Software Processes for the Development of Electronic Commerce Systems

Volker Gruhn [1], Mykhaylo Dubinskyy [2]

1) Department of E-Business/Applied Telematics, Faculty of Mathematics and Computer Science, University of Leipzig, Germany, e-mail: volker.gruhn@informatik.uni-leipzig.de, http://www.informatik.uni-leipzig.de/telematik

2) Department of Information-Computing Systems and Control, Institute of Computer Information Technologies, Ternopil Academy of National Economy, Ukraine, e-mail: md@tanet.edu.te.ua, http://www.tanet.edu.te.ua

Abstract: *The development of electronic commerce systems is subject to different conditions than that of conventional software systems. This includes the introduction of new activities to the development process and the removal of others. An adapted process must cope with important idiosyncrasies of EC system development: EC systems typically have a high degree of interaction, which makes factors like ergonomics, didactics and psychology especially important in the development of user interfaces. Typically, they also have a high degree of integration with existing software systems such as legacy or groupware systems. Integration techniques have to be selected systematically in order not to endanger the whole software development process. This article describes the development of an EC system and it generalizes salient features of the software process used. The result is a process model that can be used for other highly integrative EC system development projects. The processes described are determined by short process lifecycles, by an orientation towards integration of legacy systems and by a strict role-based cooperation approach.*

Keywords: Electronic Commerce System, Software Process Model, Software Process.

1. INTRODUCTION

In this paper, electronic commerce (EC) is defined as conducting transactions of any kind by means of electronic media, especially the Internet. The roles of suppliers and customers in these transactions can be adopted by different parties, such as consumers (C), administrations (A), businesses (B) or even their employees (E) [20,21,22]. The parties involved in EC transactions use information technology systems to automate their transactions [2,5]. The examples used in this paper are business-to-employee (B2E) EC systems. The EC portal system discussed in this article is an integration platform for different software systems: conventional (i.e. non EC) software systems such as legacy and office systems as well as other EC based systems such as shop systems.

In the same way that conventional application software systems are developed according to conventional software development processes, special software development processes are necessary to delineate the development of EC systems [3,7,9,11]. Generally the software development processes for EC systems differ from those for conventional application software systems, but some paradigms, as for example the Enterprise JavaBeans (EJB) or DCOM component models are well-suited for both types of systems [23,25].

The difference between development processes is because of the difference in their general purpose. While conventional application software systems might win user acceptance mainly through their functionality, a special class of EC systems have to win user acceptance mainly due to the user interface and performance. Statistics says that usually user tend to quit his visit of site after 8-15 seconds waiting for response [17]. It causes the loss of the prestige of the company that is represented on the site. That’s why the tasks concerning the selection of content, presentation, characterization of customer behavior, workload forecasting and performance modeling are very prominent [24]. The characteristics of EC customer behavior like peak-like request bursts and high-volume data requests are also not typical for conventional software systems [16]. Specialists for software ergonomics, didactics, graphical design and psychology should be used to perform the tasks mentioned above.

The roles that are involved to the development of EC system are more specialized and more widely spread between participating suppliers than is normally the case with conventional software systems development. In most cases, the variety of required skills could not be found in one single supplier such as a software company. The way out could be found in collaboration between many suppliers with specialized skills such as multimedia design companies, software companies including freelancers as experts and content providers. A process for the development of EC systems has to take this distribution into account, by considering contract settlement and means for easing communication between suppliers.

As argued previously, when developing EC systems, a special software development process is needed to take into account these factors. This paper presents such a process that has been defined during the development of a B2E EC portal system.

2. PROCESS MODEL DEVELOPMENT

In general, the software development process for the development of a certain system is defined by a process model. A process model presents all the activities, the required tools and the created intermediate or final products necessary to achieve the process's purpose. A process model is usually tailored to a certain development project. 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 activities that are delineated in the process model are actually performed.

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

One trend in the software process community is to promote processes which focus on components and on building systems from components. Examples are Catalysis [26] and SELECT [27]. Both describe in detail how components could be specified and how the integration of components into systems should look like. Some of the elements used in the process for developing IPSI are related to these process models.

The Internet Portal System for Insurances (IPSI) is intended to provide support for insurance agents with their daily work. The main goal of the portal is to support business-to-employee processes [15] that are including the communications between insurance company and its agents, providing information about the product portfolio, tariffs and customer contacts via portal and its subsystems. This portal system demonstrates some of the idiosyncrasies of EC systems that generate the demand for adapted software development processes.

During the requirements analysis phase of the project, it was recognized that the EC portal will serve as an integration platform for different heterogeneous subsystems [8]. Based on an n-tier-architecture, the user interface and data repository will be separated from the functional business logic [12] that will reside in multiple application components (called subsystems). These highly integrative characters of IPSI will substantially impact onto the software process chosen.

The process model for the development of the IPSI EC portal – is presented schematically in Fig.1, using the FUNSOFT net notation [6]. In order to reduce the complexity of presentation and increase the number of levels of abstraction, this notation allows distinction between elementary tasks (e.g. write story book, perform test data creation) and subprocess models (e.g. requirements analysis, subsystem evaluation, prototype development), which can again contain elementary tasks and subprocess models.

The object-oriented design using UML, prototype development, adaptors implementation to integrated software systems as subsystems of the EC portal 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 sub process models such as the user interface specification and development.

3. PROCESS DESCRIPTION

Requirement specification. The requirements analysis starts with a competition analysis, subsequent proposal and contract evaluation, and project initialization. After this, the functional and non-functional requirements for EC software system are identified.

For the development of the EC portal for insurance agents, a competition analysis should determine if other software companies already offered a similar systems and which targets groups those companies are aimed at. Afterwards, the product idea was presented to several insurance companies, and one insurance company was chosen 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 the creation of a basis on which a software system could be developed that met the client's functional and technical requirements.

The initial list of requirements resulting from the competition analysis is the starting point for the creation of a requirements catalog for the entire EC portal to be developed. This requirements catalog is checked for contradictions, redundancy and completeness in several ways; for example, by interviewing users and providers that have different, potentially competing requirements (in our case the insurance agents and the insurance company).

During the interviews some supporting systems for insurance companies are examining in order to identify further requirements. After consolidating all requirements from the different sources, the requirements catalog is corrected and extended while the requirements are re-checked for errors.

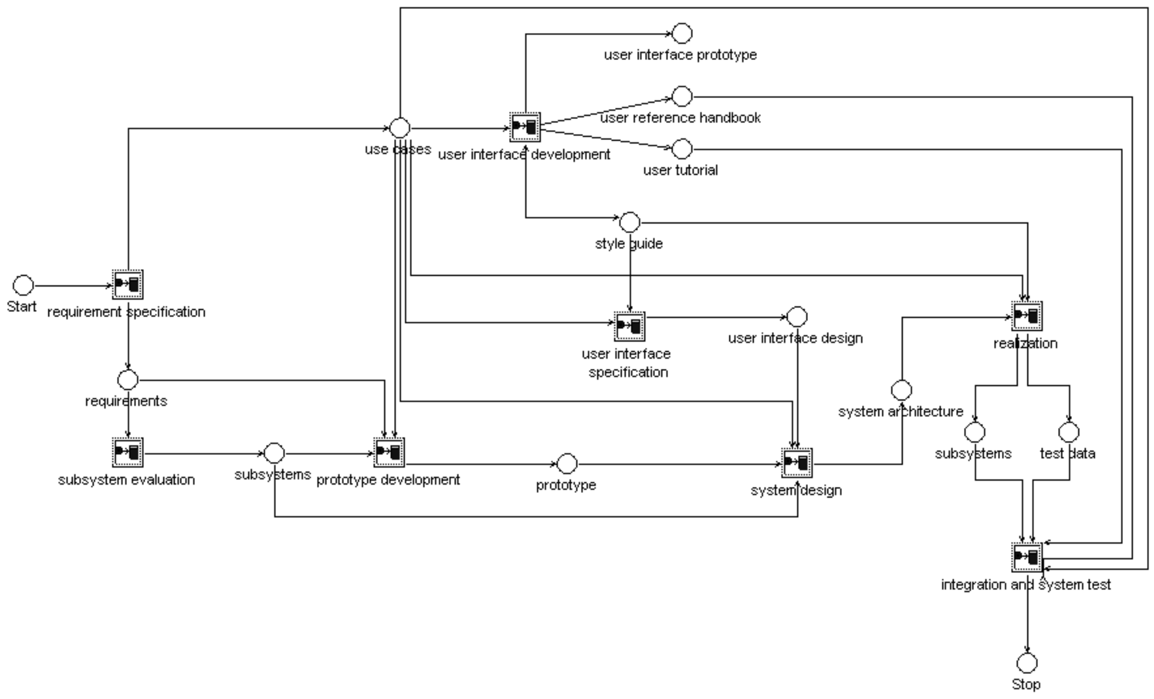

Fig.1 – Electronic commerce portal development process model.

Subsystem evaluation. Usually, EC systems have to be integrated into the existing infrastructure by sharing data with its systems. However, the sharing of data between EC 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 insurance company. In this way, the portal can supply the insurance agent with data of people insured and their contracts. Furthermore, the portal needs the functionality of a complex tariff computation module, e.g. for a life insurance. For the IPSI electronic commerce portal, it is good idea to integrate existing software systems for most subsystems. But this decision should be followed by market analysis with determination of the exact systems that should be integrated. The analysis also has to take into account non-functional criteria such as price, availability, support, platform, and possibilities of integrating the system. I the case of IPSI's office, content management, procurement and communications subsystems the most respective are: Microsoft Outlook 2000, Pirobase 4.0, SmartStore 2.0 and some freeware communication applications.

Prototype development. In addition, it had to be determined that if the software systems selected provided a programming interface (API), or if an interface could be developed. This can be achieved by developing prototypes on the basis of key features (major requirements in form of use cases), with the goal to identify opportunities for integrating the software systems between each other. First, for each software system, the key features that will be realized by the prototype have to be defined. The prototype has to 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 system could be estimated. In the case of the IPSI electronic commerce portal, more significant resources were used for the interface development of MS Outlook 2000 than for the development of the communications subsystem based on Java libraries. But sometimes the integration of some systems can be very difficult. For example, under some conditions the integration of an SAP R/2 system with an EC system can only be achieved through the generation of batch input folders and could therefore require more attention in terms of resource capacity devoted to that integration task.

GUI development. According to the IPSI Process Model, the graphical user interface for EC system is developed in two steps. First, a user interface prototype is designed. The prototype development begins with writing a storybook based on 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 system. For the IPSI electronic commerce portal, this was done for multiple access channels (WWW, WAP).

Due to the portal's specific functionality in the insurance B2E application domain, its content is a significant element. Content comprises all the information the EC portal provides, as well as its presentation within the user interface. Content often has multimedia characteristics, i.e. it comprises textual information, moving and still pictures and audio information. Consequently, a content manager responsible for multimedia information plays an important role in the software development process. This new role can consists of several other roles, such as the media author who collects textual information and reworks it for a consistent presentation; the media designer 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 multimedia content part of the application.

In addition to the role of a content manager, with its many tasks and responsibilities, the role of an ergonomics advisor has to be taken on by a team member. The ergonomics advisor's task is to ensure that the user interface of the EC portal meets ergonomic criteria, 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 him or her to follow a predefined set of procedures;
- it signals and describes user errors and allows their correction with little 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 style guide is used that describes what the complete user documentation should look like. The storybook that is used for the user interface prototype also can be used for the tutorial development (Fig.2).

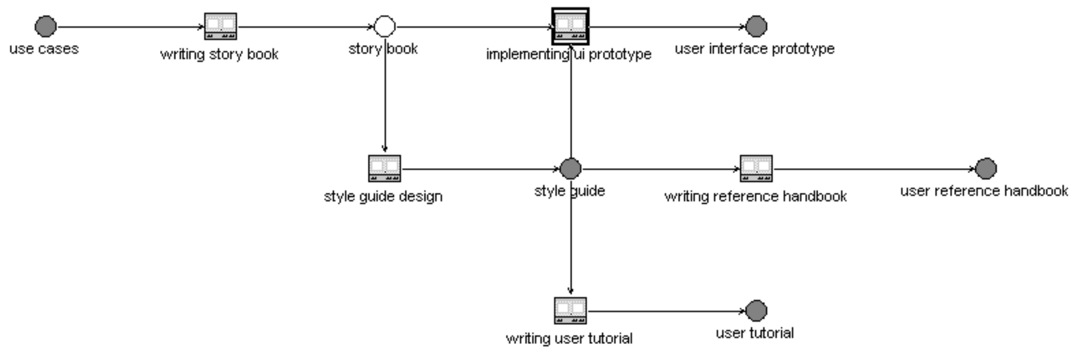

Fig.2 – User interface design subprocess model.

Integration and system test. In the implementation phase, the system architecture is being implemented (e.g. in Java). In this phase, elementary parts of the system architecture (controllers, adaptors, formatters and business objects) should be incrementally implemented, all class tests should be performed and classes should be combined to subsystems.

Every implemented subsystem subsequently has to pass a component test. Based on the use cases, test data set is created to test the functionality of every subsystem. This test usually combines black box and white box tests as described in [28]. While white box technique is applied to components for which the source code is available, black box technique have to be applied to all components purchased from vendors, since these usually do not provide access to their source codes.

In the integration phase, the tested components should be integrated into the EC portal, after that the complete integrated system is becoming the subject to system and integration tests. This test is performing, using test data sets created by extending the component tests. After successful system test, the EC portal is considered to be completed and to be ready for delivering to the customer.

4. EVALUATION OF THE PROCESS

The process chosen for the development of IPSI provides some features which are closely related to features of EC systems. It is influenced by the software process work as described in [29,30]. The most important features of EC systems reflected by the chosen software process are:

1. The development process for the IPSI electronic commerce portal is characterized by the high effort necessary to integrate the subsystems. The integration effort comprises not only the design and implementation of interfaces (APIs), but also testing 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.
2. It is rather difficult to assess the feasibility of developing an EC system, because new, unproven technologies have usually to be used. What makes estimates even more complex is the fact that in some cases the effort needed to implement a specific component depends on implementation details (like the side effects of using RMI). These details can only be clarified by developing prototypes.

3. Usability engineering is a crucial task. In EC systems users are a heterogeneous and usually not personally known set of people. To check whether are not navigation details and site structure suit them well is difficult and – in general – demands for usability engineering methods.
4. Testing urgently demands the integration of different types of testing techniques. This relates to the integration of black box and white box techniques as well as the combined use of component and system tests.

As mentioned in [7,10,30] these challenges are typical for EC system. Even though some of these challenges are properly addressed by traditional software process models, the process model applied to the development of IPSI concentrates on these challenges and therefore provides a solution to the problem of developing EC system.

5. CONCLUSION

Software processes for EC system are different from traditional software processes (as, for example, used in the development of information systems). Even though not a single of the identified challenges for the development of EC systems is completely new, process models which focus on EC systems (and which therefore are a natural choice for the development of such systems) are not available. Our approach was to start with a real problem (the development of IPSI), to model the process as it was carried out and to generalize the process in the process model.

Result is a rather lean software process model which covers most aspects of EC systems and which is flexible enough to be easily extended, if needed. The IPSI development process have included all the activities mentioned in section 3, but there are some other aspects to be kept in mind when developing EC systems, not included adequately in the IPSI development process. For example, consideration of performance issues is extremely important, especially when using highly layered object-oriented architectures for Web applications. Thus, performance modelling and testing [16,24] should be a central activities in any software development process for EC systems. In general, quality-assuring activities of any kind are often victims of the "time-to-market" philosophy. Here, the goal must be to construct software development processes that ensure a consistent high quality of EC systems, despite the changed and dynamic conditions, and take into account the shorter development time for these systems.

Our future work will be devoted to applying the proposed software process model to other types of EC systems (e.g. to a business-to-consumer system) and to integrated subprocesses which focus on activities like performance modelling and after-release monitoring which have not been appropriately considered yet.

6. REFERENCES

[1] Baker, S.; Geraghty, R.: *Java for Business Objects*. In: Carmichel, A.: Developing Business Objects, SIGS Cambridge University Press, (1998), 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*, SCS Publishing House, (2000), 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] Haire, B., Henderson-Sellers, B, Lowe, D.: Supporting Web Development in the OPEN process: Additional Tasks. In Proc. 12th COMPSAC, (2001), pp-383-389

[8] 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

[9] 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

[10] Hoffner, Y., Ludwig, H., Grefen, P., Aberer, K.: Crossflow: Integration Workflow Management and Electronic Commerce. IN: SIGecom Exchanges, Vol. 1, No. 2, ACM Press (2001), pp. 1-10

[11] Gruhn, V., Schöpe, L.: A Software Process for an Integrated Electronic Commerce Portal. In: Ambriola, V. (ed.) Proc. 8th EWSPT, Springer Verlag, Berlin, Heidelberg, New York (2001), pp. 90-101

[12] Lewandowski, S.: Frameworks for Computer-Based Client/Server Computing: In: ACM Computing Surveys, Vol. 30, No. 1, ACM Press (1998), pp. 3-27

[13] Lamond, K.; Edelheit, J.: Electronic Commerce Back-Office Integration. In: BT Technology Journal, Kluwer Academic Press, Vol. 17, No. 3, (1999), pp. 87-96

[14] Laartz, J., Scherdin, A., Carfarell, D., Hjartar, K.: Evolve your architecture. IN: CIO Magazine, Issue September, 15 (2000)

[15] Lincke, D.; Zimmermann, H.: Integrierte Standardanwendungen für Electronic Commerce – Anforderungen und Evaluationskriterien. In: Hermanns, A.; Sauter, M.: Managementhandbuch Electronic Commerce, Verlag Franz Vahlen München, (1999) pp. 197-210

[16] Menasce, D.A, Almeida, V.: Scaling for e-Business, Models, Performance, and Capacity Planning, Prentice Hall, (2000)

[17] Nielsen, J.: Designing Web Usability: The Practice of Simplicity, Riders Publ., (2000)

[18] Noffsinger, W., Niedbalski, R., Blanks, M., Emmart, N.: Legacy Object Modeling speeds Software Integration. In: CACM, Vol. 41, No. 12, ACM Press (1998), pp.80-89

[19] 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

[20] 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

[21] Zwass, V.: Electronic Commerce: Structures and Issues. In: Int. Journal of Electronic Commerce (1996), Vol. 1, No. 1, pp. 3-23

[22] Zwass, V.: Structure and Macro-Level Impacts of Electronic Commerce: From Technological Infrastructure to Electronic Marktplaces. In: Kendall, K.E.; Emerging Information Technology (1999), Sage Publ., pp. 517-526

[23] Brown, A.W.; Wallnau, K.C.: The Current State of CBSE, IEEE Software 9/10, 1998

[24] Menasce, D.; Almeida, V.: Capacity Planning for Web Performance – Metrics, Models and Methods, Prentice Hall 1998

[25] Szyperski, C.: Component Software - beyond Object-Oriented Programming, Addison-Wesley 1998

[26] D'Souza, D.; Wills, A.: Objects, Components, and Frameworks with UML - The Catalysis Approach, Addison-Wesley 1998

[27] Allen, P.; Frost, S.: Component Based Development for Enterprise Systems, Cambridge Univ. Press, 1998

[28] Beydeda, S.; Gruhn, V.: A Graphical Representation of Classes for Integrated Black- and White-Box Testing, in: Proceedings of the International Conference on Software Maintenance 2001, November 2001, Firenze, Italy

[29] Alloui, I.; Cimpan, S.; Oquendo, F.: Monitoring Software Process Interactions: A Logic-Based Approach, in: Proceedings of the 8th European Software Process Workshop, LNCS 2077, Springer

[30] Cignoni, G.: Reporting about the Mod Software Process, in: Proceedings of the 8th European Software Process Workshop, LNCS 2077, Springer