

A Specific Software Development Process for an Electronic Commerce Portal

Volker Gruhn
Dortmund University
Baroper Str. 301
D-44227 Dortmund
gruhn@cs.uni-dortmund.de

Lothar Schöpe
Informatik Centrum Dortmund e.V.
Joseph-v.-Fraunhofer-Str. 20
D-44227 Dortmund
schoepe@icd.de

Matthias Book
WebService Haltern
Heinrich-Rumpf-Str. 23
D-45721 Haltern
mbook@ws-haltern.de

Abstract

The development of electronic commerce or electronic business systems (EC/EB systems) is subject to different conditions than the development of conventional software systems. Thus, software development processes which were employed for conventional systems until now must be adapted to these new conditions. EC/EB systems typically have a high degree of interaction, which makes factors like ergonomics, didactics and psychology especially important in the development of user interfaces. They typically also have a high degree of integration with existing software systems like legacy or groupware systems. The development of EC/EB systems requires a new software development process which takes the "time-to-market" factor into account and allows the reduction of development time while maintaining the same level of quality. This paper introduces and describes an adapted software development process for EC/EB systems and its special features using the development of an electronic commerce portal system as an example.

1. Introduction

The development of EC/EB systems is a new challenge for the IT departments of companies doing business in various fields, as well as for software companies that actually develop these systems. Until now, the companies' IT departments develop, maintain and adapt conventional application software systems that are specific to their field of business. They use conventional methods, tools and programming languages within a software development process that employs a conventional distribution of tasks among roles – e.g. tasks are performed iteratively and incrementally in different phases by the system designer, the developer and the programmer or other roles. However, since the market conditions are changing constantly, companies have to adapt the existing application software systems accordingly or develop new

application software systems. EC/EB systems constitute one class of these new software systems. The IT departments have to face these new challenges with new methods, concepts and practices on the basis of an adapted software development process.

Electronic commerce is any kind of business transaction or market process where the involved parties (merchant and client) communicate electronically over a data network. Market processes comprise the market transactions pre-sales support, sales, distribution and after-sales support for products and services. Products and services that can be sold via electronic commerce – i.e., digital and semi-digital products and services [5], [2] – are finance services (insurances, stocks, fonds etc.), computer programs (software, shareware), cultural goods (admission tickets, magazines, books, videos, audio CDs) and travel services (train, plane and ferry tickets, hotels).

Depending on the involved parties, electronic commerce can be distinguished into "business-to-consumer (B2C) electronic commerce" or "business-to-business (B2B) electronic commerce" [12]. While the client is a private consumer in B2C, it can be a company or a government institution in B2B (the latter is also termed e-government). In both cases, we assume that the merchant is a company.

Electronic commerce systems support the involved parties in the electronic completion of their business transactions. Looking at the technical aspect, electronic commerce systems differ in many characteristics ranging from client/server architecture, network type, topology and distribution to communication protocols and services, as well as security concepts [13], [14].

Not all electronic commerce systems support all market processes electronically, but only various market transactions within one market processes. For example, catalog and shop systems are mainly suitable for pre-sales support and the actual sale. EDI systems or connected inventory systems focus on the sale in a narrower sense, rather than on pre- or after-sales support. Electronic commerce systems that solely handle pre- or after-sales support without actually offering products or services are

systems that only present information, e.g. news, stock market information, weather data etc.

Depending on the way the client interacts with these systems, push or pull systems can be distinguished. In these systems, pre- and after-sales support are performed by advertising in e-mails, discussion forums, newsletters, banners or affiliate programs. Payment systems (e.g. homebanking systems or the paybox) support only the aspect of payment processing during the sale.

In the same way that conventional application software systems are developed according to conventional software development processes, special software development processes are necessary to describe the development of electronic commerce systems [3], [8]. These software development processes for electronic commerce systems partially differ from software development processes for conventional application software systems regarding the types of tasks, the order in which tasks are performed, the roles that perform tasks, and the software tools used.

In software development projects where a conventional application software system is developed, the role sales usually just has the task to sell a service (i.e. the development of a software system) and to initiate the software development project that way. While conventional application software systems are usually not used as a marketing tool but just serve sales to acquire new clients, an electronic commerce system is in part a marketing tool. For this reason, marketing is heavily involved in the requirements analysis to define the project's goal.

While conventional application software systems gain user acceptance mainly through their functionality and can be positioned against the market's competition that way, a special class of electronic commerce systems (in this example, a shop system) can gain their user (i.e. client) acceptance only through their user interface. The user interface not only presents content in a certain layout, but also guides and supports the user.

The roles that are involved in the requirements analysis of the software development process are defined based on the way the electronic commerce system is acquired and used by the merchant ("make-buy-or-rent"). Distinguished roles are manufacturer, merchant, provider and client. A manufacturer develops and distributes (directly or indirectly) a software system that a provider can use to realize an electronic commerce system. A merchant uses a software system with its given functionality to realize an electronic commerce system according to the goal he defined. A merchant can also decide to build an individual electronic commerce system by developing the software on his own. A provider provides the software and hardware infrastructure for running the electronic commerce system. For example, a provider for shop systems might be an internet service provider (ISP); a provider for EDI systems might be a clearing center (CC).

A provider can be a merchant at the same time. A client uses the electronic commerce system to electronically perform market processes with a merchant (within the capabilities of the electronic commerce system).

Depending on the course of action within the software development process, the different roles are using different software tools, such as shop systems (Intershop, Openshop etc.), content management systems (Hyperwave, Gaus Inprise, Pirobase etc.) or software development/programming environments (JDK, Together J, etc.).

Because of the multitude of software systems that enable the realization of electronic commerce systems, the task of goal-oriented systems analysis, evaluation and selection during the requirements analysis has crucial significance. For example, while the merchant is responsible for maintaining the contents and the presentation of a shop system when he is simultaneously acting as provider, he only has to define the contents when he is renting a system from a provider. For rented electronic commerce systems, the provider supplies the hardware and network infrastructure and also the software tool to build the electronic commerce system. Since in this constellation, the provider strives to provide this infrastructure to several merchants in order to increase his own productivity and return on investment, he commonly uses only one software tool to realize different EC/EB systems for different merchants. In consequence, the effort for the realization of an EC/EB system can be reduced for both the merchant and the provider. On the downside, each merchant also has less opportunities for the individual presentation of his system's content.

The tasks concerning the selection of content and its presentation are associated with the systems design phase. These tasks are not included in a conventional software development process. The roles performing them are specialists for software ergonomics, didactics, graphic design and psychology.

If the integration of the EC/EB system into an existing infrastructure is necessary, methods, concepts and software tools for the integration must be available to use. Also, it must be decided if market processes for products or services should be electronically supported by an electronic commerce system. If support for products is required, the electronic commerce system has to be integrated with an open or closed inventory system of the merchant. An integration with conventional field-specific, highly individual application software systems is usually required when supporting market processes for services. These individual application software systems, termed legacy systems, are used by insurance companies, communal agencies, banks, power companies, etc. ("*...back-office integration enables two or more systems to communicate with each other, and one of these systems is a company's back-office.*" [9] In this paper, the term

"back-office" is equivalent to the term "legacy system."). In contrast, inventory systems are often offered through ERP systems of different software manufacturers (e.g. SAP, Oracle, Baan, Sage, PeopleSoft etc.). These ERP solutions provide interfaces (APIs) for the integration with other software systems. For example, this makes it possible to offer integrated solutions between Intershop and SAP, OpenShop and Sage or Oracle.

In some circumstances, the integration of several different inventory systems or individual application software systems may be necessary. This is usually the case for the implementation of electronic commerce malls or electronic commerce portals. However, an integration of systems may not be required when following the concept of "just-in-time" delivery. In this case, the integration is not performed on the level of the systems or software tools, but on the level of the user interface of the electronic commerce system.

The methods, concepts and software tools used, as well as the roles involved, depend on the way the integration is performed. For example, security aspects (use of firewalls, cryptography etc.) might have to be taken into account. These security aspects then not only have to be observed during the implementation, but also during the system draft and system design of the electronic commerce system. As another example, during pre- and after-sales support, dynamic client profiles can provide valuable information. To realize these profiles, several techniques are available (cookies, URL encoding, server-based profiles etc.) which have to be selected based on legal and technical implications.

2. The IPSI electronic commerce portal

An electronic commerce portal that is used by a special group of users in an intranet supports business-to-employee (B2E) transactions [10]. In this case, business transactions between management and employees, but also business transactions among employees are electronically supported. This requires the use of different software systems (e.g. a legacy system, shop system, web system, internet system, office system). Thus, an electronic commerce portal also acts as an integration platform for heterogeneous software systems [7].

The IPSI (Internet Portal System for Insurances) electronic commerce portal [4] is intended to support insurance agents in their daily work, using the electronic commerce portal in an intranet with the objective of optimal support of B2E transactions. For example, the insurance agents need access to information about their private and incorporated clients. Their work is also supported by a scheduler with reminder function, address book etc. Additionally, the insurance company uses the portal to supply its employees with up-to-date information on the product portfolio, tariffs and legislature. An

electronic commerce portal thus has to offer a multitude of functions. The advantage of the portal is the integration of different software systems that provide these functions, so the user can be guided directly to the needed information without having to perform awkward searches or sifting through unwanted information.

The user interface of the IPSI electronic commerce portal can be a WWW or WAP browser. The integrated software systems are termed subsystems. For subsystems that do not have their own data management functionality, a relational database management system is used.

The functionality of the subsystems office, content management, procurement and communication is provided by existing external systems which are integrated into the electronic commerce portal via adaptors (see Figure 1). Through combinations of the subsystems' functionality, new functions of the whole system are created. These are oriented towards the business processes (workflows) of the users: For example, to read an important contract date from the partner database legacy system, write an according note in the scheduler of the office system and initiate the transmission of a reminder message to the user's pager via the communications system, a series of different actions would be required using separated subsystems. Using the electronic commerce portal, however, the insurance agent can accomplish this task in a few steps.

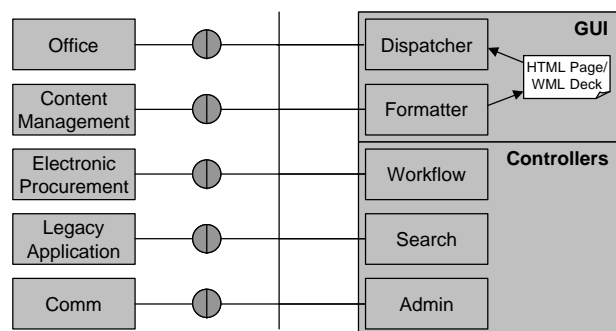


Figure 1. Architecture of the electronic commerce portal.

The business processes search and portal administration are handled by the search and the admin controllers, while all other business processes that involve subsystems are handled by the workflow controller.

Controllers and subsystems communicate by exchanging business objects, i.e. entities that have central significance for the business processes in the electronic commerce portal. "Business objects behave in a manner that is representative of real-world situations, and that makes sense to business experts" [1]. The business objects user, contact, appointment, task, message, shop item, order, order history, search request and search result are therefore known to all controllers and subsystems.

For example, to enter an appointment into the scheduler, the workflow controller creates an appointment object from the data received from the dispatcher and passes it to a method of the office subsystem that inserts the appointment into the calendar of the user. If the user wishes to be reminded of the appointment by e-mail, the workflow controller additionally creates a message object, connects it to a copy of the appointment object and passes it to the communications system, where it is inserted into the queue for e-mail messages and sent at the appropriate time.

3. Process description

The software development process for the development of a certain EC/EB system is defined by a process model. A process model presents all the activities (in a certain order), the required tools and the created intermediate or final products that are necessary to achieve the goal of the process. Thus, since a process model is tailored to a certain development project, it is more concrete than a generic practice model. A process model can however be based on a generic practice model. A process, on the other hand, is the execution of a process model (in the object-oriented way of thinking, a process is an instance of a process model), i.e. the tasks that are described in the process model are actually performed.

Although a formal description of a software development process in the form of a process model simplifies its support by automatic systems, it is not mandatory in order to achieve a positive effect in software development. In order to achieve consensus about the software development process among all involved people, a structured and comprehensive description can be sufficient. A company's knowledge about best practices was and is often described in internal documents and development guidelines. For example, ISO 9000 (part 1-3) defines only the contents of the description of best practices and development guidelines, but not their notation. However, description in a natural language bears the danger of misinterpretation because it usually has an enormous volume, and some concepts, dependencies and prerequisites can not always be precisely formulated.

The process model for the development of an actual EC/EB system – in this context, the IPSI electronic commerce portal – is presented schematically in Figure 2, using the Funsoft net notation [6]. In order to reduce the complexity of presentation and increase the level of abstraction, this notation allows to distinguish between elementary tasks (e.g. coarse draft, integration) and subprocess models which can again contain elementary tasks and subprocess models (e.g. requirements analysis, subsystem selection, prototype development).

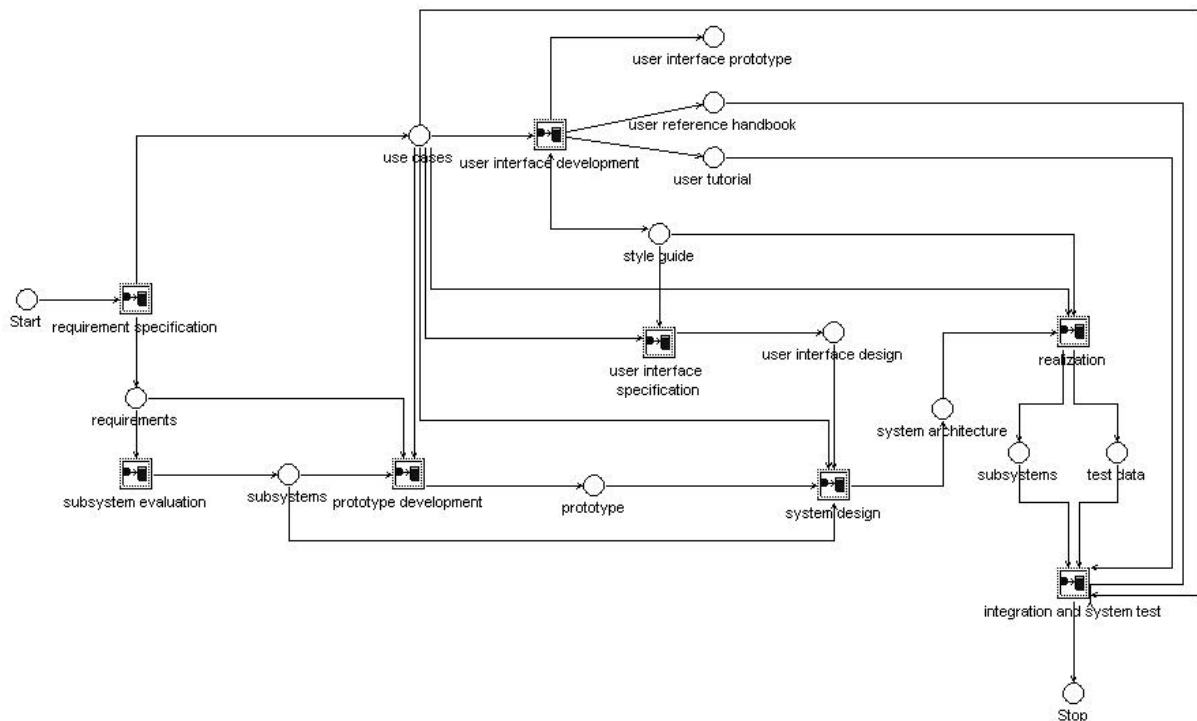


Figure 2. Electronic commerce portal development process model.

The object-oriented design using UML, the implementation of adaptors to integrate software systems as subsystems of the electronic commerce portal using the Java programming language, and the use of a middleware (CORBA/RMI) for communication within the portal are represented in this software development process. The development process also shows that the use cases described in UML are an important prerequisite for several subprocess models.

In the following sections, the complex development process of the IPSI electronic commerce portal is described in reduced form by completely describing the subprocess models, but not their internal details.

3.1. Requirements analysis

The requirements analysis comprises market analysis, subsequent proposal and contract evaluation, and project initialization. After this, the functional and non-functional requirements for the EC/EB software system are identified.

The first step of the market analysis is the creation of a positioning diagram. In this diagram, all potential competitors on the market are identified before extracting the direct competitors from the whole set and juxtaposing these with the company's own position on the market. The goal of the positioning diagram is to find new niches and to gain an overview of the duration of a competitive advantage on the market.

For the development of the electronic commerce portal for insurance agents, it had to be determined if other merchants on the market already offered a similar portal and which audiences those merchants targeted. Afterwards, the product idea was presented to several insurance companies, and one insurance company was won as a partner and potential client. During the proposal and contract evaluation, the feasibility of the client's requirements was clarified. The goal was a contract basis that was stable in every regard (content-wise, legal, mercantile), and a software system that met the client's functionality and quality requirements.

The identification and description of the portal's functionality and the priority-based structuring of these functions are very important tasks. On the one hand, the functionality must be sufficient to cover all requirements of the client, on the other hand, it must stand out from the functionality that competitors offer in order to gain an advantage on the market. It must also contain potential for further developments to ensure future market advantage.

An advantage over other products on the market can be achieved through high functionality. However, the realization of high functionality requires a certain effort, which is mirrored in the amount of time it takes to realize an EC/EB system. Thus, an advantage can also be achieved by making the EC/EB system available on the

market early. This means that according to the "time-to-market" concept, especially EC/EB software systems have to be developed and introduced to the market quickly. The identification of requirements and the assignment of priorities to those requirements with attention to their impact on development time thus is a highly significant task when developing EC/EB systems.

The initial list of requirements that results from the market analysis is the starting point for the creation of a requirements catalog for the whole electronic commerce portal that is to be developed. This requirements catalog is checked for contradictions, redundancy and completeness in several ways, for example by interviewing users and providers. Users are persons or groups of persons who will actually use the portal, while providers are persons or groups of persons who will run the portal in order to provide its services to the users (in the context of this paper, the users are the insurance agents and the provider is the insurance company). Both users and providers have different, potentially competing requirements.

During the interviews for the IPSI portal, it became clear that some insurance companies already use supporting systems for their agents. These systems were examined in order to identify further requirements. After consolidating all the requirements from the different sources, the requirements catalog was corrected and extended as needed, and the requirements were checked for errors again.

3.2. Subsystem selection

In most cases, EC/EB systems are not developed independently of an existing hardware and software infrastructure. Usually, the EC/EB systems have to be integrated into that infrastructure by exchanging data with it. However, the exchange of data between the EC/EB system and existing software systems may not always be sufficient – sometimes, the use of existing functionality is necessary. Thus, the IPSI electronic commerce portal exchanges data with its subsystems as well as with the database systems of the client (e.g. UDS, BS2000). This way, the portal can supply the data of insured persons and their contracts to the insurance agent. Furthermore, the portal needs the functionality of a complex tariff computation module, e.g. for a life insurance. Existing software systems like the latter one are termed legacy systems. In order to realize each subsystem, it must be decided if existing software systems fulfill the client's requirements, and if an existing software system can be integrated or if it is necessary to develop new software.

For the IPSI electronic commerce portal, it was decided to integrate existing software systems for most subsystems. This decision was followed by a market analysis to determine which existing systems should be used. The analysis took into account non-functional

criteria such as price, availability, support and platform and led to the selection of MS Outlook 2000, Pirobase 4.0, SmartStore 2.0 and JavaMail 1.1 for the subsystems office, content management, procurement and communications (see Figure 3).

Office	Content Management	Electronic Procurement	Legacy Applications	Comm	Admin
e-Mail Folders	Product Portfolio	Office Material (Toner, ...)	Partner Database	Sending Reminders, Messages, etc.	User Management
Address Book	Company Handbook	Promotional Material (Flyers, ...)	Contracts Database	by Fax SMS e-Mail	Monitoring
Calendar	Marketing Information	Company Services (Courses, ...)	Tariff Computer		Search
To-Do List	Law Documents				Portal-wide Full Text Searches
Outlook <small>Microsoft</small>	pirobase <small>PIRONET</small>	SmartStore <small>smartstore</small>	Partner DB <small>citronet</small>	sendfax, yaps, JavaMail <small>stun</small>	

Figure 3. Subsystems of the electronic commerce portal.

3.3. Prototype development

Next, it had to be determined if the selected software systems provided a programming interface (API) or if an interface could be developed. This was achieved by developing prototypes on the basis of the use cases and requirements, with the goal to identify opportunities for integrating the software systems with each other. For each software system, key features were defined that had to be realized by the prototype. The prototypes should show if the features of the underlying software system could be accessed through its interface.

Based on the prototypes, the effort, cost and time for the development of the whole EC/EB system could be estimated. This estimate was used to verify the "time-to-market" that was aimed for by marketing, and to plan accompanying measures like advertising etc. In the case of the IPSI electronic commerce portal, more resources were necessary for the development of an interface to integrate MS Outlook 2000 than for the development of the communications subsystem based on Java libraries. The effort required to integrate the partner database legacy system was relatively low since the adaptor could be realized using XML [11].

However, this is not always the case. Depending on the type of legacy system, integration may be more difficult. For example, under some conditions the integration of an SAP R/2 system with an electronic commerce system can only be achieved through the generation of batch input folders [Sche00].

3.4. GUI development

The graphical user interface for an EC/EB system is developed in two steps. First, a user interface prototype is designed. This prototype is also used by marketing to support accompanying advertising measures.

The prototype development begins with writing a storybook that is based on the use cases. This storybook is then used to define a style guide and, in a second step, to realize and implement the user interface for the EC/EB system. For the IPSI electronic commerce portal, this was done for multiple access channels (WWW, WAP).

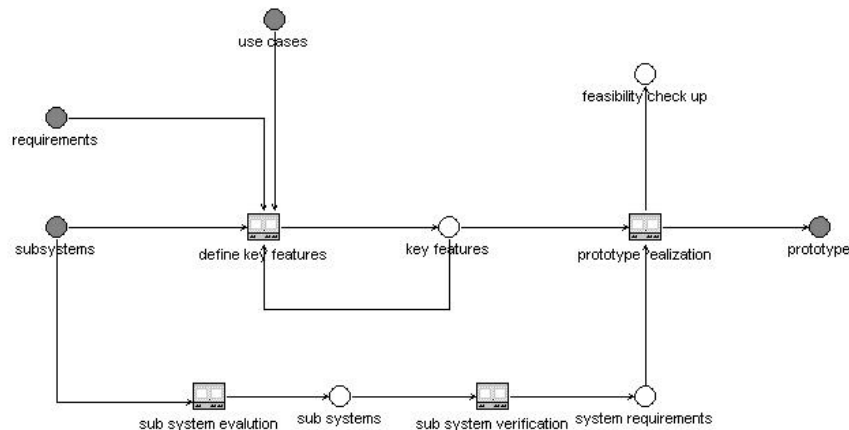


Figure 4. Prototype development subprocess model.

Besides the portal's specific functionality in the B2E application domain, its content is also a significant element. The content comprises all the information that the electronic commerce portal provides, as well as its presentation within the user interface. Content often has multi-media characteristics, i.e. it comprises textual information, graphic figures, moving and still pictures and audio information. Consequently, a content manager who is responsible for the multi-media information is an important role in the software development process. This is a new role that can comprise other roles, such as the media author who collects information and reworks it for a consistent presentation; the media designer who is responsible for the audio-visual design of the user interface; and the media producer who researches available media, creates images, graphics, animations, audio and video sequences, and clarifies copyright issues. Media editors are responsible for quality assurance in the multi-media content part of the application.

In addition to the role of content manager with its many tasks and responsibilities, the role of ergonomics advisor has to be taken by a team member. The ergonomics advisor's task is to ensure that the user interface of the electronic commerce portal is design ergonomically, i.e.

- it is suited to the tasks the user has to accomplish
- it guides the user by being self-explanatory and gives additional help on request
- it lets the user decide how to use the system without forcing them to follow a predefined set of procedures

- it describes errors the user makes and allows their correction with limited effort
- it can be adapted to the user's level of experience

User manuals can be differentiated into tutorials and references. For the creation of the user manuals, a styleguide is used that describes what the complete user documentation will look like. For the creation of a tutorial, the storybook that was already used for the user interface prototype is used again.

3.5. Integration and system test

In the realization phase, the design – to be more precise, the concrete system architecture – was implemented in the Java programming language. In this phase, elementary parts of the system architecture (controllers, adaptors, and formatters as described in section 2) were incrementally implemented. These parts are termed components. All realized subsystems subsequently went through a component test. Based on the use cases, test data sets were created to test the subsystems' functionality.

In the integration phase, the tested components were then integrated into the electronic commerce portal. The completely integrated system was then subjected to a system and integration test. To do this, the test data sets used for the component test were extended, and new sets were created. After a successful system test, the electronic commerce portal was delivered to the client together with the user tutorial in the system delivery phase.

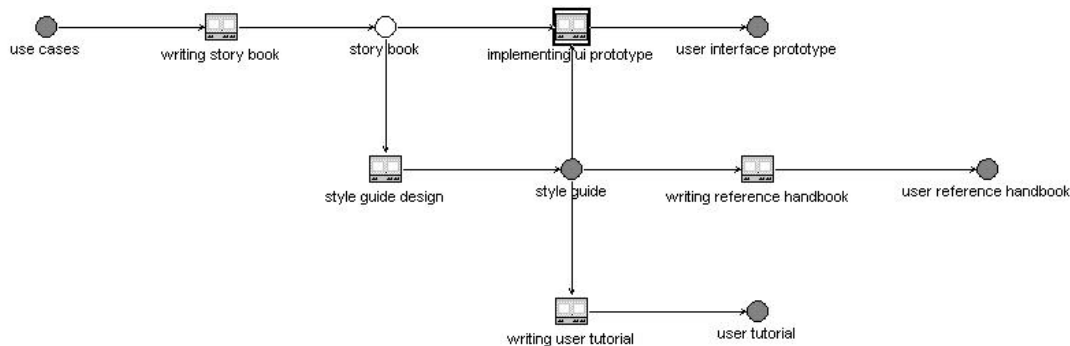


Figure 5. User interface design subprocess model.

4. Conclusion

It has become clear that the software development processes for electronic commerce systems can be partially different from conventional software development processes regarding

- the types of tasks
- the order in which the tasks are performed
- the roles that perform the tasks
- the software tools used

For this reason, it makes sense to examine conventional software development processes with regard to their optimal suitability (time, cost, resources) for this new class of software systems.

The development process for the IPSI electronic commerce portal is characterized by the high effort that was necessary to integrate the subsystems. This experience can be transferred to the development of other EC/EB systems, because the system usually always has to be integrated into a pre-existing software and hardware infrastructure. The integration effort comprises not only the design and realisation of interfaces (APIs), but also the test of those interfaces. The more complex the subsystems are, the more effort is required for the interface test since the necessary test drivers and stubs have to be equally complex.

Every introduction of an EC/EB system on the market should happen "time-to-market". Consequently, an estimate of the feasibility, effort and especially the duration of the development project has to be made early.

A solid estimate of these factors can be achieved through incremental and iterative prototyping.

As in every software system, features supporting the user (e.g. a self-explanatory user interface and online help) should not be neglected in EC/EB systems, either. It is important that the user-supporting features are tailored to the intended audience of the EC/EB system. For example, in the e-government area with its very heterogeneous audience, user-supporting features are mandatory. The same is true for EC/EB systems used in an intranet, like the electronic commerce portal for insurance agents. The software development process must contain tasks for the creation of these features.

The way the user interface of an EC/EB system is designed significantly contributes to the user acceptance of the system. This means that the software development process must include the creation of a user interface prototype that can serve as a marketing tool and be a basis for discussions with ergonomics specialists. For host-based application systems, the user interface design opportunities are limited. Here, acceptance must be based on functionality.

Unfortunately, quality-assuring measures can be victims of the "time-to-market" philosophy. This is true for all EC/EB system development projects and can also be observed in the development process for the IPSI electronic commerce portal. However, the goal must be to model software development processes that ensure a consistent high quality of EC/EB systems despite the changed and dynamic conditions, and take into account the shorter development times for these systems.

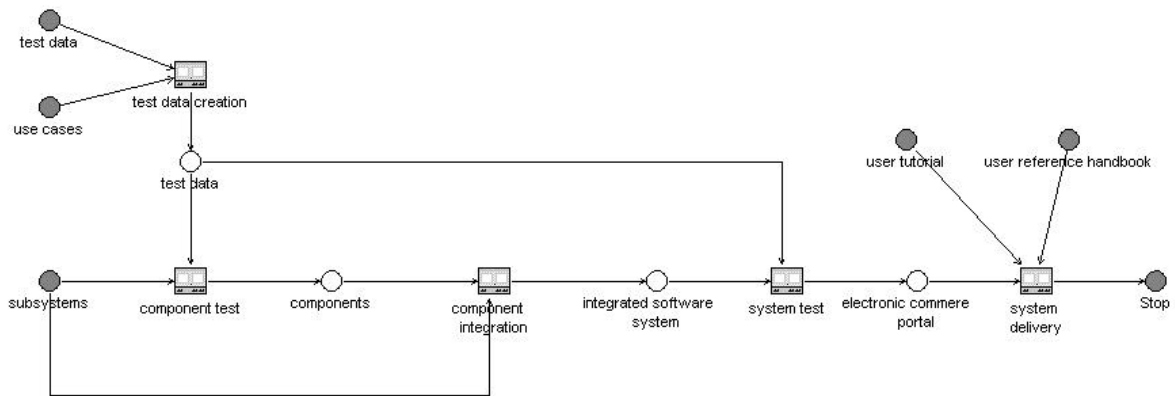


Figure 6. Integration and system test subprocess model

References

- [1] Baker, S.; Geraghty, R.: Java for Business Objects. In: Carmichel, A.: *Developing Business Objects* (1998), SIGS Cambridge University Press, pp. 225-237
- [2] Adam, N.R.; Yesha, Y. (eds.): *Electronic Commerce: An Overview*. In: Adam, N.; Yesha, A.: *Electronic Commerce*. LNCS 1028, Springer Verlag, Berlin (1995), pp. 4-12
- [3] Bayer, F.; Junginger, S.; Kühn, H.: A Business Process-Oriented Methodology for Developing E-Business Applications. In: Baake, U.; Zobel, R.; Al-Akaidi, M. (eds.): *Proc. 7th European Concurrent Engineering Conference* (2000), SCS Publishing House, pp. 123-132
- [4] Book, M.; Gruhn, V.; Schöpe, L.: Realizing An Integrated Electronic Commerce Portal System. In: Chung, M. (ed.): *Proc. of the Americas Conf. on Information Systems AMCIS 2000*, Ass. for Information Systems (2000), pp. 156-162
- [5] Chesher, M.; Kaura, R.: *Electronic Commerce and Business Communications*. Springer Verlag, Berlin Heidelberg New York (1998)
- [6] Deiters, W.; Gruhn, V.: The Funsoft Net Approach to Software Process Management. In: *Int. Journal of Software Engineering and Knowledge Engineering*, Vol. 4, No. 2 (1994), pp. 229-256
- [7] Hasselbring, W.; Koschel, A.; Mester, A.: Basistechnologien für die Entwicklung von Internet-Portalen. In: Heuer, A.; Leymann, F.; Priebe, D.: *Datenbanksysteme für Büro, Technik und Wissenschaft (BTW)* (2001), pp. 517-526
- [8] Harrison, W.; Ossher, H.; Tarr, P.: Software Engineering Tools and Environments. In: Finkelstein, A.: *Proc. 22nd Int. Conf. on Software Engineering* (2000), pp. 263-277
- [9] Lamond, K.; Edelheit, J.: Electronic Commerce Back-Office Integration. In: *BT Technology Journal*, Kluwer Academic Press, Vol. 17, No. 3, 1999, pp. 87-96
- [10] Lincke, D.; Zimmermann, H.: Integrierte Standardanwendungen für Electronic Commerce – Anforderungen und Evaluationskriterien. In: Hermanns, A.; Sauter, M.: *Managementhandbuch Electronic Commerce* (1999), Verlag Franz Vahlen München, pp. 197-210
- [11] Haifi, L.: XML and Industrial Standards for Electronic Commerce. In: *Knowledge and Information Systems*, Vol. 2, No. 4, Springer Verlag London (2000), pp. 487-497
- [12] Shaw, M.J.: Electronic Commerce: State of the Art. In: Shaw, M.; Blanning, R.; Stader, T.; Whinston, A. (eds.): *Handbook on Electronic Commerce*, Springer Verlag, Berlin, Heidelberg New York (2000), pp. 3-24
- [13] Zwass, V.: Electronic Commerce: Structures and Issues. In: *Int. Journal of Electronic Commerce* (1996), Vol. 1, No. 1, pp. 3-23
- [14] Zwass, V.: Structure and Macro-Level Impacts of Electronic Commerce: From Technological Infrastructure to Electronic Marketplaces. In: Kendall, K.E.: *Emerging Information Technology* (1999), Sage Publ., pp. 517-526