TOWARDS A RAD FRAMEWORK FOR E-COMMERCE SYSTEMS

Ernest Cachia
University of Malta
Msida, Malta

Mark Micallef
University of Malta
Msida, Malta

Keywords: E-Commerce, Rapid Application Development, Software Metrication, Software Quality Assurance, Software Development Life Cycles.

Abstract: Given the ever-increasing revenues being generated from e-commerce systems (Emarketer.com, 2005), the quality of such systems is becoming ever more important. Industry studies carried out by the authors of this paper show that customers want higher quality systems built in ever-shorter timespans. This paper establishes the need for the creation of a Rapid Application Development (RAD) based framework specifically tailored towards e-commerce systems. A skeletal version of such a framework is also proposed.

1 INTRODUCTION

Year after year, statistics constantly indicate that revenues from online shopping are steadily increasing throughout the world (Emarketer.com, 2005)(J.H.Y. Yeung, 2003)(News, 2005). As customers get more comfortable with shopping online, it has almost become a natural step for business to offer their goods and services online. To many, this may seem like a simple issue involving the setting up an online catalogue and an online payment facility. However, in reality, things are more complex. For example, existing business processes need to be replicated online or changed accordingly to reflect the online aspect of the business (E. Lawrence, 2000). Also, the ways in which customer difficulties are dealt with take on a new dimension in the online world due to the potentially enormous customer base.

The authors of this paper carried out two studies in the course of their research. The first was a study amongst 350 online shoppers which examined various aspects of online shopping and consumer behaviour (E. Cachia, 2004). The second study was carried out amongst 10 software development companies who specialise in the development of e-commerce systems. The latter focused mainly on each company's business environment (types of clients, types of systems, etc) and their approach to requirements definition, quality assurance, software engineering processes and so on. Results from these studies will also be used to substantiate various arguments made throughout this paper.

2 WHY RAD AND E-COMMERCE?

It has often been observed that by the time a computerised business solution is delivered, it no longer satisfies the requirements of the client who ordered it. This is because in the months (and sometimes years) it takes for a system to be specified, designed, built, tested and deployed, the client's requirements tend to change (Howard, 2002). The nature of modern markets also has a hand in the dynamic nature of system requirements (Small, 2000). In 1991, James Martins proposed the concept of Rapid Application Development (RAD) (Martine, 1991). RAD is basically a "high speed" adaptation of the linear sequential model in which rapid development is achieved by using component-based construction (Pressman, 2000). The main idea is to reduce the time between determining requirements and implementing them, thus reducing the likelihood that they would change in the interim (Howard, 2002). This is achieved by a lean development cycle and a number of principles of techniques such as *Joint Application Development (JAD)*, *Time Boxin*, *Prototyping* and *Clean Rooms*.

Cachia E. and Micallef M. (2006).
TOWARDS A RAD FRAMEWORK FOR E-COMMERCE SYSTEMS.
In *Proceedings of WEBIST 2006 - Second International Conference on Web Information Systems and Technologies - Internet Technology / Web Interface and Applications*, pages 234-241
DOI: 10.5220/0001242102340241

There are a substantial number of indicators that seem to show that e-commerce systems development would benefit from a RAD approach. These are discussed below.

Primarily, by their very nature, **e-commerce systems are business-oriented software systems**. They support a number of business processes which, as indicated by Martins (Martine, 1991), tend to be dynamic. Clearly, decreasing the timespan in such projects would reduce the likelihood that clients' requirements will change in the time it takes for a system to be developed. Participants in our industry survey unanimously agreed that customers' requirements are highly flexible and are likely to change quite often. Cases in which a client sticks to initial requirements without making changes are rare.

Our industry interviews show that **most e-commerce systems are developed within six months**. 60% of participants claim that their systems take no longer than three months to develop. Kerr (J. Kerr, 1994) points out that if a business application can be modularized in a way that enables each major function to be completed in less than three months, it is a candidate for RAD.

E-Commerce users can be very unforgiving. For example, in our study of 350 online shoppers, it was discovered that only 18.4% of shoppers would remain unconditionally loyal to their site if it suffered performance problems. High quality and reliability is therefore important in e-commerce systems. The RAD approach promotes reuse of components as these would have already been tested. Also, new components must be tested and all interfaces must be fully exercised (Pressman, 2000).

In the course of our research, we also delved into cases where RAD is likely to fail and analysed whether e-commerce systems would also be vulnerable in this area. This is discussed below.

Butler (Butler, 1994) lists four potential points of failure for RAD. Two of them are technical whilst the other two are more to do with human nature and commitment issues. Firstly, Butler stresses that not all types of applications are appropriate for RAD. In order for a system to be compatible with RAD, it needs to be modularisable. This does not pose a problem with e-commerce systems because by nature, they are modularised into high-level components such as online catalogues, shopping carts, customer support mechanisms, and so on (E. Cachia, 2005). A second potential technical point of failure is the level of technical risk within a project. If technical risks are high, RAD would not be the appropriate life-cycle (Butler, 1994). Our industry interviews reveal that most e-commerce systems make use of standard technologies and do not carry high technical risks.

The non-technical two potential points of failure mentioned by Butler are as follows. Firstly, RAD developers and customers need to be committed to the rapid-fire activities necessary to get a system complete in a much abbreviated timeframe (Butler, 1994). This is dependant on the people involved in individual projects and cannot really be characterised on the basis of the type of system being developed. Therefore, e-commerce systems suceptibility to this point is on the same level as any other type of system. Finally, Butler points out that RAD requires sufficient human resources to create the right number of RAD teams. Although this too depends on particular circumstances, there is some evidence to show that small and medium enterprises (SMEs) tend to dedicate more human resources to existing traditional business, even when developing new e-commerce systems because that is what currently earns them money (J.H.Y. Yeung, 2003). This suggests that development of systems for SMEs may not always be appropriate for RAD integration. However, this depends on individual situations and a customer's commitment to the e-commerce project.

The arguments presented in this section clearly indicate that e-commerce systems are likely to be highly suitable candidates for rapid application development.

3 THE NEED FOR A SPECIFIC FRAMEWORK FOR E-COMMERCE SYSTEMS

In the section 2, we showed that e-commerce systems are well suited for rapid application development. However, it is also being argued that RAD does not do a good enough job for e-commerce systems as it stands. One might suggest that development teams should simply utilise the existing RAD methodology as-is, as opposed to creating an RAD-based framework specifically tailored for e-commerce systems. However, in 2004, Cachia (E. Cachia, 2004) showed that e-commerce systems differ significantly from other types of systems. The characteristics that separate e-commerce systems from other types of systems are as follows.

Firstly, e-commerce systems tend to be **content driven**. Most, if not all, e-commerce systems would need to be connect to a data source from which the resulting web pages would be constructed. A typical

scenario would be an online catalogue being generated from a product database. This characteristic is important because when building such a system, developers need to come up with the best possible ways to present the information to the user. How will the user access the information? Which information will be given priority? How will it be organised? And so on.

Another important identifying characteristic of e-commerce systems is the fact that they are **exposed to security risks**. The open nature of the internet makes such systems susceptible to security risks. Hacking attempts are frequent and high profile cases persist. Only recently, CNN reported that a database with details of 40 million credit cards had been compromised (CNN, 2005).

Unlike other systems, online stores are mainly **accessed through WWW browsers**. This is an important difference because browsers adhere to the HTTP protocol which does not really provide rich event-driven programming which most of us are used to in other systems. Attempts have been made to fix this by means of scripting technologies but then again, such technologies are not uniformly interpreted by all browsers (K. Chandra, 2003).

Most businessman's ambitions when deploying an e-commerce system would be to cultivate an **enormous user base**. Few systems can boast the same potentially enormous user-base inherent in e-commerce systems. Although business-wise, this is a good prospect, it does raise a number of technical issues which a development team would need to consider. For example, is the system stable enough to take a hundred concurrent users? A thousand users? A million? More often than not, dealing with increased popularity is not simply a matter of scaling up hardware but it is also a matter of algorithmic efficiency (D. Menasc, 2001).

Finally, e-commerce systems are **likely to change quite often**. Most online stores will change frequently. Changes may be as simple as changing the price of particular items. However, customer requirements and a dynamic environments can require the system to change radically in order to maintain a competitive edge.

Based on these differences and a survey carried out amongst 350 online shoppers, Cachia (E. Cachia, 2004) identified five important quality attributes in e-commerce systems as being *security*, *reliability*, *navigability*, *performance* and *portability*. This indicates that when drawing up specifications and designing and e-commerce system, it is not only core system functionality that should be specified. In addition to what the system must do, there should be specifications such as the level of security to be employed, the browsers which the system should be compatible with, the number of concurrent users it can handle, the type of navigation schemes it should employ, and so on. Participants in our industry interviews were asked whether they specify such requirements in advance. The replies were varied with most companies specifying security and then one or two of the others quality attributes. However, most agreed that it is important to specify these attributes explicitly early on in the development process.

Considering the fact that Cachia (E. Cachia, 2004) showed considerably uniqueness in e-commerce systems, it is the opinion of the authors that having a RAD-based framework specifically tailored for the development of e-commerce systems would be beneficial. Such a framework would take the particular characteristics of such systems into account thus enabling faster development of higher quality systems. It would also give developers who utilise it a sense of security that all important aspects of the system have been catered for when they follow the framework.

4 FRAMEWORK REQUIREMENTS

Prior to proposing a framework, it would be desirable to set out a list of requirements and criteria which should be present in such a framework.

4.1 Rapid Application Development Basis

As discussed in section 2, e-commerce systems are ideal candidates for RAD development. The framework should take this into account and adapt tools and techniques from Martin's (Martine, 1991) work without, so to say, "reinventing the wheel".

4.2 Explicitly Cater for E-Commerce Systems Characteristics

The proposed framework should cater for particular e-commerce systems characteristics as described in section 3. That is to say, issues pertinent to such systems should be explicitly addressed in mechanisms and tools provided by the framework. For example, if the framework is to include a number of metrics, then instead of (or in addition to) generic metrics which

apply to all systems, there should be metrics for attributes such as security, portability, navigability, and so on. Such "features" in the framework would increase stakeholder confidence in the fact that important aspects will be automatically taken care of if they use the proposed framework.

4.3 Emphasis on Quality Assurance

Studies have shown that online shoppers are not very loyal (E. Cachia, 2004). If a site is too slow when loading, crashes at some point, or if navigation is too complex, users can simply move to another site offering the same service. Therefore it is of the utmost importance that the framework include a number of quality assurance measures in order to minimize major errors after deployment.

4.4 Provide Measurability

It is important that stakeholders be able to track the level of a number of criteria during the development process. This can vary from say a "percentage-complete" measure for a particular project or module to, for example, the security level built into the system. Having a numerical view of a number of aspects of the system will enable stake holders to make informed decisions.

4.5 Flexibility and Configurability

Different businesses will have different circumstances, environments, budgets, perceived client base, and so on (J.H.Y. Yeung, 2003). The proposed framework should therefore not be rigid, as this could be counter-productive in some cases. It would be ideal if developers who are using the framework could set certain parameters such as various levels of importance of certain criteria before starting. So if for example, a particular company is developing an e-commerce system where portability is not important (maybe because target users use one particular browser), the framework will automatically reduce or eliminate all checks and procedures related to portability.

4.6 Software Tool Support

Some of the criteria we are setting such as *Emphasis on Quality Assurance* (section 4.3) may seem to conflict with the rapid-development aspect of the framework. Although this is not necessarily true, users of the framework would consider software tools that automate parts (or all) of the framework to be very useful. Hence, the framework should provide some form of plugability for software tools whereby anyone wanting to automate part of the process could do so with ease.

5 PROPOSING A SKELETAL FRAMEWORK

Having argued for the need for a RAD-based framework for the development of e-commerce systems, we now propose a skeletal version of such a framework. It is a skeletal version in the sense that although we have identified individual components and how they may work together, most components are defined at an abstract level. This means that further research is required to concretise it.

5.1 Elements of the Framework

The proposed framework will consist of the following elements:

Principles, Guidelines and Techniques to provide users of the framework with the fundemtal knowledge and techniques for rapidly developing high quality e-commerce systems. These principles, guidelines and techniques will be utilised in conjunction with other elements throughout the framework.

Development Lifecycle which engineers can follow when developing an e-commerce system.

Metrics and Warning Mechanisms to provide a numerical view of systems under development and also to provide a means of warning stakeholders when recommended metrics thresholds are surpassed.

Quality Assurance Activities and Tools to ensure that systems developed with the framework are of high quality.

Software Support Tools which would be able to help stakeholders automated parts of the framework so as to speed things up.

A description of these elements and their integration is given below.

5.2 Principles, Guidelines and Techniques

The principles, guidelines and techniques provided by the framework would be very similar to those proposed by Martins (Martine, 1991) for RAD. That is to say there would be a principles and techniques such as *Function Prioritisation*, *Risk Analysis*, *Joint Application Development (JAD)*, *Time Boxing*, *Prototyping*

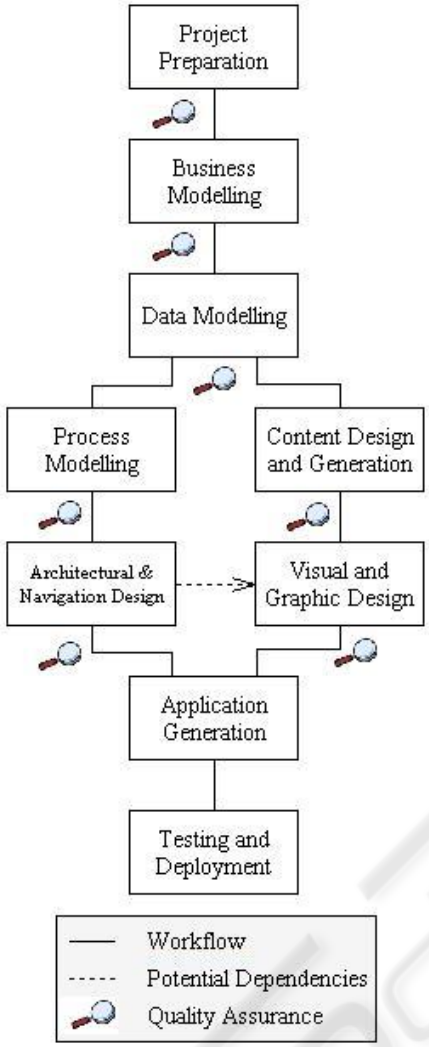

Figure 1: The proposed lifecycle.

and *Clean Rooms*. A heavy emphasis will also be placed on reuse of components which have already been utilised in other systems beforehand.

5.3 The Development Lifecycle

The development life-cycle is proposed to consist of nine stages, and is mostly sequential but having two parallel tracks at one point as depicted in figure 1. In accordance with RAD (Martine, 1991), this life-cycle is intended to be used simultaneously by multiple teams on different modules of a system. The stages are as follows:

Project Preparation - Involves the setting up of project scope, parameters, target users and so on. This stage is important in order for the framework to be able to be adjusted for the current project as specified in section 4.5. A typical preparation activity would be to define the level importance of various quality attributes for this particular project.

Business Modelling - Very similar to the business modelling stage in Martin's (Martine, 1991) RAD. Involves th modelling of information flow in a way that answers questions like: What information drives the business process? What information is generated? Who generates it? Where dose it go? Who processes it? (Pressman, 2000).

Data Modelling - Conctretises the information from the previous *Business Modelling* stage into data objects. The attributes of each object and the relationships between different objects are also defined. Again, this is adapted from the original RAD process (Martine, 1991).

Process Modelling - Another important adaptation from Martin's (Martine, 1991) work, process modelling will define operations on the data objects created in the *Data Modelling* stage. This is done so as to to achieve the information flow necessary for implementing one or more business functions. This stage is very important because according to Cachia, 35% of abandoned e-commerce transactions are due to the processes involved being too long and cumbersome (E. Cachia, 2004).

Architectural and Navigation Design - This stage focuses on how the data and processes will be structurally represented and linked in website used by potential customers. Questions such as the following need to be answered: How will information be organised? How will the user navigate through it? Will there be multiple navigation schemes? How will pages within the system be linked together? This stage is deemed to be critical because Cachia's study of 350 online shoppers (E. Cachia, 2004) found that users who are not happy with the navigational aspect of a site are very likely to simply move to another site.

Content Design and Generation - This stage would involve content experts designing, generating and assimilating content. This may include content such as product descriptions, company mission statement, frequently asked questions, and so on. This stage follows the *Data Modelling Stage* and can carried out in parallel to *Process Modelling* and *Architectural & Navigation Design*.

Visual and Graphic Design - Whether or not a site is visually pleasing has a profound effect on whether a potential client stays to browse or not, although online shoppers tend to give navigability a higher level of importance (E. Cachia, 2004). During this stage, a graphic designer is responsible for creating the look and feel of the site. This design

will later be incorporated into the actual application.

Application Generation - As soon as deliverables from the *Architectural & Navigation Design* stage are available, application generation can commence. At some point it will need to be merged with the deliverables from the *Visual and Graphic Design* stage although deliverables from the latter are not strictly necessary for application generation to begin.

Testing and Deployment - Involves utilising a variety of testing techniques (black box, white box, integration, regresion, etc) prior to release. Reused components need not be tested thoroughly but all new components must be tested and all interfaces fully exercised (Pressman, 2000).

5.4 Metrics and Warning Mechanisms

The framework should incorporate of a number of metrics for measuring various aspects of a system. Warning mechanisms can then be implemented by monitoring these metrics and warning if certain thresholds are breeched at any point.

One of the main issues in metrication is concerned with how one can compare values of the same metric across different systems. For this reason, the authors have created and published a metric to measure the functionality offered by an e-commerce system (E. Cachia, 2005). Each system is assigned a number of points, similar to function points (Albrecht, 1979)(Arthur, 1985) and any other metric readings across systems can be normalised with the functionality reading as a base. For example *errors per point*, *cost per point* and so on.

A number of metrics should be included in the framework and used by developers according to the particular circumstances of the project. Metrics can be categorised into *internal product metrics*, *external product metrics* and *process metrics*. The individual metrics to be used will be specified at a later stage when this framework is refined and concretised as discussed in the beginning of this section.

5.4.1 Internal Product Metrics

These metrics will be measure the internal quality attributes of the system. That is to say attributes which are not visible to the user but could have an influence on the visible quality aspects (i.e. external attributes) of the system (Kan, 2003). Such metrics would measure attributes such as:

- Security
- Testability
- Efficiency
- Maintainability

These metrics are not to be taken as the recommended list of internal attributes to be measured by the framework. Rather, they are examples of what might be recommended in future and such a list will be available once further work in this area has been carried out.

5.4.2 External Product Metrics

Metrics in this category will measure quality attributes which may be observed by the users of a system. Although numerous external metrics have been created throughout the years, very little work has been done on e-commerce related external metrics. The framework should provide a mix of generic and e-commerce related metrics so as to cater for attributes which are soley present in e-commerce systems while not abandoning generic attributes which also apply to e-commerce systems. Typical measured attributes would include:

- Reliability
- Performance and Scalability
- Usability and Navigability
- Portability

As in section 5.4.1, this list of attributes are not to be considered as the recommended list of attributes to be measured by the framework as further work still needs to be done before such a list is made available.

5.4.3 Process Metrics

Process metrics will be a tool for stakeholders to be able to keep track of how system development is going. They will help provide answers to questions such as *"Are we on schedule?"*, *"Is the project within budget?"*, and so on.

5.5 Quality Assurance Activities and Tools

Upon observing the development lifecycle proposed in section 5.3 and depicted in figure 1, one notices that there are quality assurance activities involved at ever stage of the lifecycle. At first, this may be seen to be contradictory to a rapid development cycle because quality assurance activities have the reputation of being time consuming and tedious. However, system quality has been shown to be of high importance in e-commerce systems (E. Cachia, 2004) and the framework should include a number of quality assurance

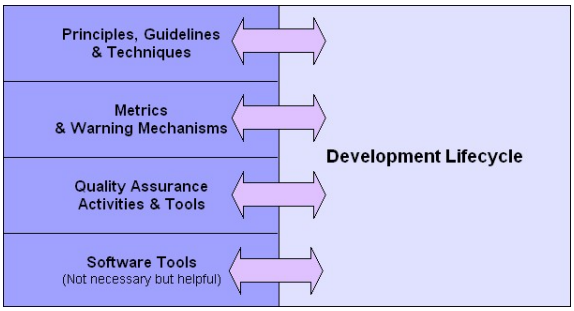

Figure 2: An overall view of the framework.

activities at each stage that adhere to the following criteria:

Customisable - Depending on the parameters decided on in the *Project Preparation* stage of the life-cycle (refer to section 5.3), the proposed framework will dynamically create a mix of quality assurance activities that are ideal for the current project.

Non-Intrusive - Extensive and complicated procedures should be avoided as much as possible. The QA activities would ideally be limited to activities such as going through checklists, checking whether metric readings are within the required thresholds and so on.

Automated or Semi-Automated - In order to speed things up, the QA activities associated with the framework should be as automated as possible. Use of metrics and techniques such as *Code Smells* (M. Fowler, 2000) would speed up the development process without compromising system quality.

5.6 Software Support Tools

As discussed in section 4.6, heavy software tool support is required in order to maintain a high-quality product output in a RAD setting. The nature of these tools will become apparent at a later stage when more work is carried out to concretise this skeletal framework. However one may already venture to propose tools such as automated metric collection tools, projcet management tools, checklist management tools, and so on.

5.7 An Overall View

Figure 2 depicts how elements of the framework will integrate together to produce the required result. It can be observed that the key element is the software development lifecycle which makes use of all of the other elements as needed throughout the whole project timespan.

6 CONCLUSION AND FUTURE WORK

It is believed that this paper has showed a need for a RAD-based framework for e-commerce systems. Initial steps have also been taking towards such a framework by the definition of a number of stated requirements which should be met by the framework and a proposal for a skeletal version of such a framework.

Future work will focus on implementing all aspects of the framework proposed in this paper followed by the carrying out of case-studies whereby the framework could be observed in action so as to hopefuly verify its effectiveness.

REFERENCES

Albrecht, A. (1979). Measuring application development productivity. In *Proceeding of the IBM Application Development Symposium*. IBM.

Arthur, L. (1985). Measuring programmer productivity and software quality. Wiley-Interscience.

Butler, J. (1994). Rapid application development in action. In *Applied Computer Research*.

CNN (2005). 40 million credit cards exposed. In *http://www.cnn.com/2005/BUSINESS/06/18/us.credit.ap/index.html*. CNN.com.

D. Menasc, A. V. (2001). Capacity planning for web services - metrics, models, and methods. Prentica Hall PTR.

E. Cachia, M. M. (2004). Towards appraising online stores. In *Proceedings of the Software Engineering Knowledge Engineering Conference*. Knowledge Systems Institute.

E. Cachia, M. M. (2005). Measuring the functionality of online stores. In *Proceedings of the IEEE Computer Applications Software Applications Conference 2005*. IEEE.

E. Lawrence, e. a. (2000). Internet commerce - digital modes for business. WILEY.

Emarketer.com (2005). Online retail update: Latest q4 quote. In *http://www.emarketer.com/news/article.php?1002631*. Emarketer.com.

Howard, A. (2002). Rapid application development: Rough and dirty or value for money engineering? In *Communications of the ACM*. ACM.

J. Kerr, R. H. (1994). Inside rad. McGraw-Hill.

J.H.Y. Yeung, e. a. (2003). Current progress of e-commerce adoption: Small and medium enterprises in hong kong. In *Communications of the ACM, September 2003*. ACM.

K. Chandra, S. S. C. (2003). A comparison vbscript, javascript and jscript. In *Journal of Computing in Small Colleges*.

Kan, S. (2003). Metrics and models in software quality assurance. Addisson Wesley.

M. Fowler, K. B. (2000). Refactoring: Improving the design of existing code. Addison-Wesley.

Martine, J. (1991). Rapid application development. Mcmillan.

News, S. (2005). Net spending goes up. In *http://www.sky.com/skynews/article/0,,31500-13381218,00.html*.

Pressman, R. (2000). Software engineering - a practitioner's approach. McGraw-Hill.

Small, P. (2000). The entrepreneurial web. Pearson Education.