Benchmarking the Customer Configuration Updating Process of the International Product Software Industry

Slinger Jansen[1], Wouter Buts[1], Sjaak Brinkkemper[1], and André van der Hoek[2]

[1] Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands, {s.jansen, wftbuts, s.brinkkemper}@cs.uu.nl
[2] Department of Informatics, University of California, Irvine, CA, USA, andre@ics.uci.edu

Abstract. Product software vendors have trouble defining their customer configuration updating process; release, delivery and deployment are generally underrated and thus less attention is paid to them. This paper presents an international benchmark survey providing statistical evidence that product software vendors who invest in the customer configuration updating process perceive that they are more successful than those who do not. Furthermore, the benchmark survey provides a vivid picture of the customer configuration updating practices of product software vendors and that customer configuration updating is an underdeveloped process.

1 Introduction

Software engineering research has always focused on improving methods and processes that have an end-goal of producing a working system. This focus generally does not include the processes around release, delivery, deployment, and run-time monitoring of a system. Only few theoretical studies regarding the CCU process have been conducted, such as Carzaniga's software deployment characterization framework [1] and Hall, Heimbigner, and Wolf's development of tools for software deployment [2] and Jansen et al.'s evaluation of deployment tools [3]. Empirically, CCU has received little scientific attention. The survey presented in this paper attempts to fill that niche.

Product software is defined as a packaged configuration of software components or a software-based service, with auxiliary materials, which is released for and traded in a specific market [4]. Customer configuration updating (CCU) is defined as the combination of the vendor side release process, the product or update delivery process, the customer side deployment process and the usage process [5]. The CCU model, created by Jansen and Brinkkemper, was used to create a survey among several Dutch product software vendors, to establish how mature Dutch product software vendors are in regards to CCU [6]. In this paper the survey is extended and repeated in an international setting. The survey was conducted from the University of California, Irvine. Seventy-eight companies,

from twenty-six countries, participated in the survey. The respondents are product managers. The participating companies received a custom-made benchmark report in which their CCU process is compared to the total respondent base and competitors in the same industry.

The contribution of this research is threefold. First, the state of the practice of the CCU process of international product software vendors is investigated and reported. Second, a benchmark is created for international product software vendors, which gives them a clear insight in their CCU process. Third, three hypotheses are investigated. Section 2 presents the CCU model. Based on this model, CCU processes and practices are defined. In Section 3, the research design is presented, consisting of the hypotheses and research approach. The research approach includes the creation of the survey that is designed to measure the processes and practices defined in Section 2. In Section 4, the raw survey results are presented, including the data for each CCU process. This data enables us to report on the state of the practice of CCU. Section 5 presents our in-depth analysis of the hypotheses and section 6 touches upon validity constraints. Finally, in Section 7 the conclusions are presented and possibilities for future research are discussed.

2 Customer Configuration Updating

The survey presented in this paper, is based on Jansen and Brinkkemper's CCU model, displayed in Figure 1. The CCU model consists of a state diagram of the customer on the right side and the supporting vendor functions on the left. A specific concern for product software vendors is that every vendor has to release, deliver and deploy its product(s) on a wide range of systems, in many variations, and for a wide range of customers [7]. Each of the CCU processes represents several practices. The practices consist of one or more capabilities. A vendors CCU maturity is measured by one question per capability in the survey.

The processes and their practices used in the benchmark survey are defined using the SPICE model for process assessment [8] [6]. The SPICE model defines a process as "a statement of purpose and an essential set of practices that address that purpose". SPICE was used as a tool to help shape the survey and enabled the researchers to take the CCU model and derive practices and capabilities from the processes in a structured matter. The processes are defined as follows:

Release - The release process is made up of four practices. The first practice is release frequency. The capabilities falling under the frequency practice are that a vendor must frequently release major, minor, and bug fix releases and that a vendor must synchronize these releases with customer convenience and demand. The second practice is explicit release planning, which constitutes how releases are planned within the organization. The third practice concerns release scenarios and touches upon aspects such as scenario management and particular policies involved. The final practice is management of (external) components and dependencies. All dependencies between components, be they products that have been built by the vendor or purchased externally, must be managed by creating explicit dependencies between these products and components.

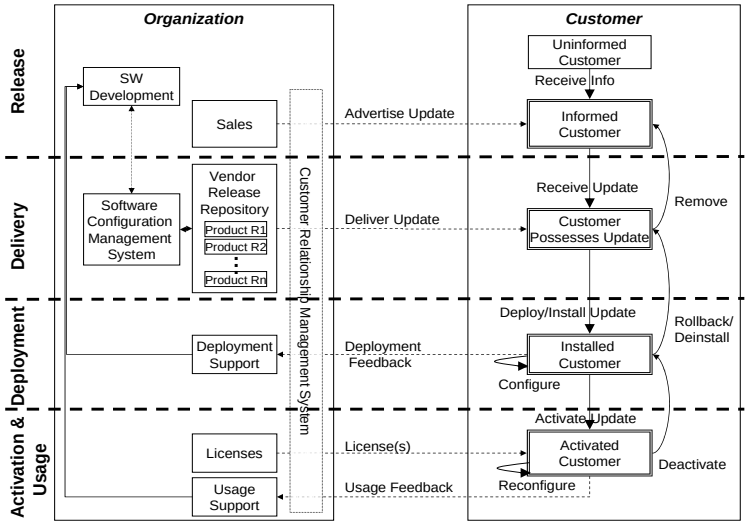

Fig. 1. CCU Model

Delivery - With regards to the delivery process there are two practices to be defined. The first practice prescribes that vendors must use every possible channel for the distribution of products and updates [3]. A vendor conforms to this practice when the product is (for instance) not only available through the commercial website but through physical media as well, such as for instance USB sticks. The more channels the vendor uses to eliminate manual intervention for product and update delivery, the higher its scores are. The second practice states that every possible method for delivery must be applied for knowledge delivery. The knowledge, such as product updates, product news, and customer feedback, should, for instance, be delivered through (semi-) automatic push and pull delivery.

Deployment - There are four practices defined for the deployment process. To begin with, a product must be removable without leaving any remnants of data on a system. Secondly, dependency management should be implemented, consisting of making dependencies explicit, and checking the customer's configuration before product deployment, to foresee and prevent errors. The third deployment practice is that updates and installations must be able to deal with customizations made by customers or third parties. A vendor supports the customization practice when a software architecture is in place that enables customizations. The fourth practice is deployment reliability, which is ensured by capabilities such as validity checks, feedback reports, and externalization of customer specific changes and data [9].

Usage - A vendors usage process is based on three practices. The first is license usage. A vendor must (semi) automatically handle all license requests and distribute licenses. A vendor supports the license practice once customers can

explicitly manage their licenses and once licenses are generated automatically when a sales contract is signed. Secondly, vendors must make use of feedback to gain as much knowledge about the product in the field as possible (use usage- and error reports). The third practice is that a vendor must be aware of its payment methods. A vendor sufficiently implements the payment practice when it incorporates payment calculation through usage, user (name), time unit, floating user, lump sum etc.

3 Research Design

In the area of CCU some explorative empirical work has been undertaken in the form of case studies [5] and design research [10] [11] [2]. As the research area matures, surveys can be used to generalize theories and findings from the explorative work. During the execution of the Dutch CCU survey comparable hypotheses were addressed, as presented below [6]. Based on the hypothesis in the Dutch survey, researchers found relations between change in the CCU process and success. Furthermore, vendor views on the CCU process were recorded through the use of open questions. In this research we repeat some of these questions in order to find out if they hold on an international scale. We address the following hypotheses:

H1: Recent changes to the CCU process make the software product more successful. When a software vendor changes its CCU process, it is likely to improve customer experience and reduce the cost of updating per customer. Research shows that Dutch companies *experience* increased product success after adjusting the CCU process [6].

H2: The priority that the vendor gives to the CCU process has significant impact on the success of the software product. It is hypothesized that software vendors who highly prioritize the CCU process dedicate more resources to the CCU process, which should result in a more successful product.

H3: In their estimations for the future with regard to the state of the practice of CCU, product software vendors are looking for update tools that automate the CCU process and are customizable as well. Based on Hall et al. [2], it is hypothesized that the CCU process will evolve towards a more automated and customizable CCU process. At this point update tools are available, but most of them are not generic enough to be adopted by software vendors. Building tools (internally) from scratch, on the other hand, is an expensive and time-consuming process. It is hypothesized that product managers are looking for a tool combining the best of both worlds.

Respondents - Respondents organizational roles and functions matched the following profile: the respondent is a product manager (or someone with a related function who has more than average knowledge of the CCU process) at a product software company and completes the survey with regard to one product in the vendor's portfolio. Furthermore, the product is a software product that is delivered to customers and runs at the customer site. The respondent's derived profile was explained in the invitation letter regarding the survey as well as at the front page of the survey website. Respondents were approached direct,

indirect and through domain specific discussion groups. The respondents that were targeted directly were approached through University of California industry connections. These contacts consisted of business partners, alumni, and external student advisors. From these leads, a list of people matching the profile was generated. Potential respondents were invited by email twice. The survey was sent out to over 200 e-mail adresses that were gathered from websites. Also, the survey was posted on a well-visited forum. No questions concerned the way in which the respondent heard about the survey.

Survey tool - The research question and hypotheses stated above are investigated by conducting an online benchmark survey and the analysis of the data resulting from this survey. The survey consists of fifteen sections with four to fourteen questions per section, adding up to a total of ninety-four questions. The survey contained open, yes/no and multiple-choice questions[3].

An online survey tool was chosen because a large number of questions had to be answered. We aimed for more than fifty respondents, which makes interviews and paper surveys inefficient tools to use. The survey was accessible through a website of the University of California, Irvine. After considering a final selection of tools, Net Questionnaires was selected. Net Questionnaires offers the same functions as a lot of other online survey tools but in addition offers support for multiple types of questions and extensive SPSS export functionality. The tool also allows respondents to pause the survey to continue at a later point in time. The ability to pause the survey is convenient for such a lengthy survey. The respondents are also able to browse back and forth through different survey sections to confirm and change earlier answers. The survey was pre-tested by two product software mangers during think aloud sessions, which led to minor changes in use of language and terms. The majority of the questions contained mouse over events that provided the respondents with explanations and examples to clarify question goals. The survey's underlying concepts were tested in five expert interviews. A thinking aloud session with two potential respondents going over the questionnaire resulted in small changes.

4 Results

The results can be downloaded[4]. The majority of the respondents' job functions are closely related to the CCU process (developer, development manager, chief executive or chief operating officer). Most of the participating companies are relatively small: the smaller participant companies are startups or small businesses (one to ten employees). The current market is constantly bombarded with new entrants and these companies are more eager to share their information in an effort to improve their organization. A comparable amount of respondents can be found in the range of fifty up to five hundred employees. The majority of the respondents produce products that are released in more than one mar-

[3] The survey can be downloaded (in pdf format) from the following website: http://www.ccubenchmark.com, verified on 15/10/'09

[4] http://www.softwareecosystems.org/SCR-2008-051.pdf, verified on 15/10/'09

ket. Furthermore, most commonly the respondents build products for business productivity or system management.

To date, C++ is the main development technology for the survey respondents. Next to C++, C, Java, Basic and C# are frequently used technologies. 66.7% of the respondents has their headquarters in the United States. The dataset contains companies from 26 different countries. 95% of these companies release their product in the US, compared to 73% in Europe and around 50% in both central and southern America as well as Asia. Moreover, 73% of the companies build their product in the US. Europe and Asia coming (respectively 28 and 14 percent) are also favorable building locations. The US leads with 74%, Europe comes in second place with 24% in regards to the region where intellectual property is registered. The product sizes range from 200 to 2000 KLOC. About half (53%) of the respondents indicate that their products architecture is based on the client-server principle. Almost the same can be said for the products following the stand-alone architecture (49%). Many companies introduce parts of their products as web-based components (33%). To complete the general data overview: 18% of the respondents indicate they use open source components, where 78% do not. This leaves 4% of the company products to be completely open source.

Release - Generally, major product releases are published on a yearly basis. In addition to these yearly major releases, minor releases are made available to the market every three to ten months. Bug fix releases show a different pattern. Some respondents publish these releases (almost) daily, others choose to adapt to a reoccurring monthly schedule. Nine respondents do not use any form of pilot testing before releasing the product or update. The rest use between one and one hundred fifty pilot testers, with a mean of fourteen. Slightly more than half of the product software vendors take customer convenience into account with regard to release time (57%). Less than half of the tested companies use a formalized release planning that contains specific dates for upcoming major, minor and bug fix releases (45%). Of the respondents that actually perform release planning, the largest part publishes its release planning in such a way that all internal and product stakeholders can access the planning at all times (83%). Also, with regard to responding vendors that indicate to use a release planning, more than half of them use a formal publication policy concerning the release planning document, which specifies policy decisions for a specific release (57%). Just more than half of the respondents use a formalized release scenario that describes what happens step by step on a release day (56%). 70% of the companies use tools to support the CCU process. Another aspect related to explicit management of tools is dependency management on third-party components. 69% of the software vendors show that they explicitly manage relations between the products they are selling and the external components that are needed to deliver an executable package. Finally only 38% uses components-of-the-shelf. Half of the vendors release their product when it is most convenient to the customer. More than two thirds of the respondents explicitly manage these relations.

Delivery - When one is dealing with the delivery process in a product software company, choices have to be made regarding the information flows to customers. As it turns out, the majority of companies informs their customers through a website with all relevant information (84%). Sending individual emails is a popular tool as well (81%). Interesting to note here is that only a small portion of the software vendors uses unconventional methods to approach their existing and new customers. Possibilities such as informing the customer through the product (17%), newsletters (9%), general announcement lists (10%) and automatic push of updates and information (8%) are used less frequently. Other approaches are the use of skilled account/sales managers, blogs and user forums. 69% of the software vendors inform their customer between once a week and once every three months. Most of the respondents initiate contact on a monthly basis. The majority of respondents use e-mail as a way for customers to contact them and convey their problems (81%). Almost half of that same group indicates to offer an online bug system for their customers to report problems (46%). 72 companies let their customers manually pull the releases from the vendor servers. 23 let users automatically pull releases from their servers. 18 respondents manually push the product to the customer, where only 10 do the push automatically. 30% of the responding companies use an update tool that updates the product on the customer side. Respondents estimate that, on average, 7.9% of their product installations fail at first attempt.

Deployment - For major releases, respondents indicate that 86% of their customers install the product between one day and three to six months after receiving news of an update. For minor releases, the same conclusion can be drawn. Bugfix releases are generally deployed sooner. 85% of bugfix releases are installed between one day and one month with the highest answer density towards one day. The use of USB sticks is attractive (20%) for software delivery. Another frequently used method is the file transfer protocol (FTP, 30%). In 56% of these software vendors the update tool is able to update the program at runtime. All except one of the companies using an update tool are sure that their tool is still able to deploy the product if the customer implements customizations, extensions and/or customer specific solutions. 56% manage the relations between proprietary and external products. Only 56% separates all data produced by the user (documents, preferences) form the product data (components). Only thirty percent of respondents use an update tool to deploy its product at the customer side, even though on average eight percent of the product installations fail.

Usage - The respondents prefer payment per user (name): 46% of the respondents use this licensing method. Other frequently used methods are pay per usage (35%), lump sum (19%), pay for services (12%) and open source/no payment (12%). Of the companies providing licenses, 17 out of 57 allow the customer to renew, extend or expand the license without action or intervention from the product software vendor side. Exactly half of the respondents offer their customers licenses that expire. In 65 percent of the software vendors the responding company is aware of the way that their customers customize products. 27 of the 78 participating companies send automatic error reports when

errors occur during product usage by the customer. From those 27, one does not actually analyze the data fed back through the error reporting system. 31 companies indicate that their products generate usage reports, 25 of them actually analyze these generated results. License management is generally automated.

Other results - Respondents indicate to have made small changes to their CCU process over the last two years (49%). 23% indicates not to have changed anything, 22% has made large changes and 6% has completely re-designed the CCU process. The reasons given most frequently for improving the CCU process, are to serve customers more cost effectively, serve more customers, reduce the number of installation problems and shorten the release cycle. Respondents indicate to have other goals as well, such as simplifying the CCU process, increasing stability of the product, better personal customer service and product quality improvement. The respondents were asked how their product's success had changed over the last two years. Finally, 34 respondents see the CCU process as a (very) high priority process, 35 are indifferent thinking CCU is neither a high or low priority process and 9 respondents think it is a low priority process. The following was derived from some of the open questions:

Deployment failure - According to submitters, future deployment failure will be reduced by introducing testing procedures during the development phase, user acceptance testing, and in depth pre-release code reviews. Submitters agree that more planning, a controlled release process, auto updates, a more robust setup process, and dependency checks should be used.

Custom made and commercially purchased tools - 50% of the respondents see proprietary tools as being adequate. A small group says they would rather have bought a solid commercial build tool to avoid a big part of current problems and have access to support. The problem that is mentioned most often is that of license management. A large group of submitters would like to be able to implement a CRM module accessible by clients.

Evolving CCU process - Most experts believe that better organized firms will make CCU a more central part of their infrastructure and procedures. The only way to make it a more central part is to automate a significant part of the CCU process, which will also involve support automation. Managers think that more commercial tools will become available to support customer release management. Integration between aspects of customer release management such as installation tools, product delivery and customer registration and bug tracking is needed. Others think that there is no magic bullet and no significant changes, since large corporations tend to more tightly control software deployment. These problems will remain issues for as long as tools/environments/systems change. When asking for the best possible solution for CCU the experts indicate that they are looking for a customizable, stable, fully automated updating product with sophisticated configuration options, robust logging and reporting. But, managers are not sure if there is a 'best' solution that is perfect for every single case.

			3.		4.		
			Please indicate how your product developed over the last two years				
			The product is much more succesfull than two years ago	The product is more succesfull than two years ago	The product is about as succesfull than two years ago	The product is less succesfull than two years ago	Total
Please indicate how your CCU process has evolved over the last two years	No Changes	Count	2	4	6	6	18
		Percentage	11,10%	22,20%	33,30%	33,30%	100,00%
	Small improvements (new CM system etc.)	Count	9	12	16	1	38
		Percentage	23,70%	31,60%	42,10%	2,60%	100,00%
	Large improvements (automatic license gen.)	Count	9	7	1	0	17
		Percentage	52,90%	41,20%	5,90%	0,00%	100,00%
	Complete process re-design	Count	5	0	0	0	5
		Percentage	100,00%	0,00%	0,00%	0,00%	100,00%
Total		Count	25	23	23	7	78
		Percentage	32,05%	29,49%	29,49%	8,97%	100,00%
			1.		2.		

Fig. 2. CCU process change / product success crosstab

5 Results Analysis

H1: Recent change in the CCU process has significant impact on the *percieved* success of the software product. In order to investigate H1 two specific questions were asked in the survey, being "Please indicate how your CCU process evolved over the last two years" and "Please indicate how your product developed over the last two years". To the first question the respondent could answer on a four point scale ranging from minor changes to a complete process redesign and to the second question the respondent's answer could range from much less successful to much more successful. A cross-tab analysis was conducted with a chi-square test between the variables of change and success (Pearson Chi-Square = 37.7, $p < 0.01$). Figure 2 shows the cross-tab analysis results. The horizontal axis shows the dependent *success* variable and the vertical axis shows the *change* variable. The category: "the product is much less successful" was omitted in the analysis because no respondents chose it. The figure shows increasing percentages towards larger changes in the CCU process combined with the respondents belief of a more successful product as the line in Figure 2 indicates. The software vendors are divided in four main groups in Figure 1. Groups two and three support the hypothesis. Group two contains the software vendors with small to no changes in the CCU process and are "less successful" as well "as as successful as two years ago". Group three contains the software vendors with large and complete redesign changes in the CCU process whose products success increased. The relationships are in line with the hypothesis. Groups one and four do not support the hypothesis. By creating a group division one is able to create a three dimensional table. In addition to these questions, an extra question was asked that explicitly linked the change and success variables. Group two and three were expected to show a higher percentage towards a strong

relation between success and changes because these groups implemented large changes that resulted in more success or implemented small or no changes, which resulted in no or less success. The opposite goes for group one and four. *H1 holds*, with a minor proviso. For this survey, product sales, revenues, and profit were not taken into account due to the diverse nature of respondents. As such, the relationship is a perceived relationship and can not be validated fully.

H2: The priority that the vendor gives to the CCU process has significant impact on the success of the software product. In this hypothesis the relation between the vendor's CCU process priority and its effect on product success is investigated. The same approach as in H1 is adopted. Two question variables, being product success and CCU process priority, were compared in a cross-tab analysis. The chi-square test was conducted (Pearson Chi-Square: 44.5, $p < 0.01$), which proves that there is a relation. Looking at these results it can be concluded that when product software vendors regard their CCU process to be of a (very) high priority, their product success is significantly higher. *H2 holds*, though be it with the same proviso as for *H1*.

H3: In their estimations for the future with regard to the state of the practice of CCU, product software vendors are looking for customizable update tools that automate the CCU process. Based on the results from the open questions we accept the hypothesis. The respondents see automation as the way to further reduce cost per customer. An interesting addition is the mention of web-based solutions by some of the respondents. Where possible, some of the vendors hope to soon release a web based version of their product, such that customers no longer have to bother with deploying a product themselves. *H3 holds*.

Table 1. Relation (success) development and changes in CCU

	Group 1	**Group 2**	**Group 3**	**Group 4**
Success strongly influenced by changes	26%	10%	81%	0%
Success partially influenced by changes	33%	76%	19%	0%
Success not influenced by changes	41%	14%	0%	100%

There are several differences between the Dutch and international respondents. For instance, the number of employees is 120 times bigger on average for international respondents, ten times more end users are served, and products consist of more KLOC. Surprising results are that the number of used programming languages is smaller for international vendors and that Dutch vendors work more efficient with a 1:4 (employee/customer) ratio, over a 1:1.5 ratio for international companies. The Dutch outperform their international colleagues in some areas, such as pre-installation configuration checks.

6 Threats to Validity

The **construct validity** of the research is safeguarded in different ways. The questions in the survey are based upon scientific constructs derived from liter-

ature. Respondents were asked to provide data from an existing product. To counteract guessing behavior, each question included the "I don't know" option. Survey heuristics developed by Fowler were applied in detail [12]. Possible problems with regard to understanding the questions were checked through expert interviews and think-aloud sessions.

In regards to **internal validity** The survey respondents were treated anonymously, with regard to each other and could not see input of other participants. Such a participant approach improves the truthfulness and reliability of answers. Furthermore, the respondents were promised to receive an extensive benchmark report including customized advice as incentive to participate in the research. An important threat to H1 is the possible bias of providing social desirable answers. In other words: when a respondent indicates to have made (major) changes to its CCU process over the last two years, it might be more tempted to say that its product was (a lot) more successful after this. There are vendors that indicate to have a product that is less successful even though changes in the CCU process might have been made. These results strengthen our belief that the possible bias is of low risk to the research outcomes.

The challenge of getting respondents to participate in the research was addressed by offering them a customized benchmark report. Commercial resources and contacts through industry and university networks were used to advertise the survey amongst the target audience to gain a large data set to improve **external validity**. After excluding incomplete- and non targeted respondent software vendors from the dataset, seventy-eight software vendors remained.

	International	Dutch
avg # employees	3124	26
avg # customers	1001-5000	51-100
avg # end users	5001 -10000	501-1000
avg # KLOC	501-1000	121-200
avg # programming languages used	1,59	2,57
avg # non english translations	3,68	2,85
avg # developers (FTE)	8,17	7,29
avg product age (years)	7,6	9,8
avg # pilot testers	13,54	5,08

Fig. 3. Dutch/International Comparison

7 Discussion and Conclusions

This paper presents the results of survey research amongst seventy-eight international products software vendors. The survey is based on the CCU model [3]. The survey allows us to compare research results to data from earlier research [6] and generalize conclusions. Results from specific hypotheses show that changes in the CCU process as well as prioritization of the CCU process have a significant positive effect on product success.

The benchmark survey results show that CCU is an underdeveloped area. More attention and better tooling is needed. Investments in the CCU process improve the *perceived* product success, customer service and update and deployment reliability. The realization that almost every CCU process is barely covered and the fact that the majority of the respondents suggest the use of a tool that is currently unavailable to counteract these problems show that CCU is still developing. The significant relation between CCU improvement and product success proves that CCU is a worthy area for further research.

References

1. A. Carzaniga, A. Fuggetta, R. Hall, A. van der Hoek, D. Heimbigner, and A. Wolf, "A characterization framework for software deployment technologies," 1998.
2. R. S. Hall, D. Heimbigner, and A. L. Wolf, "A cooperative approach to support software deployment using the software dock," in *Proceedings of the International Conference on Software Engineering*, 1999, pp. 174–183.
3. S. Jansen, S. Brinkkemper, and G. Ballintijn, "A process framework and typology for software product updaters," in *Ninth Eur. Conf. on Software Maintenance and Reengineering.* IEEE, Los Angeles, California, United States, 2005, pp. 265–274.
4. S. Brinkkemper and L. Xu, "Concepts for product software," *European Journal of Information Systems*, vol. 16, pp. 531–541.
5. S. Jansen, G. Ballintijn, S. Brinkkemper, and A. van Nieuwland, "Integrated development and maintenance for the release, delivery, deployment, and customization of product software: a case study in mass-market ERP software ," in *Journal of Software Maintenance and Evolution: Research and Practice*, vol. 18. John Wiley & Sons, Ltd., 2006, pp. 133–151.
6. S. Jansen, S. Brinkkemper, and R. Helms, "Benchmarking the Customer Configuration Updating Practices of Product Software Vendors," vol. 0, pp. 82–91, 2008.
7. H. van der Schuur, S. Jansen, and S. Brinkkemper, "Becoming Responsive to Service Usage and Performance Changes by Applying Service Feedback Metrics to Software Maintenance," in *Proceedings of the 4th Intl. ERCIM Workshop on Software Evolution and Evolvability*, 2008, pp. 53–62.
8. Joint Technical Subcommittee between ISO (International Organization for Standardization) and IEC (International Electrotechnical Commission), "Iso/iec 15504, a.k.a. spice (software process improvement and capability determination)."
9. E. Dolstra, "Integrating software construction and software deployment," in *11th International Workshop on Software Configuration Management (SCM-11)*, ser. LNCS, B. Westfechtel and A. van der Hoek, Eds., vol. 2649. Portland, Oregon, USA: Springer-Verlag, 2003, pp. 102–117.
10. E. Dolstra, E. Visser, and M. de Jonge, "Imposing a memory management discipline on software deployment," in *Proceedings of the International Conference on Software Engineering.* IEEE, 2004.
11. S. Jansen, "Pheme: An infrastructure to enable any type of communication between a software vendor and an end-user ," in *In Proceedings of the International Conference on Software Maintenance 2007, tool demonstration*, 2007.
12. J. F. J. Fowler, *Improving survey questions: Design and evaluation.* Thousand Oaks, CA: Sage, 1995.