An Approach to Creating Design Methods for the Implementation of Product Software: The Case of Web Information Systems

Lutzen Luinenburg[1], Slinger Jansen[2], Jurriaan Souer[1], Sjaak Brinkkemper[2], and Inge van de Weerd[2]

[1] GX, Wijchenseweg 111, 6538 SW Nijmegen, The Netherlands
{lutzen.luinenburg,jurriaan.souer}@gxwebmanager.com
[2] Department of Information and Computing Sciences, Utrecht University, The Netherlands
{s.jansen,s.brinkkemper,i.vandeweerd}@cs.uu.nl

Abstract. Development of a new software product is a complex process, in which requirements, release deadlines, and technical challenges fight for priority. These products are designed with the first release in mind which can sometimes cause software developers to have to make extensive design changes in subsequent releases. This paper presents an approach that creates design methods for product software design projects. The resulting design method is used to create a complete, maintainable, and durable product design. The approach has been applied in a practical case study of a series of web information system design projects. In this paper we describe the part of the case where key features were gathered and associated to method fragments, and how they were validated by an expert group.

1 Introduction

Today's software product vendors face several development challenges such as dealing with fast-changing business requirements, negotiating with a large number of stakeholders, and reducing the technical complexity of software products. While trying to find balance between these development challenges, software vendors also need to guarantee the success of their software product implementations in terms of efficiency and cost-effectiveness. When implementing a software product at a customer, a vendor has to once again deal with fast-changing requirements. In many implementation projects, a generic software product is adapted to and extended towards the needs of an organization. In addition, these requirements have a high impact on the architecture and design of an application. Once captured, it is hard to translate these business requirements into the pre-existing features of a software product. To make this translation transparent, flexible and less complex, design methods are needed for the implementation of software products. However, there are a myriad of software engineering domains and even more software products that are part of these domains. Additionally, it is a difficult task for software vendors to create a design method that meets the situational design context of a design project. When looking at the current body of knowledge in this field, no generic approaches or methods

I.-Y. Song et al. (Eds.): ER Workshops 2008, LNCS 5232, pp. 426–436, 2008.

exist to support product software design with the aim to create durable and maintainable product software designs during implementation projects.

In this paper we propose an approach for design projects, using situational method engineering. Situational method engineering has been applied successfully in several cases and is an approach for finding the right method for an organization [1]. "Method engineering is the engineering discipline to design, construct and adapt methods, techniques and tools for the development of information systems" [2]. A special type of method engineering is situational method engineering which allows for creating methods that are tuned towards the unique development project situation [1, 20, 21]. Assembly-based situational method engineering is an approach in which method components are extracted and stored in a method base driven by situational method requirements [14, 20]. A method fragment is a description of an IS (Information Systems) engineering method, or any coherent part thereof [10].

The approach for design projects provides a guide in the selection and assembly of suitable method fragments into a product software design method. The approach helps a method engineer to assemble a design method, deriving development project situations, storing relevant methods, and selecting the right method fragments based on product-related situational factors. We call this approach the *Association and Assembly approach (AaAa)*.

We continue this paper by presenting related work. In the third section we describe the AaA approach. Section 4 presents a WIS case and the derivation of key feature groups. Section 5 provides an overview of key feature groups established in the WIS case. The last section presents some comments and conclusions.

2 Creating the Association and Assembly Approach

Creating a design method that addresses every gathered functional requirement of a software product in a certain domain is complex. When looking at the domain of web engineering [6] many WIS design methods seem to be suitable for the design of data-intensive and process-intensive web applications. Examples of these methods are: OO-H [8], WebML [5], OOWS [13], Hera [22], and UWE [12]. However, these methods focus on web applications that are built from scratch without any pre-existing components and ignore domains in which pre-existing components are used to develop new applications. An example of a domain where pre-existing components are reused for new projects is the domain of Web Content Management Systems (WCMSs).

The Association and Assembly approach (AaAa) consists of the following steps:

1. Identify web engineering project situations
2. Identify feature groupings of a software product
3. Select candidate methods for the identified feature groupings
4. Model relevant method fragments in a method base
5. Associate feature groupings with candidate method concepts
6. Assemble situational web design method.

In order to address the fact that pre-existing components and functionality are available, we have adapted an existing assembly-based situational method engineering approach by adding two extra steps [20]. The first additional step (*step 2*) decomposes a software product into its main feature groups, in order to get more insight into its key features and product characteristics. The second additional step (*step 5*) creates an association table wherein selected method fragments and software product feature groups are formally compared.

In the first step, development project situations can be specified by means of categorizing unique development project characteristics. Basically, different development situations allow for the development of situational methods. For each development project situation, specific project needs can be defined, such as applying UML for modeling since all developers have knowledge of this modeling language.

Second, key feature groups of a software product domain are identified. These feature groups provide selection criteria for the assembly of a new method from a functional perspective.

Third, the domain of the software product is determined and in that domain relevant candidate methods are selected. This can be done by means of a literature study which provides an overview of the available methods. For scoping and relevancy purposes, it is important to narrow down the list of candidate methods into a comprehensive set of methods by means of useful selection criteria (e.g. acceptance in the community).

Fourth, method fragments of the selected candidate methods are extracted and stored in a method base. A method base structures the method fragments and thus captures knowledge about engineering methods [10]. For a uniform specification of all methods, a meta-modeling technique needs to be used which addresses the process as well the product perspective of a method [3].

Fifth, the selected candidate methods are compared by means of positioning the feature groups against the method fragments from a product perspective (e.g. deliverables in a meta-model) in an association table. This provides insight into which feature groups are addressed by the selected design methods. Next, it shows overlap between the methods in a way that a feature group is addressed by multiple method fragments. The association table also shows which feature groups are partly or not at all addressed by a method fragment.

In the final and sixth step, the results from the association table and the identified project needs are input for the assembly process in order to create a situational design method for the implementation of a software product. To ensure that all selected method fragments are assembled in a meaningful way, assembly guidelines should be taken into account. Brinkkemper et al. [3] have proposed logic rules for the assembly of a meaningful method addressing five quality requirements, such as completeness, consistency, efficiency, applicability and reliability. Having knowledge about feature groups allows a method engineer to analyze and select method fragments taking the feature groups and functionalities into account.

The final result of the AaA approach is a situational design method for the implementation of a software product taking the product's feature groups and development project situations into account. The resulting design method comprises of design activities and deliverables that allow for the translation of requirements into designed features that are ready for development in the follow-up phase. For each development

situation, a specific route map can be chosen that influences the type and number of design deliverables produced during a project (*step 1*). Using this approach, a company can decide to use a more light-weight design approach for regular projects which enables them to produce design deliverables in an efficient way.

3 Derivation of Key Feature Groups in Practice: WIS Case

The AaA approach has been validated by means of a case study. This case has been performed within GX, a Web Content Management software company. GX develops and implements its proprietary Web Content Management System called GX WebManager [19]. As part of the Web Engineering Method (WEM), a situational WCMS design method has been constructed. WEM is a web engineering approach for the implementation of WCMSs and is consists of six traditional development phases: acquisition, orientation, definition, design, realization and implementation [17]. The AaA approach is evaluated through an expert validation and validated by practical projects according to Yin [23].

Souer et al. [17] positioned the term CMS-based web application which is a web application for the management and control of content. This type of web application can be seen as a subtype of a web information system and a WCMS. In figure 1, the implementation of AaA approach in a WCMS situation is depicted. In the first step, two development project situations have been defined, namely standard and complex development projects. Standard development projects were defined as development projects that did not need any customization from a component perspective. During the second step, fourteen key feature groups were identified.

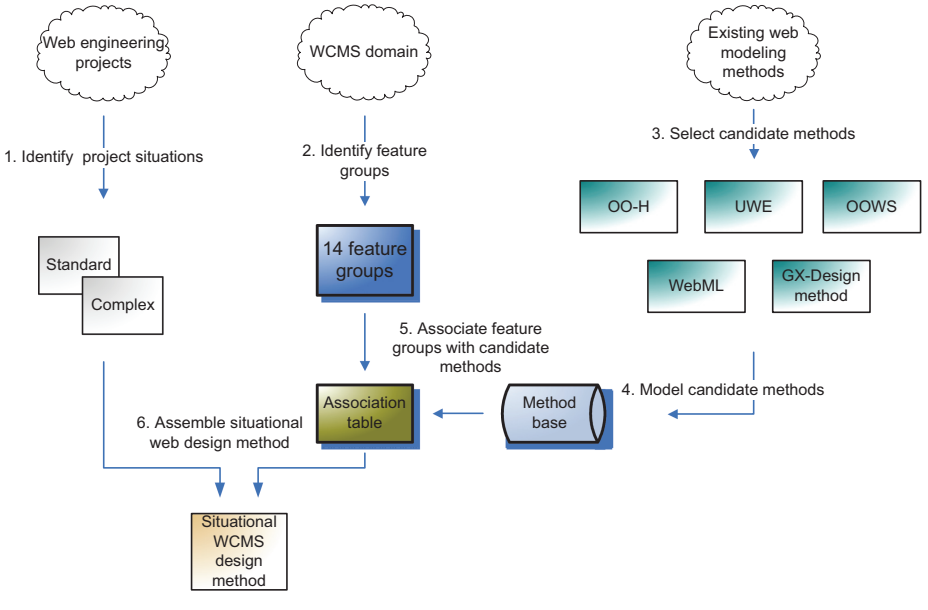

Fig. 1. The application of the Association and Assembly approach in a WIS case

These feature groups including the identification process are explained in the following sections. After the identification of feature groups, four mature WIS design methods and a propriety design method were selected. Three main selection criteria were applied: acceptance in the community, evolution of the approach and advanced tool support. In the fourth step, all relevant method fragments (e.g. requirements engineering, design and implementation fragments) were stored by means of using a meta-modeling technique called Process Deliverable Diagram (PDD) [20]. In total, 32 method fragments have been stored.

In step five, all candidate methods and key feature groups have been compared for selection purposes. During this association process it became clear that only three feature groups were well addressed by all methods: site management, personalization and e-forms. The domain and navigation models were suitable for these feature groups. Finally, a situational WCMS design method was assembled using the association table results and taking into account the development project needs. The situational WCMS design method consisted of four main activities: conceptual design, architecture design, presentation design and detail component design.

4 Key Feature Groups

Software product design addresses the requirements gathered during requirements engineering, therefore these product-related requirements have influence on the design of a software product. In this paper we call these design requirements feature groups. We define a feature group as "a class of functional design requirements." The purpose of such feature groups is to enable the association, selection and analysis of method fragments in order to assemble a situational method which is tailored mainly toward the design needs of a software product on a functional level. Please see the following section for more details on this process.

4.1 The Process of Establishing Key Feature Groups

No architectural reference models or frameworks are available in scientific literature concerning WCMSs that can be used for decomposing requirements into feature groups [17]. There is not much scientific literature present about WCMSs within the domain of web engineering. When analyzing the market of WCMSs and comparing it to the current state of literature, it can be argued that the business is ahead of the currently available scientific research. Therefore, the business of WCMSs is investigated in order to decompose a WCMS's functionalities into feature groups.

Despite the presence of many business selection tools, scientific well-founded reference models or standardized classifications of WCMSs are absent. Therefore, a marketing hype trend analysis has been performed in order to extract main feature groups from available market resources. Two of these WCMS feature groups are presented and described in the following section.

To gather key feature groups a literature review has been conducted wherein seven useful market resources have been identified. During this review step, a document analysis was also performed on the requirements and architectural documents of three complex development projects. The aim of this analysis was to find additional feature

groups and to match already found features with feature groups implemented during development projects at GX. From these resources, all categories and features are derived which led to a total of 113.

Next, all features have been categorized based on occurrence. When a feature was mentioned more than once, it was listed. After this, the remaining features were analyzed and categorized by means of adding labels to the features. After labeling all the remaining features, all features were again counted and organized. During this categorization process, some features were left out since they were not relevant (e.g. speed). This step resulted in a list with 26 preliminary feature groups and their accompanying sub features. The last refinement step of all the feature groups was done by means of two expert validation interviews within GX. Based on the discussion and the comments, the list of feature groups has been refined and narrowed down to a final list of 14 feature groups.

5 A WCMS Feature Group Overview

In this section an overview of all feature groups identified during the WIS case is given. First, we elaborate on two of these fourteen feature groups, namely: e-forms / transaction management and personalization. The reason for elaborating on the personalization feature group is that is has been an important subject within literature in relation to e-commerce. Nowadays, personalization plays still an important role in web applications. E-forms are a means of user interaction and transactional services. The importance of this feature group has risen, since business processes run more and more via the web.

E-forms / Transaction management - E-forms are the means that allows a user to interact with a web application as well as its services and data. Moreover, e-forms enable companies to provide transactional services to customers in an easy way (e.g. application for insurance, banking, etc.). The design of e-forms requires a systematic approach to control the structure, appearance, semantics, user assistance information and validation logic [9]. A special language to design e-forms are XForms, which is an embeddable XML language, which means that it is designed to be incorporated into other XML languages. XForms provide a consistent and declarative structure for dynamic e-forms [4].

Personalization - Srivastava et al. [18] state that personalization of the web experience for a user is the holy grail of many web applications, pointing to individualized marketing for e-commerce. According the authors the following data categories can be collected for personalization: content, structure, usage and user profile. In conceptual WIS design modeling, personalization is about tailoring content, presentation, or navigation based on user preferences or user behavior [7, 11]. Besides, personalization on the web is often used for campaign management, such that context-aware or customized information is published for a particular user target group.

The other twelve identified feature groups are: (1) Authoring; (2) Authorization management; (3) Community technologies; (4) Connectivity management; (5) Content repository; (6) Deployment and replication; (7) Digital Asset Management; (8)

Layout and presentation management; (9); Multi-channel delivery and syndication; (10) Site Management; (11) Web usage Mining; and (12) Workflow.

5.1 Using Feature Groups for Association and Selection

After the identification of the key feature groups of a software product, the feature groups can be used for the association and selection of relevant method fragments which support the design of a software product (see step 5). The list of feature groups is made situational by means of mapping the feature groups onto the existing functionalities of the software product. The reason for doing this is that a design method of a software product must address the implementation of these groups in the product itself. This makes the situational method specific for a particular software product. The feature group e-form was translated in several functionalities of the WCMS like validation, step, handler, router and field.

Referring to figure 2, the navigation models of OOWS and OO-H are associated with the specialized e-form and personalization feature groups. From the association table, the most suitable method fragments were selected with regard to coverage of the feature groups. These method fragments were also mapped onto the project needs defined in the first step of the AaA approach. The situational method has been assembled by means of creating relationships between all method fragments. For instance, concepts of the domain model were connected to navigation model concepts in order establish a mapping between those concepts. During the assembly of the situational method, we have applied assembly guidelines (e.g. consistency) in order to raise the quality of design [3].

For the association and selection of method fragments, an association table is created which positions feature groups and their product specializations against method fragments on three granularity layers: method, model and concept [3]. By comparing WCMS key feature groups with method fragments, a functional overview of all methods included from a WCMS perspective is obtained. Next to that, the association table gives insight into the coverage of all feature groups by several methods and therefore

		OOWS										OO-H						
		Navigational model										NAD						
		Navigational map	Context	Navigational link	AIU	Search mechanism	Navigational class	Navigational relationship	Process context	Activity Container	Main-AIU	navigational target	Navigational node	Service node	Collection node	Class node	Link	OCL expression
E-Form	Step		x		x		x		x	x	x					x		
	Handler								x					x				
	Validation													x				
	Router		x	x					x									
	Field						x											
Personalization	Personalization rule	x	x			x											x	x
	User profile																	
	User access	x												x		x		

Fig. 2. The association table (excerpt)

supports the selection of relevant method fragments. At the same time, the association table provides a path of evolution for the existing WIS design methods as well for situational design methods within a particular product domain.

5.2 Validation

We have applied two types of validation. First, the situational WCMS design method has been validated by means of an expert validation. Second, the complex route map of the conceptual design activity has been evaluated in two projects.

During the expert validation the completeness and correctness was evaluated in order to improve the design. The expert group consisted of two software architects and two consultants. Based comments provided by the experts, the design of the method has been improved.

The situational WCMS design method has been applied in a practical case study of a series of web information system design projects. The projects have been performed at a large Dutch Telecommunications Provider (Tel) and at a Dutch Governmental Organization (Gov). For the Tel project, sixteen use cases regarding a personal online space have been designed from retrospective, and for the Gov project, two complex use cases regarding an intranet web application have been designed conceptually.

Two consultants and three software developers were involved during the design phase. After creating the conceptual models, the project members filled out a survey. In the survey, questions were asked per conceptual design deliverable, addressing the following aspects: readability, abstraction level, correctness, tool supportability and applicability. Only for the Gov project questions were asked about the design process relating to aspects such as structure, involvement and communication. Subsequently, informal interviews were held in order to discuss the situational WCMS design method addressing topics such as customer communication, usefulness, superfluity and redundancy of information.

From these two projects, we observed that the situational WCMS design method was positively valued by all project members. However, software developers and consultants had different perceptions on the usefulness of several conceptual design models. In contrast to the software developers, consultants were less positive about the domain model, since they considered the models as too complex to understand for customers. Though, both groups stated that the domain model was useful for designing components that require customization. Next, it was commented that user modeling should not be a separate design step, but integrated with domain modeling. The two consultants had different opinions about the supportability of navigation models. The Gov consultant on the one hand considered the navigation model as supportive in order to get a better understanding in the information structure of a web application. On the other hand, the Tel consultant questioned the added value of the navigation model concerning communication and conceptual design in general. Furthermore, the Business Process Diagram (BPD) was perceived as the most readable and useful model in comparison to all other models.

In figure 3, a BPD is given that represents an illness notification process for employees of the Gov project. A user can fill in an illness notification and when confirmed, and based on this request the WCMS tries to get user information from a DB,

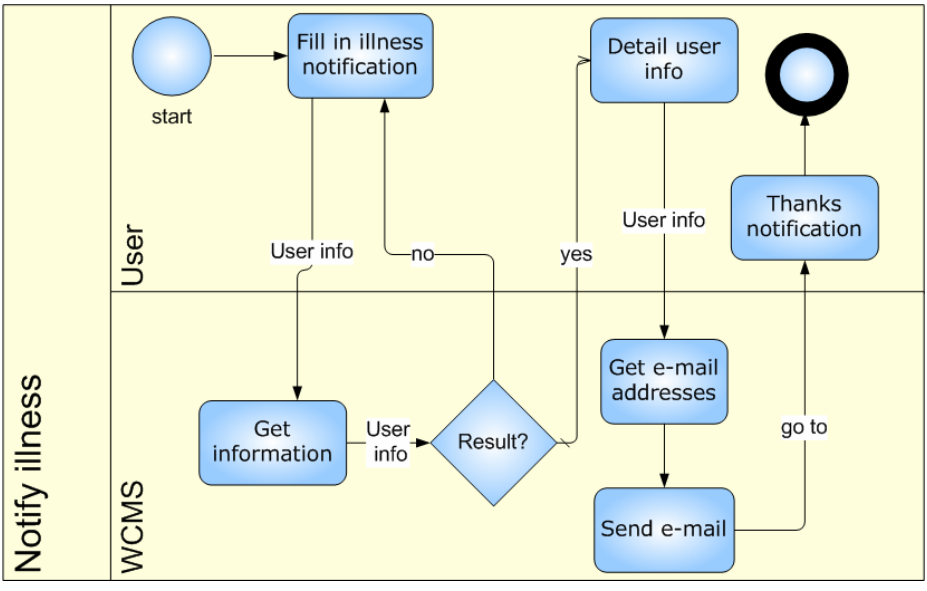

Fig. 3. A BPD that represents the 'notify illness process' of the Gov project

evaluates the results and based on that information, the user is able to provide additional user information. The process ends by retrieving relevant e-mail addresses for sending a notification to direct superiors of a particular employee.

6 Discussion

Before we can claim that the AaA approach is applicable to other domains, more case studies are required in different product domains. Currently, only one case has been used in order to validate the AaA approach.

Situationality was introduced into the generic design method for recurrent development projects in order to increase the method's applicability and productivity. We speculate that feature groups are related to product roadmaps, in that the product roadmap will include new feature groups, whereas feature groups might be able to assist in the creation of the product roadmap [15].

The AaA approach has been successfully applied to the assembly of a design method for recurring WIS design projects. The approach can provide a first design method to domains that are in need of more formal methods, tools or techniques.

7 Conclusions and Further Research

In this paper we presented an approach for creating a fitting software product design method for any type of software product implementation project. The approach provides a guide in the selection and assembly of suitable method fragments that create a product software design method. The approach helps a method engineer to assemble a

method, derive development project situations, store relevant methods, and select the best-fitting methods based on product-related situational factors. We call this a feature group. Next, we showed how to use these feature groups for the association and selection of relevant method fragments. The approach was validated by applying it to one WIS design case.

Future research includes the investigation of applying metrics to the association table in order to support the selection of the right method fragments [16]. Finally, it is interesting to investigate the applicability of the AaA approach for the development of product line design methods.

References

1. Aydin, M., Harmsen, F.: Making a Method Work for a Project Situation in the Context of CMM. In: Proceedings of the 4th International Conference on Product Focused Software Process Improvement, pp. 158–171 (2002)
2. Brinkkemper, S.: Method Engineering: Engineering of Information Systems Development Methods and Tools. Information and Software Technology 38(4), 275–280 (1996)
3. Brinkkemper, S., Saeki, M., Harmsen, F.: Meta-Modeling Based Assembly Techniques for Situational Method Engineering. Information Systems 24(3), 209–228 (1999)
4. Cardone, R., Soroker, D., Tiwari, A.: Using XForms to Simplify Web programming. In: Proceedings of the 14th international Conference on World Wide Web, pp. 215–224 (2005)
5. Ceri, S., Fraternali, P., Bongio, A., Brambilla, M., Comai, S., Matera, M.: Designing Data-Intensive Web Applications. Morgan Kaufman, San Francisco (2003)
6. Deshpande, Y., Murugesan, S., Ginige, A., Hansen, S., Schwabe, D., Gaedke, M., White, B.: Web Engineering. Journal of Web Engineering 1, 3–17 (2002)
7. Eirinaki, M., Vazirgiannis, M.: Web mining for Web Personalization. ACM Transactions on Internet Technologies 3(1), 1–27 (2003)
8. Gomez, J., Cachero, C.: OO-H Method: Extending UML to Model Web Interfaces. In: Information Modeling for Internet Applications, pp. 144–173. IGI Publishing, Hershey (2003)
9. Gouscos, D., Rouvas, S., Vassilakis, C., Georgiadis, P.: An Object-Oriented Approach for Designing Administrative E-forms and Transactional E-services. In: Bellahsene, Z., Patel, D., Rolland, C. (eds.) OOIS 2002. LNCS, vol. 2425, pp. 19–30. Springer, Heidelberg (2002)
10. Harmsen, F.: Situational Method Engineering. Ph.D thesis, University of Twente, Moret Ernst & Young Management Consultants, The Netherlands (1997)
11. Kappel, G., Proll, B., Retschitzegger, W., Schwinger, W.: Customisation for Ubiquitous Web Applications a Comparison of Approaches. International Journal on Web Engineering Technologies 1(1), 79–111 (2003)
12. Koch, N., Kraus, A.: The Expressive Power of UML-based Web Engineering. In: Schwabe, D., Pastor, O., Rossi, G., Olsina, L. (eds.) Second International Workshop on Web-oriented Software Technology, pp. 105–119 (2002)
13. Pastor, O., Fons, J., Pelechano, V., Abrahao, S.: Web Engineering: Theory and Practice of Metrics and Measurement for Web Development. In: Mendes, E., Mosley, N. (eds.) Conceptual Modeling of Web Applications: The OOWS Approach, pp. 277–301. Springer, New York (2005)

14. Ralyte, J., Deneckere, R., Rolland, C.: Towards a Generic Model for Situational Method Engineering. In: Eder, J., Missikoff, M. (eds.) CAiSE 2003. LNCS, vol. 2681, pp. 95–110. Springer, Heidelberg (2003)
15. Regnell, B., Brinkkemper, S.: Market-Driven Requirements Engineering for Software Products, Engineering and Managing Software Requirements. In: Aurum, A., Wohlin, C. (eds.) Engineering and Managing Software Requirements, pp. 287–308. Springer, Berlin (2005)
16. Rossi, M., Brinkkemper, S.: Complexity Metrics For Systems-Development Methods And Techniques. Information Systems 21(2), 209–227 (1996)
17. Souer, J., van de Weerd, I., Versendaal, J., Brinkkemper, S.: Situational Requirements Engineering for the Development of Content Management System-based Web Applications. International Journal on Web Engineering Technologies 3(4), 420–440 (2007)
18. Srivastava, J., Cooley, R., Deshpande, M., Tan, P.-N.: Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data. ACM SIGKDD Explorations Newsletter 1(2), 12–23 (2000)
19. Van Berkum, M., Brinkkemper, S., Meyer, A.: A Combined Runtime Environment and Web-based Development Environment for Web Application Engineering. In: Persson, A., Stirna, J. (eds.) CAiSE 2004. LNCS, vol. 3084, pp. 307–321. Springer, Heidelberg (2004)
20. Van de Weerd, I., Brinkkemper, S., Souer, J., Versendaal, J.: A Situational Implementation Method for Web-based Content Management System-applications: Method Engineering and Validation in Practice. Software Process: Improvement and Practice 11(5), 521–538 (2006)
21. Van Slooten, K., Brinkkemper, S.: A Method Engineering Approach to Information Systems Development. In: Proceedings of the International Federation on Information Processing Working Group 8.1 Working Conference on Information System Development Process, pp. 167–186. North-Holland Publishing Co., Amsterdam (1993)
22. Vdovjak, R., Frasincar, F., Houben, G.J., Barna, P.: Engineering Semantic Web Information Systems in Hera. Journal of Web Engineering 2(1-2), 3–26 (2003)
23. Yin, R.K.: Case Study Research: Design and Methods. SAGE Publications, Thousand Oaks (2003)