# COMPARING POSSIBLE E-PROCESSES

Frina Albertyn

Eastern Institute of Technology
Napier, NZ
falbertyn@eit.ac.nz

Proceedings of the 16th Annual NACCQ, Palmerston North New Zealand July, 2003 (eds) Mann, S. and Williamson, A. www.naccq.ac.nz

## ABSTRACT

E-commerce takes place in a fast changing and competitive environment. It is important to produce high quality e-commerce systems to meet the needs of e-commerce users. Developers and customers need guidance on what development processes are available and what areas of the whole development process are covered in the different processes, in order to make choices. This paper compares the Rational Unified Process, Open Source Development Process and Agile Modeling in order to give some direction in making a choice between competing e-processes.

## 1. INTRODUCTION

Software development and specifically e-commerce development can be complex or easy. More than half of all software development takes place without a defined methodology being used, which is usually not a problem with a small e-commerce site. Not using a methodology with complex sites can lead to high failure rates (Astels *et al.* 2002; Kaschek *et al.* 2003).

E-development requires different approaches in its development. Some of the differences are that an e-commerce site will be published on the Internet and that profiling the user is more difficult in e-commerce systems because it is challenging to uniquely identify the client (Schewe *et al.* 2002; Kaschek *et al.* 2003).

E-development focuses on two parts; the front-end, being the interface used to communicate with the user over the Internet, and the back-end, which provides the capabilities necessary to capture and process customers' orders, control inventory, and process product distribution.

The e-processes being investigated in this paper are: The Rational Unified Process (RUP), Open Source Software Development (OSP) and Agile Modeling with Extreme programming (AM/XP). RUP was chosen because many tertiary institutions use Rational Rose as a case tool for development. Some authors believe that RUP is over developed and has become too large for ease of use. (Schewe 2000; Hesse 2002) Agile was developed as a possible solution to RUP's problems and is therefore also included. The NZ public sector has just followed international trends by encouraging open source use, thus OSP is the third e-process included.

The fast changing e-commerce environment, combined with the nature of the development processes, demand that an informed choice be made on which process to use for development. A large number of development processes exist and for the new developer it is difficult to make an informed decision on which one to use. This paper compares some processes to assist with the identification of the best-suited e-process to a specific scenario.

## 2. THE E-PROCESSES

### 2.1 The Rational Unified Process

RUP is a popular development process, uses software engineering processes, has a well-defined structure and uses an object-orientated approach. RUP provides a whole development environment using UML (Unified Modeling Language) as its basis.

The horizontal dimension of RUP represents the dynamic aspect of the process in terms of time, cycles, phases, iterations and milestones (Larman 2002; Medvidovic *et al.* 2002). A software product is developed using a number of incremental iterations of the phases of development. The vertical dimension represents the static aspect of the process in terms of activities, disciplines, artifacts and roles (Bloomberg 2000; Graham 2001; Kruchten 2001). RUP involves five different views of the system's architecture; namely, Use-case view, Logical view, Implementation view, Process view and Deployment view (Reed 2002).

The four phases of RUP are:

**Inception:** Envision the project scope, vision and business case. Determine whether the stakeholders agree on the vision of the project and determine whether the project is feasible.

**Elaboration:** Develop a domain model, design model, software architecture document, data model, test model, implementation model, use-case storyboards and user interface prototypes, system sequence diagrams and events names.

**Construction:** Elaboration ends when the high risk factors have been resolved. The design aspects have been solved. Build the product and start developing user guides and online help.

**Transition:** Deploy the system operationally. Develop user guides and training materials. Data must be converted for use in the life system. Market and implement the system.

### 2.2 The Open Source Software Development Process

OSS development is based on the idea of using software released under a license as defined by the Open Source Initiative. The Open Source Initiative (OSI) is a non-profit corporation that manages and promotes the software. This software is free, re-distributable and with unlimited users and usage. The source code is available and can be modified to suit the requirements of the development process. (Feller and Fitzgerald 2000; OSI 2003).

In an OSS development approach, programmers and developer have the freedom to innovate and modify the code as required. Developers are potentially a large number of volunteers. The programmer has the ability to freely distribute modifications to software to others. A good developer knows what codes need to be newly created and what codes can be re-used. It is important that the developer responds to user requests quickly(Mockus *et al.* 2002).

Steps in the process include defining roles and responsibilities, identifying work to be done, assigning and performing development work, prerelease testing, and inspecting and managing releases (Mockus *et al.* 2002). Open source software development is an idea, which has reached maturity and is being used by the commercial world more and more (OSI 2003).

### 2.3 Agile Process

Agile modelling (AM) is a practice-based methodology for modelling and documenting software-based systems. The Agile Alliance promotes interaction and individuals over tools and processes, working software over extensive documentation, customer collaboration over contract negotiation and responds to change over sticking to the plan. AM focuses on a portion of the whole development process and needs to be used with Extreme Programming (XP). The idea is to start with XP and incorporate AM into it (AM 2001; Beck Kent 2001; Ambler 2002).

The values of AM are communication, courage, feedback, humility and simplicity. The principles of AM include model with a purpose, assume simplicity, embrace change, incremental change, multiple models, quality work, rapid feedback, software production is the goal, know your tools and models and maximise stakeholder investment. Some of the best practices for AM are stakeholder need to actively take part, collective ownership, create number of models in parallel, apply the right artefacts, depict models simply, iterative processes, prove with code and apply standards (Ambler 2002). The core practices of AM are: Continuous, active stakeholder participation; Apply the relevant models and artefacts to the correct application; Everybody owns the whole project and are allowed to work on any of the parts; Promote quality assurance and develop tests before developing the software; Develop several models parallel; Keep requirements, models etc. as simple as possible; Place developed models on a wall where they will be visible to the whole development team; Change focus to another part of the project if you get stuck on anything; Model small portions at a time; Communicate your ideas to others and get their input; Prove ideas with code; Use basic tools such as a

**Table 1: Process Characteristics**

| Characteristic | RUP | OSS | AM(XP) |
|---|---|---|---|
| Project Scope | Large projects. Many developers all with access to a central repository. | Large Projects. Many developers. Can accommodate a large user base. | Suited for small projects. Divide larger projects into smaller scaled ones for AM(XP) |
| Type of systems | All types. | Infra-structural, multi-user. | Small systems or subdivided smaller systems. |
| Principles involved | Develop iteratively, manage the requirements, use component-based architecture, visually model the software, verify software quality and control changes to software. | Freedom to innovate and modify. Ability to distribute modifications to software to others free. Know what to create and what to re-use. Rapid responses to user requests. | Work with the customers; Use metaphors to describe difficult processes; Plan; Short meetings; Test first; Keep it simple; Develop programs in pairs; Code to standards; Collective software ownership; Continuous Integration; Release early and often; Work short hours; Open to change. |
| Few possible reference site/s | www.therationaledge.com www.rational.com | www.opensource.org | www.agilealliance.org www.agilemodeling.com www.extrememodeling.org |

whiteboard and basic drawing tools and apply standards.

Business people and software developers often see the traditional software development methods as too slow.(Astels *et al.* 2002) Extreme modeling combines the advantages of methodologies based on the Unified Modeling Language with the advantages of Extreme Programming (XP). The best practices of UML are combined with the flexibility of developing and testing XP code. The philosophy of XP is to invest just enough effort to understand what is intended and then build it to see whether the design is right.(Ambler 2000; Boger 2002; Astels *et al.* 2002)

The Agile phases are:

**Conceptualize the system:** Create a vision of the system, Write user stories, Develop the acceptance tests, Find a solution and check the solution.

**Plan the new system:** Estimations, Plan releases, Develop an iteration schedule, Tactical planning.

**Develop the system:** Develop a system of pair programming, continuous testing, Design using the Agile values, principles and practices, Develop the code and refractor; Integrate daily.

**Deliver the system.**

Developers are often responsible for a company's decision to adopt AM(XP) as a development environment. It is important to convince managers that this development process has merit and depth.

# 3 COMPARISON OF RUP, OSS AND AM(XP).

The three development processes are compared according to their process characteristics, people involved and the development process.

## 3.1 Process characteristics

Table 1 addresses the basic characteristics of the three e-processes.

## 3.2 People characteristics

Table 2 addresses the people involved with the development process.

3.3 Development characteristics

Table 3 addresses the characteristics of the different e-processes.

(Hogarth; Perens 1999; Bloomberg 2000; Feller and Fitzgerald 2000; AM 2001; Kruchten 2001; Mockus *et al.* 2002; Astels *et al.* 2002; Larman 2002; Reed 2002; RUP 2002)

# 4. CONCLUSION

Future research should incorporate other development processes into comparisons. These should include Story Boarding and User Profiling. (Schewe *et al.* 2002/2003; Kaschek *et al.* 2003) Communicating with the customer should also have priority when a new e-commerce site is being developed. Communication and collaboration are the

**Table 2: Developers**

| Characteristic | RUP | OSS | AM(XP) |
|---|---|---|---|
| Roles and responsibilities of developers | Large development teams possible with all having access to the repository. | Potentially a large number of volunteer developers. | Pair development. Roles: Coach; Tracker; Facilitator and Architect. |
| Roles and responsibilities of users (customer) | End-users see the developed parts quickly and the users are engaged to provide feedback for adaptation, so that the end product can meet the needs of the stakeholders. | Users play a large role as testers, documenters as well as defining new requirements promptly | The customer decides scope, priority and release content. The customers form an integral part of the development process. The roles identified are storytellers, acceptors, resource providers, planners and advocate. |

critical success factors when building a new web site. (Wallace and Matthews 2002; Schewe *et al.* 2002/ 2003) All three development processes investigated, especially OSS and AM(XP), involve the customer on a high level.

The main implication from the investigation is that e-processes, though evolving, have not been sufficiently adjusted to accommodate all the challenges that e-commerce development holds. Choosing the right development process for specific situations is not an easy task and more research should be done to ascertain ways of identifying the development process best suited to a specific scenario.

## REFERENCES


AM (2001). The Agile Modelling site www.agilemodeling.com.

Ambler, S. W. (2000). "Extreme modeling - Design is fundamental to the XP process, regardless of what the name may imply." www.sdmagazine.com/documents/s-738/ sdm0011m/0011m.htm.

Ambler, S. W. (2002). Agile Modeling (AM) An overview, Ronin International. 2003 www.ronin-intl.com.

Audris Mockus, Roy T Fielding, et al. (2002). "Two case studies of open source development: Apache and Mozilla." ACM Transactions on Software Engineering and Methodology (TOSEM) 11(3): 309-346 doi.acm.org/10.1145/ 567793.567795.

Beck Kent, e. a. (2001). Manifesto for Agile Software Development. 2003 www.agilealliance.org.

Bloomberg, J. (2000). "Building E-businesses with the Rational Unified Process." www.rhodes.com/ articles/ebiz.html.

Boger, M. b. M. (2002). Welcome to xTremeModeling.org, XM - Extreme Modeling. 2002 www.extrememodeling.org.

David Astels, Granvilee Miller, et al. (2002) A practical guide to eXtreme programming. Upper Saddle River, Prentice Hall 0-13-067482-6.

Graham, T. (2001). Introducing the Rational Unified Process, IEEE Computer Society. 2002 www.computer.org/software/bookshelf/2001/ 5501bks_2.htm.

Hesse, W. (2002). "DinoSaur Meets Archeaeopteryx? Seven Thesis on Rational's Unified Process (RUP)."

Hogarth, M. A framework for Open Development of Terminologies mycin.ucdvis.edu/presentations/ opendevterminilogies.ppt.

Joseph Feller and B. Fitzgerald (2000). A framework analysis of the open source software development paradigm. Proceedings of the 21st Annual International Conference on Information Systems (ICIS 2000), Brisbane, Australia 58-69.

Kaschek, R., K.-D. Schewe, et al. (2003). "Story Boarding for Web-based Information Systems." Submitted to Journal

Kruchten, P. (2001). What is the Rational Unified Process?, The Rational Edge. 2002 www.therationaledge.com/content/jan_01/ f_rup_pk.html.

Larman, C. (2002) Applying UML and Patterns. Upper Saddle River, Prentice Hall PTR 0-13-095004-1.

Medvidovic, N., D. S. Rosenblum, et al. (2002). Modeling software architectures in the Unified Modeling Language. ACM Transactions on Software Engineering and Methodology (TOSEM), ACM Press 2-57.

OSI (2003). Open Source Initiave, OSI News. 2003 www.opensource.org.

**Table 3: Development process**

| Charact-eristic | RUP | OSS | AM(XP) |
|---|---|---|---|
| Development work | Detailed development phases have to be laid down. Iterative lifecycle and risk-driven development. | No explicit system-level design or detailed design. No project plan, schedule or list of the deliverables. | Not a complete process but focuses on effective modeling of requirements and documentation and should be combined with a process such as XP. |
| Phases identified | Inception, Elaboration, Construction and Transition. | Define roles and responsibilities; Identify work to be done; Assign and perform the development work; Prerelease testing; Inspections; Managing releases; | Conceptualize the system.<br>Plan the new system.<br>Develop the system.<br>Deliver the system. |
| Testing | Testing is organized around single components first and then gradually gets expanded to include larger sets of integrated components. Product and process quality must be monitored continuously. | Continuous testing | Tests are developed before coding. Everything that can possibly be tested should be tested. Test before and after refractoring as well as when a new task is implemented. Acceptance testing, Performance testing and Quality testing should all form part of the deployment plan. |
| Managing implementation and releases | Delivery of small releases, but transition phase consists of beta tests and then full deployment. | New parts of the project are released frequently. | Delivery of small releases to customers. |
| Modeling or artifacts used | **Business Modeling:** Domain Model, Partial artefacts in each iteration **Requirements:** Requirements, Constraints, Use-case models, Vision Statement, Supplementary Specifications and Glossary **Design:** Design model and Software architecture. **Project Management Plan. Test:** Test plan (acceptance tests from requirements – use cases) **Environment:** Development case | Process model needs to be developed. | Uses case models, class models, data models and user interface models. Develop different models in parallel. |
| Development tools or Case Tools available | Full software support for the whole development process. | Browsers, editors, query servers, authoring servers and distribution servers required. Developers Email list. 3GLs. Application area more complex. | Enhances other software processes, but does not have its own Case Tools. |
| UML modeling | The whole development environment using UML as a basis. | | UML provides a number of diagrams. XP has found 3 of these diagrams useful to help find solutions to problems namely the class, sequence and state diagrams. |

Perens, B. (1999). Open Sources: Voices from the Open Source Revolution, O'Reilly online catalog. 2003 www.oreilly.com/catalog/opensources/book/perens.html.

Reed, P. R. J. (2002) Developing Applications with Java and UML. Indianapolis, Addison-Wesley 0-201-70252-5.

RUP (2002). Rational Rose:Product Information, Rational the software development company. 2002 www.rational.com/products/rose/prodinfo.jsp.

Schewe, K. D. (2000). UML: A modern dinosaur? A critical analysis of the Unified Modelling Language. 10th European-Japanese Conf. on Information Modelling and Knowledge Bases., Saariselka/Finland

Schewe, K.-D., R. Kaschek, et al. (2002). Modelling Web-Based Banking Systems: Story Boarding and User Profiling. ER2002: Conceptual Modeling Approaches for e-Business: A Web Service Perspective (eComo 2002), Tampere Finland, ER2002 p. 63 - 74.

Schewe, K.-D., R. Kaschek, et al. (2002/2003). Emphasizing the Communication Aspects for the Successful Development of Electronic Business Systems. Emphasizing the Communication Aspects for the Successful Development of Electronic Business Systems

Wallace, C. and C. Matthews (2002). Communication: key to success on the Web. ER2002: Conceptual Modeling Approaches for e-Business: A Web Service Perspective (eComo 2002), Tampere, Finland p. 75 - 85.