the system.
7. *Active action – Passive action*: Something you do versus something you don't do, absence of doing something, or not acting at all.
8. *Cause – Consequence* (supplied).
9. *Critical – Not critical* (supplied).
10. *Easy to handle – Hard to handle* (supplied).

exceed five or six.) In our case, the triads were usually pooled in a random manner, but it was ensured that all elements were used approximately the same number of times. If there were any combinations of elements that were particularly interesting, these were pooled deliberately, see [10] for further suggestions. A process known as *laddering* [10] was performed if clarification of the constructs was needed. Laddering is the process of going from general constructs to specific constructs or vice versa according to what is interesting for the interviewer. For a given construct one can go from specific to general (up the ladder) by asking "why" questions ("why is it that *Lack of knowledge* and *Who does what* are resource related?"). If a construct is too general and needs to be more specific to give meaning relative to the research objective, one may go down the ladder by asking "how" questions ("how is it that *Lack of knowledge* and *Who does what* are resource related?").

One of the main ideas behind the repertory grid technique is that it should be free from interviewer bias. The nine perceptions in Section 3.1 were therefore not mentioned to the interviewees. Still, such interviews are normally carried out for a specific purpose, which in our case, is to answer our research objective. It is the interviewer's responsibility to guide the interviewee to come up with constructs that relate to the purpose of the interview. One way to guide the interviewee is to introduce the phrase "in terms of" when comparing elements. We thus asked "what separates pain points A and B from pain point C, in terms of how they influence integration work at Company X?". This will guide the interviewee to present relevant constructs without the interviewer actually suggesting constructs [10, 41].

In some applications of the repertory grid technique, it may be pertinent to provide predefined elements and constructs [10, 23, 41]. For our study, three constructs were provided for all the interviewees:

1: *Cause – Consequence*

2: *Critical – Not critical*

3: *Easy to handle – Hard to handle*

These constructs were provided in order to elicit information that may be useful for solving the challenges. The first construct was supplied in order to give an indication of where to try to solve the problems; attacking the cause is, in principle, better than ameliorating the consequence. The intention of the next construct, was to make the interviewee prioritize the problems according to the level of criticality. A possible problem with this construct is that the problems might be rated according to their scheduled priority. For instance, a problem that is critical, but will require a lot of work to solve may be rated as *Not critical* because other problems that are easier to solve will be prioritized. To resolve this issue the third construct was provided.

The cue cards with elements written on them were used in the elicitation of constructs. Three cards at a time were placed in front of the interviewee, who was asked to move the cards around, grouping and regrouping them. Moving elements around physically is meant to facilitate the mental processes that are involved in finding similarities and differences [27]. The constructs of one of the interviewees (with explanations) are presented in the second half of Table 1.

3.2.3 Grid

The last phase in the elicitation session, is to rank each element on each of the constructs. The nine cards with elements were placed in front of the interviewee. The two poles of a specific construct were written on two fresh cards and placed some distance apart. The interviewee was then asked to place the elements relatively to each other between the two construct poles, according to how close the interviewee felt an element associated to one or the other pole, relative to the other elements. The range of the scale used in the ratings can be chosen by the interviewer. In most modern applications of the technique, one normally uses a 5- or 7-point scale [23]. These scales will provide enough points to capture nuances, without giving too wide a range. (The interviewee might find it difficult to differentiate between the elements if the range is too wide.) The use of a scale with an uneven number of points will allow the interviewee to place elements medial to the poles. Sometimes it is simply not possible to determine which of the poles the element is closest to. In such cases the grid would not provide the correct picture of the interviewee's construct system if he or she was forced to choose one of the poles. However, the interviewee was encouraged not to use the mid-point of the scale as a way out when he/she was unsure where to put the element or if the element was not possible to rate on the given construct. Thus, a 5-point scale was chosen for the ranking. Elements that were not possible to rank according to the given construct were left out. For alternatives to rank ordering, e.g., absolute rating, see [23].

After the ranking, each interviewee was asked to give a brief explanation of why he or she had placed each of the elements on the specific rank on the scale. This was done for two reasons. First, the interviewee was given the opportunity to verbalize his or her decisions. Such verbalization, which is intended to reason or explain action (Type 3 verbalization, in Ericsson and Simon's terminology [8]), has been shown to increase performance, e.g., [5]. In all cases, this verbalization resulted in the interviewee making changes to the ranking. The other reason was that the verbalizations were necessary for the interviewer's understanding of the often tacit choices made by the interviewee.

3.2.4 Preparations

In the repertory grid technique, the determination of the salient concepts of the interview are left to the interviewee. However, some guidance should still be given, such as to what the subject matter is (i.e., enterprise software integration challenges), and whether the interviewee should think in terms of, e.g., problem finding or problem solving. To this end, it is important to gain some insight into what sort of responses one might expect in a particular domain (e.g., software development). Prior to conducting the interviews proper, two researchers at Simula Research Laboratory with substantial industrial backgrounds were interviewed by the first author on two topics in software development, namely configuration management and software cost estimation. Based on this, many of the decisions mentioned above regarding the elicitation session were made. Then, one of the research contacts at Company X was interviewed to assure that the technique would provide suitable material and be adequate for the purpose of answering the research objective. Based on this, changes were made to the elicitation process. These changes were mainly on the introductory presentation of the subject matter to the interviewee, the introduction of the three supplied constructs, and the decision to conduct the interviews depth first, rather than breadth first, see [23].

3.2.5 Threats to Validity

The main threats to validity of the study concern the reliability of the elicitation and interpretive processes involved in the repertory grid technique and the subsequent content analyses. Attempts to minimize these threats were made by recording the interviews for context understanding, by feedback to the interviewees, and by dual content analyses.

4. Findings

The data from the repertory grid sessions were analyzed per individual, per organizational group, and overall.

4.1. Analyses per Interviewee

Table 1 presented the elements and constructs of one of the interviewees. The responses of a single interviewee may be analyzed in different ways [10, 23, 41]. In our case, we looked at which constructs that were similar in terms of how many elements that received similar ratings.

For example, for Developer 1 (Figure 2, uppermost diagram), the constructs with the highest score of similarity were *Active action – Passive action* and *Critical – Not critical* (similarity score 80.6%). An interpretation of this fact is that those elements (challenges) that are critical are also those that are active actions in the integration project, and that those elements that are non-critical are also passive actions, i.e., activities that might have made integration projects less challenging but that are not carried out at present. The correlation between criticality and activeness did not apply to the elements *Missing flow of information* and *Lack of knowledge*, which were seen as passive, but critical actions. Further, the constructs *Critical – Not critical* and *Easy to handle – Hard to handle* had a similarity score of 77.8%. This indicates that the challenges that are most critical to overcome in order to succeed with integration are also the ones that are easiest to handle. However, the two most critical elements, *Meaning* and *What information*, are not easy to handle.

Similar results for all interviewees are summarized in cluster diagrams[1] in Figures 2–4. Each cluster diagram displays the constructs with poles on either side of the scoring table. Note that the poles may be reversed to account for negative correlations. The constructs are sorted according to similarity on element scores (numbers in the table). The elements are displayed beneath the scoring table. The dendrograms (trees) to the right are a summary of the similarity between constructs. Each node in the tree is an aggregation of the two subtrees that correspond the most.

Each analysis per interviewee was fed back to the individual, who was given the opportunity to comment on the analysis. This proved very useful for the interviewee, since quite often, the interviewee became aware of implicit understandings and relationships in his/her mental model that he/she had never thought of before. For Company X, the information gained from these individual analyses provided valuable insights into the challenges of integration.

4.2. Aggregated Analyses

An underlying expectation in this study was that perceptions on integration challenges would vary according to organizational group. We therefore conducted aggregated analyses in order to compare the responses of one group to another. A *content analysis* is a systematic way of aggregating individual statements into more general classes of meaning [18, 31]. We conducted a content analysis of both elements and constructs.

4.2.1 Aggregation of Elements

The content analysis of elements was carried out in an exploratory inductive manner following guidelines in [23]. The element cue cards were spread out on a table. If two elements seemed to be related, they were grouped, thus forming an initial inductively formed category. Other elements might be added to such a category, or they might form a new category. This aggregation process continued until all element cards were allocated to a category. Both authors performed this process independently. Inter-rater reliability in this case concerns which categories that arise and then how elements are allocated to categories. Prior to computing agreement scores, we consulted each other's categories and decided which ones corresponded semantically to the other's categories. As expected in such inductive approaches [23], inter-rater agreement was low on

[1]The cluster diagrams were generated in *WebGrid* of the Centre for Person-Computer Studies (`repgrid.com`).

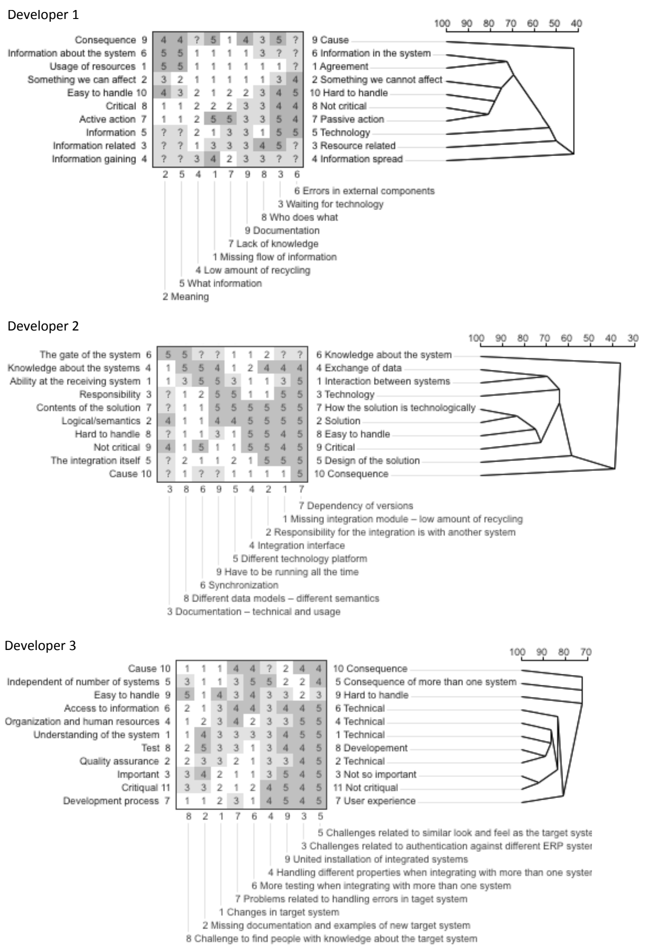


Figure 2. Cluster analyses developers

the first pass: 51.25% when considering both corresponding and non-corresponding categories and 74.5% when considering only corresponding categories. Then, the non-corresponding categories were negotiated upon until a full category set, with agreed-upon names, was obtained. Afterwards, all elements were recategorized independently giving an agreement score of 92.5%, which is acceptable. The remaining discrepancies were discussed and resolved. The resulting categories are given in the upper part of Figure 5, together with the distribution of elements to the categories, both overall and by organizational group.

The categories to which most elements were allocated, were (12) *Difficult communication between stakeholders regarding integration requirements* and (14) *Allocating/acquiring the right personnel/knowledge/skills* (Figure 5). In the latter category, the three organizational groups are represented evenly (via their respective elements). This indicates that this category summarizes challenges that concern all three groups, and that there is agreement that allocating the appropriate human resources is indeed a chal-

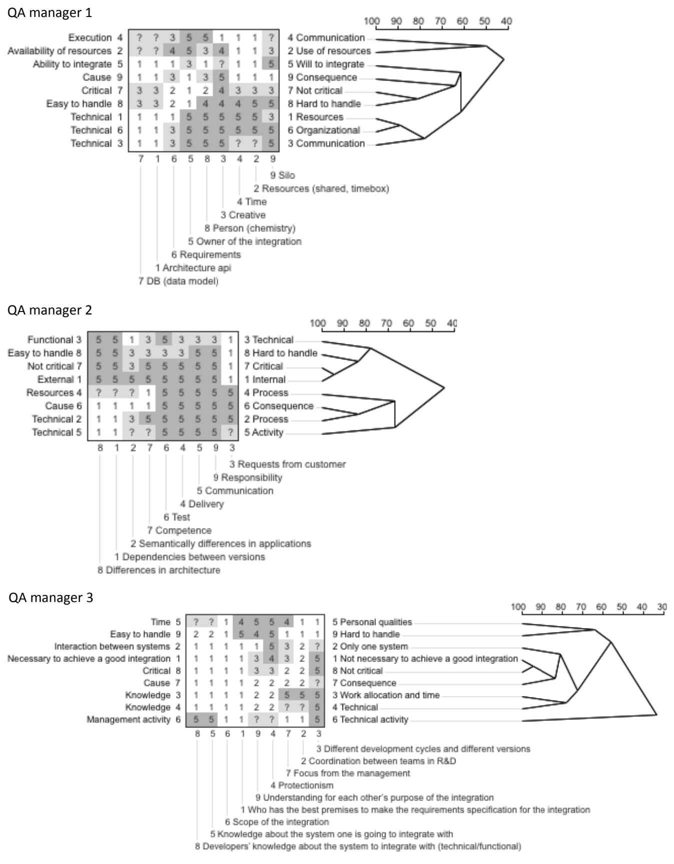


Figure 3. Cluster analyses QA managers

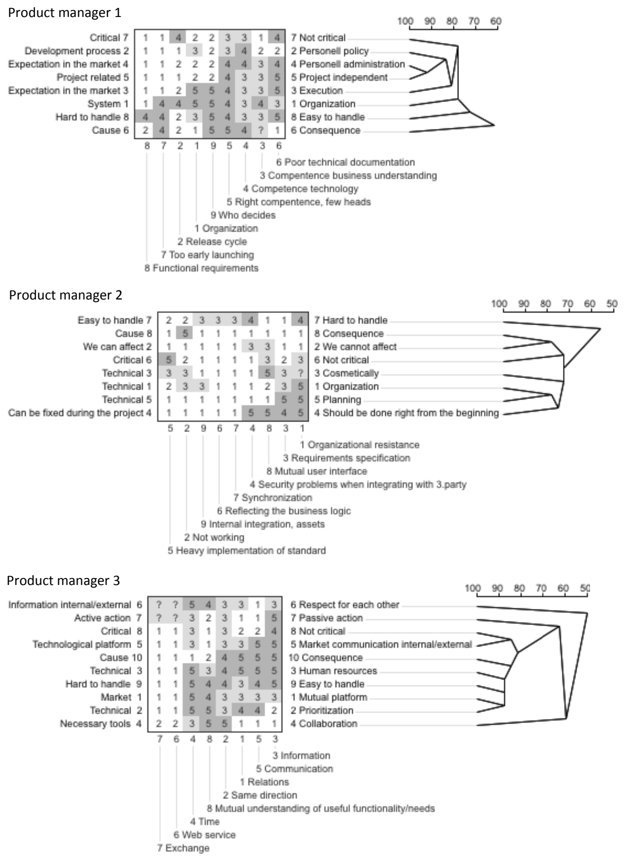


Figure 4. Cluster analyses product managers

lenge. In contrast, the former category (12) only contains elements put forth by QA managers and product managers.

Developers. The categories that contain most elements put forth by developers are the technically focused (5) *Coordination of functionality* and (6) *Coordination of platform*.

QA managers. QA managers have most challenges in (13) *Lack of ownership of integration tasks*, and (14) *Allocating/acquiring the right personnel/knowledge/skills*. They seem less focused on technical challenges and more focused on responsibilities and allocating human resources.

Product managers. Product managers place most elements in (16) *Too product focused/Resistance to integration* and (12) *Difficult communication between stakeholders regarding integration requirements*, which indicates that product management is concerned with challenges that are related to organizational issues and communication.

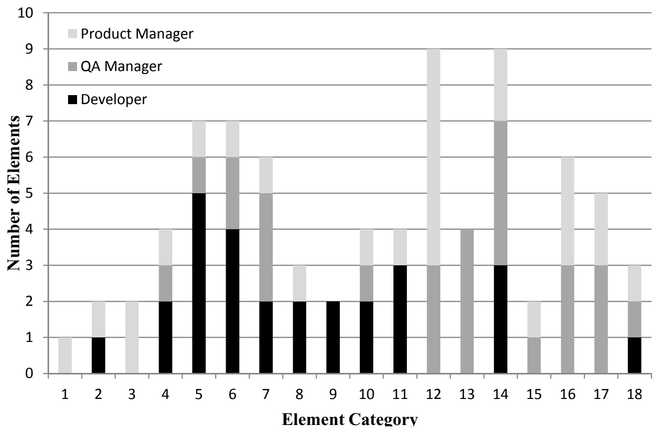
4.2.2 Aggregation of Constructs

The content analysis of the constructs was carried out in a similar manner as for the elements. Inter-rater agreement was 82% on the first pass, and 87.5% on the second pass. However, three categories were abstracted into a single category, which gave a final interrater agreement score of 96.4%. The remaining discrepancies were discussed and resolved. The resulting categories are given in the lower part of Figure 5, together with the distribution of constructs to categories, both overall and by organizational group.

The categories that contain most constructs are (6) *Organizational challenges – Technical challenges* and (8) *Integration implementation and technology – Integration requirements, design, and interfaces*. In the former category, all three organizational groups are represented, but with more constructs from product managers and QA managers. The latter category is dominated by developers' constructs.

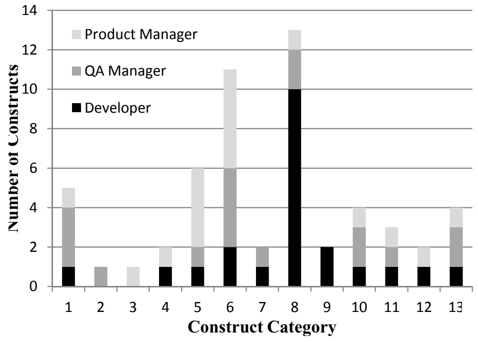
Developers. This group spread their constructs over the most categories. This group also placed the most constructs in one category (8) *Integration implementation and technology – Integration requirements, design and interface*, which may indicate a joint focus on the dilemmas of prioritizing the more technical parts of integration engineering in contrast to the more planning-related phases.

QA managers. The QA managers are represented in the fewest categories, although their constructs are evenly distributed in the categories in which they are represented. Most constructs from QA managers are placed in the cate-



Element Categories

1. *Unstable/new technology*
2. *Spurious errors*
3. *Integration with 3rd party products*
4. *Coordination of data*
5. *Coordination of functionality*
6. *Coordination of platform*
7. *Coordination in time*
8. *Difficulties with on-site installation of integration*
9. *Difficult/lack of will to reuse code*
10. *Lack of communication on technical issues*
11. *Lack of documentation*
12. *Difficult communication between stakeholders regarding integration requirements*
13. *Lack of ownership of integration tasks*
14. *Allocating/acquiring the right personnel/knowledge/skills*
15. *Interpersonal issues*
16. *Too product focused/Resistance to integration*
17. *Management lack of priority*
18. *Other*



Construct Categories

1. *Need resources to handle – Need (only) to agree what to do or to allocate resources*
2. *Ability to do integration – Will to do integration*
3. *Challenges associated to a specific integration project – General challenges*
4. *Challenges that demand involvement – Challenges that do not demand action*
5. *Challenges within Company X – Challenges that involve 3rd parties, including customers*
6. *Organizational challenges – Technical challenges*
7. *Challenges due to multiple integrations – Challenges with integrating with one system*
8. *Integration implementation and technology – Integration requirements, design, and interface*
9. *Challenges associated to development – Challenges associated to quality assurance*
10. *Challenges associated to development process – Challenges regarding to human resources*
11. *Critical challenges – Non-critical challenges*
12. *Challenges that are possible to solve – Challenges that are outside handling*
13. *Other*

Figure 5. Element and construct categories and distribution of elements and constructs to categories

gories (1) *Need resources to handle – Need (only) to agree what to do or to allocate resources* and (6) *Organizational challenges – Technical challenges*, which may indicate a main concern among QA managers toward trade offs between (human) resources and organization on the one hand, and tasks to be done and technology on the other hand.

Product managers. Product managers put most of their constructs in (5) *Challenges within Company X – Challenges that involve 3rd parties, including customers* and (6) *Organizational challenges – Technical challenges*. This may indicate a view among product managers that integration challenges primarily concern balancing internal procedures with external customer-driven demands, as well as with balancing organizational and technical issues.

4.2.3 Correspondence with Supplied Constructs

Following Honey's method [19, 23], we computed the similarity scores of each construct against each of the three supplied constructs: *Cause – Consequence*, *Critical – Not critical*, *Easy to handle – Hard to handle*. These three supplied constructs are intended to map the elicited constructs to practice. If an elicited construct, say *Development – Testing* receives a high similarity score with, say *Critical – Not critical*, this means (1) that the elements that were ranked on the elicited construct were ranked similarly on the *Critical – Not critical* construct, and (2) that the respective poles of the two constructs describe each other; i.e., that development is critical, while testing can wait.

Aggregated results can be obtained by comparing construct categories to the supplied constructs. Thus, similarity scores for individual constructs within a category were aggregated into an average score. See Figure 6 for the average similarity scores for each of the 13 construct categories with respect to each supplied construct.

We summarize the correspondence with the supplied constructs below. Note that categories that contain few constructs or only constructs from one interviewee (e.g., categories 4, 9 and 12) are not commented upon since they provide a weak basis for generalization. Of the remaining categories, only those where the majority of the constructs had high similarity scores are commented in depth.

Critical – Not critical. In category (5) *Challenges within Company X – Challenges that involve 3rd parties, including customers* five of the six constructs allocated to this category have a high similarity score with the supplied construct *Critical – Not critical*. Half of the constructs indicate that the internal challenges are the most critical, while the other half indicates that the external challenges are the most

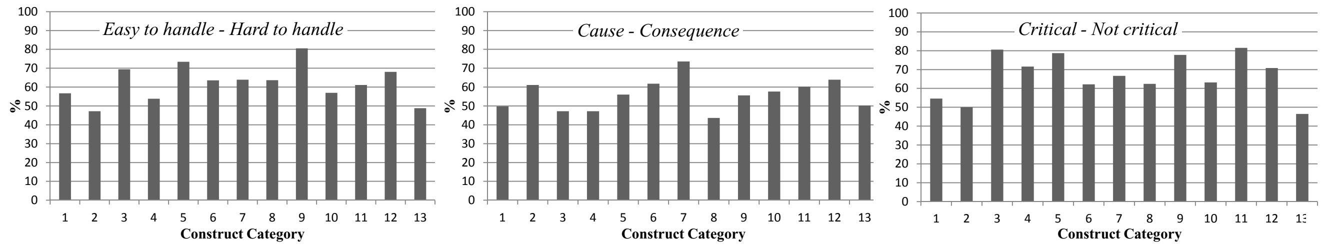


Figure 6. Average percentage similarity with supplied construct

critical. In the latter case, all the constructs are from product managers, while in the former case there is one construct from each organizational group.

In the category (6) *Organizational challenges – Technical challenges* most constructs indicate that the organizational challenges are the most critical. The constructs have mixed similarity scores, with an emphasis on intermediate scores. All organizational groups are represented evenly, which indicates that there is an agreement that the organizational challenges are more important than the technical.

The product managers also have the highest similarity score to the supplied construct *Critical – Not critical* within category (6), which might indicate that the constructs placed in this category have the same meaning as *Critical – Not critical* to the product managers, i.e. the elements have a similar rating on *Critical – Not critical* as on the constructs from the category.

Within the category *Need resources to handle – Need (only) to agree what to do or to allocate resources*, the constructs from QA managers are the ones with the highest similarity score to the supplied construct *Critical – Not critical*. This would indicate that the constructs in this category might be perceived by the QA managers as having the same meaning as *Critical – Not critical*.

Easy to handle – Hard to handle. In (5) *Challenges within Company X – Challenges that involve 3rd parties, including customers* half of the constructs indicate that the internal challenges are easier to handle than the external challenges. These constructs are put forth by product managers. In contrast, the other half of the constructs in this category indicate that the external challenges are easier to handle than the internal challenges. These constructs constitute one from each group. This indicates that there is not a consensus on whether the external or internal challenges are easiest to handle, but that most of the product managers see the internal challenges as easier to handle.

Similarly to the previous category, there seems to be differing opinions about the organizational and technical challenges. Half of the constructs in the category (6) *Organizational challenges – Technical challenges* indicate that the organizational challenges are easier to handle than the technical challenges. These constructs are put forth by product managers and QA managers. The other half indicates the opposite, i.e. that the technical challenges are easier to handle than the organizational challenges. In this latter half, all three groups are represented. However, the constructs from QA managers and product managers are put forth by only one person from the each group. In the former half, each group is represented by at least two persons, and therefore there is a stronger correlation. It is therefore likely to assume that the product managers and QA managers perceive the organizational challenges as easier to handle than technical challenges. There is a correlation indicating that the critical challenges are easier to handle than the non-critical challenges. All groups agree on this correlation and they all have high similarity scores.

Cause – Consequence. There is no strong correlation between the construct *Cause – Consequence* and *Easy to handle – Hard to handle*. Nevertheless, most constructs with intermediate and high similarity scores indicate that the causes are easier to handle than the consequences. There is consensus between all three groups that the critical challenges are the causes. In the category *Organizational challenges –Technical challenges* there seems to be an agreement between the QA managers and the product managers that the technical challenges are causes. Two developers agree on the opposite, that the organizational challenges are causes. (The former correlation is stronger than the latter.)

5. Discussion and Conclusion

The research objective laid down in Section 3.1 was twofold: to address Company X's specific issues in a systematic manner and to address and disseminate the findings to a wider community.

For Company X, this study was the initial step to understanding and overcoming software integration challenges. The study elicited concrete formulations of pain points seen by key practitioners involved in the company's integration projects. The interviewees were chosen from several integration projects, giving a diversity of situations in which integration challenges were experienced.

Viewpoints from three organizational groups were elicited. Process improvement is traditionally initiated by managers, but as mentioned introductory, it is important to involve several organizational groups. Both differences and similarities in perceived challenges and their perceived criticality and tractability were uncovered between groups.

A mutual awareness and understanding of differences in perceived challenges is crucial for efficient problem solving. For example, the developers mostly mentioned technological challenges and challenges related to information and knowledge. The QA managers on the other hand, were more concerned about human resources and responsibilities, while the product managers perceived challenges related to organization and communication as most important. Even though differences such as these might not be particularly surprising, they should still be explicated and subject to analysis when improving development processes.

On the other hand, when there is consensus across all groups on certain issues the company has a valuable incentive to initiate process improvement. For instance, challenges concerning the allocation of the appropriate personnel and knowledge were mentioned by all three groups.

Further indications on how to prioritize process improvement is given by the challenges' rankings on the supplied constructs (*Cause — Consequence*, *Critical – Not critical*, *Easy to handle – Hard to handle*). Challenges that are perceived as *critical causes* and that are *easy to handle* may be a good place to start. Three main areas can thus be extracted as possible starting points for an improvement process:

A: *Responsibility*. This concerns issues of allocating ownership and responsibility for the integration. Currently, the responsibility is placed with the system initiating the integration. However, according to the practitioners, this is not perceived as the most appropriate way of organizing the integration work.

B: *Requirements*. This was pointed out by several of the interviewees as a challenging activity for integration. In this case, challenges related to requirements concern both what functionality to include and the translation of the market needs into a technical specification.

C: *Knowledge*. Challenges related to knowledge are both about access to documentation as well as people's knowledge about the systems.

The challenges that were uncovered by the grid analysis confirm several of the representatives' initial perceptions (Section 3.1). In particular, the area *Responsibility* relates to perceptions 3 (Vague responsibilities...) and 8 (Little emphasis...on project management...), *Requirements* relates to 4 (...things are not specified...), and *Knowledge* relates to 7 (Too much time...discussing interfaces...) in Section 3.1.

Further, perception 1 (A disproportionate amount of time is spent on integration...) and 5 (There is little time for specification) are variants of the interviewees' perceptions that they always have too little time on integration projects. Perception 2 (The causes...are due to managerial, rather than technical, reasons) is interesting: There was an overall consensus that managerial (organizational) challenges were critical. However, developers saw organizational challenges mostly as causes. QA and product managers viewed mostly technical challenges as causes (and organizational challenges as consequences). Perception 6 (Standardization is lacking) was not stated explicitly by the interviewees, though some of the developers stated that the low amount of recycling is due to the interfaces not being standardized.

We also investigated terminological differences. Such differences might cause difficulties when communicating about integrations. Minor terminological differences were identified both between and within groups, and all differences were of the type "different terminology for the same concept". Different terms for the concepts requirements, semantics and the concept of protecting your own system were identified. Though the differences are perhaps not serious, they are now revealed. By minimizing these differences and introducing consolidated terms, the integrations might be perceived as less challenging.

The overall results of the study were presented at an internal seminar at Company X. The authors will initiate further discussions with the interviewees, with other representatives of the three groups, and with senior management. The objective will be to lay down a course of action for the ongoing integration projects, for future projects, and for further elaborations of this study. This study provides a starting point for other companies that are involved in software integration and that see a need to systematize the company's knowledge on this topic.

The second part of the research objective concerns impact beyond Company X. The main purpose of the study was not to generalize the results to a wider range of process improvement situations, or even to integration projects in other companies, but to use scientific methods to shed light on the situation in a specific company, see also [20]. What may be used in other situations is the research method. This study provides a starting point for other companies that are involved in software integration and that see a need to systematize the company's knowledge on this topic. In time, the salient concepts (elements and constructs) that are elicited and thereafter systematized by grid and content analyses, may form taxonomies or ontologies [14, 15]. These may be used to define empirically-based concepts of software integration challenges and their solutions. An ontology is a system of agreed-upon concepts (a Type 1 theory [13]), and is, in our context, the first step toward a practitioner-based scientific theory.

Acknowledgments

The authors are grateful to the interviewees for their insights and cooperation and to Hans Christian Benestad, Bent Foyn, Stein Grimstad, Dag Lorås, and Rune Steinberg for valuable discussions.

References

[1] C. Argyris. *Knowledge for Action*. Jossey-Bass Publishers, 1993.

[2] C. Argyris and D.A. Schön. *Organizational Learning II. Theory, Method, and Practice*. Addison-Wesley Publishing Company, Inc., 1996.

[3] N. Baddoo and T. Hall. Practitioners roles in software process improvement: An analysis using grid technique. *Software Process Improvement and Practice*, 7:17–31, 2002.

[4] V.R. Basili, G. Caldiera, and D. Rombach. The goal question metric paradigm. In J.J. Marciniak, editor, *Encyclopedia of Software Engineering*, pages 528–532. Wiley & Sons, Ltd., 1994.

[5] M.T.H. Chi, N. de Leeuw, M.H. Chiu, and C. LaVancher. Eliciting self-explanations improves understanding. *Cognitive Science*, 18:439–477, 1994.

[6] W.E. Deming. *Out of the Crisis*. MIT Press, 2000.

[7] T. Dybå, B.A. Kitchenham, and M. Jørgensen. Evidence-based software engineering for practitioners. *IEEE Software*, 22:58–65, 2005.

[8] K.A. Ericsson and H.A. Simon. *Protocol Analysis*. The MIT Press, revised edition, 1993.

[9] D.N. Ford and J.D. Sterman. Expert knowledge elicitation to improve formal and mental models. *System Dynamics Review*, 14(4):509–340, 1998.

[10] F. Fransella, R. Bell, and D. Bannister. *A Manual for Repertory Grid Technique*. John Wiley & Sons, Ltd., 2004.

[11] D. Gentner and A.L. Stevens, editors. *Mental Models*. Lawrence Erlbaum Associates, Inc., 1983.

[12] G. Gigerenzer. *Gut Feelings. The Intelligence of the Unconscious*. Viking, Penguin, Ltd., 2007.

[13] S. Gregor. The nature of theory in information systems. *MIS Quarterly*, 30(3):611–642, Sept. 2006.

[14] T.R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, 1993.

[15] M. Gruninger and J. Lee. Ontology – applications and design. *Communications of the ACM*, 45(2):39–41, 2002.

[16] T. Gulledge. What is integration? *Industrial Management & Data Systems*, 106(1):5–20, 2006.

[17] G. Hohpe and B. Woolf. *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. Addison-Wesley, 2004.

[18] O.R. Holsti. Content analysis. In G. Lindzey and E. Aronson, editors, *The Handbook of Social Psychology*, volume 2. Addison-Wesley, 1968.

[19] P. Honey. The repertory grid in action. *Industrial and Commercial Training*, 11:452–459, 1979.

[20] F. Houdek. External experiments—a workable paradigm for collaboration between industry and academia. In N. Juristo and A.M. Moreno, editors, *Lecture Notes on Empirical Software Engineering*, volume 12, chapter 4, pages 133–166. World Scientific, 2003.

[21] T. Hurson. *Think Better: An Innovator's Guide to Productive Thinking*. McGraw-Hill, 2007.

[22] IEEE Standard Computing Dictionary—a compilation of IEEE standard computer glossaries, 610–1991, 1991.

[23] D. Jankowicz. *The Easy Guide to Repertory Grids*. John Wiley & Sons, Ltd., 2004.

[24] P. Jarvis. *The Practitioner-Researcher*. Jossey-Bass Publishers, 1999.

[25] P.N. Johnson-Laird. *Mental Models: Towards a Cognitive Science of Language, Inference, and Consciousness*. Cambridge University Press, 1983.

[26] P.N. Johnson-Laird. Mental models and thought. In K. Holyoak and R.J. Sternberg, editors, *The Cambridge Handbook of Thinking and Reasoning*, pages 185–208. Cambridge University Press, 2005.

[27] A.R. Kearney and S. Kaplan. Toward a methodology for the measurement of knowledge structures of ordinary people: The Conceptual Content Cognitive Map (3CM). *Environment and Behavior*, 29(5):579–617, Sept. 1997.

[28] G.A. Kelly. *The Psychology of Personal Constructs*. Norton, 1955.

[29] G.A. Kelly. *A Theory of Personality*. Norton, 1963.

[30] B. Kitchenham, T. Dybå, and M. Jørgensen. Evidence-based software engineering. In *Proc. Int'l Conf. Software Engineering (ICSE'04)*. IEEE Computer Society, 2004.

[31] K. Krippendorff. *Content Analysis: An Introduction to its Methodology*. Sage, second edition, 2004.

[32] M. Markus and C. Tanis. The enterprise system experience—from adoption to success. In R. Zmud, editor, *Framing the Domains of IT Management: Projecting the Future through the Past*, pages 173–207. Pinnaflex Educational Resources, Inc., 2000.

[33] L. Mathiassen. Reflective systems development. *Scandinavian J. Information Systems*, 10(1&2):67–118, 1998.

[34] G.M. McGrath and E. More. Data integration along the healthcare supply chain: The pharmaceutical extranet gateway project. In *Proc. 34th Hawaii Int'l Conf. on System Sciences*. IEEE Computer Society, 2001.

[35] J. McKeen and H. Smith. New development in practice II: Enterprise application integration. *Communications of the Association for Information Systems*, 8:451–466, 2001.

[36] G.A. Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 63:81–97, 1956.

[37] N. Niu and S. Easterbrook. So, you think you know others' goals? a repertory grid study. *IEEE Software*, pages 53–61, March/April 2007.

[38] D.A. Norman. Some observations on mental models. In D. Gentner and A.L. Stevens, editors, *Mental Models*, chapter 1, pages 7–14. Lawrence Erlbaum Associates, Inc., 1983.

[39] W.A. Shewhart. *Economic Control of Quality of Manufactured Product. 50th Anniversary Commemorative Issue*. American Society for Quality, 1980.

[40] W.A. Shewhart and W.E. Edwards. *Statistical Method from the Viewpoint of Quality Control*. Dover, 1986.

[41] V. Stewart and A. Stewart. *Business Applications of Repertory Grid*. McGraw-Hill, 1981.

[42] M. Themistocleous, Z. Irani, R. O'Keefe, and R. Paul. ERP problems and application integration issues: An empirical survey. In *Proc. 34th Hawaii Int'l Conf. on System Sciences*. IEEE Computer Society, 2001.