Developing Web Applications for Small and Medium-sized Enterprises – An Experience Report

Markus Aleksy*, Ralf Gitzel*, Michael Schwind

*Lehrstuhl für Wirtschaftsinformatik III
Universität Mannheim
Schloss
68131 Mannheim
aleksy@wifo3.uni-mannheim.de
gitzel@wifo3.uni-mannheim.de
michael.schwind@web.de

Abstract: In this paper we describe one of the authors' experiences working for an IT service provider in an SME environment. The paper starts with an analysis of the factors typical for the SME business environment and discusses the typical shortcomings resulting from this constellation. Finally, a theory is proposed why the potential of object-oriented methods is not yet realized in the SME environment as perceived in this experience report. While not an empirical study, the paper tries to provide some material for discussions and motivation for specialized approaches for SME software development.

1 Introduction

Small and medium-sized enterprises (SMEs) typically call upon external IT service providers to implement their web presence. This has created a unique market where various SME service providers implement a myriad of different systems which are tailored to the specific needs of vastly different customers. In this paper we present an experience report from this particular environment in order to examine some of the shortcomings of the state of the practice as experienced by one particular developer as opposed to the state of the art as discussed in various scientific publications. While the paper does not contain an empirical study, the various observations found within are intended to instigate discussion and provide a motivation to come up with a solution which in particular addresses a situation as described.

Section 2 of the paper describes the web application market for SMEs as well as the stakeholders involved. Section 3 discusses various problems found in this field. Finally, we conclude with several speculations with regard to why object-oriented and component-based systems have so far failed to fix the problems they were intended to fix in the case of SMEs.

2 Environment

Before looking at the problems endemic for software development in an SME environment, it is important to analyze the typical conditions surrounding the development process. Of particular importance are the various stakeholders, namely customers and service providers, both containing different interest groups. Another factor are the technologies commonly used in this context.

Being SMEs customers often don't have a dedicated IT department but rather employ individuals responsible for a broad range of tasks concerning information technology. The level of expertise of these individuals is generally broad rather than deep, i.e. they are generally unaware of the technical details which are part of the design process. The IT services demanded by SMEs in many cases involve their need to be present on the web, showcasing their products or services. The new software sometimes incorporates a subset of their business processes, however, the SME customer is often unwilling or unable to alter these processes in order to tighten integration with the new software. The ultimate goal usually is the acquisition of new customer groups and the reduction of cost.

In times of economic stagnation service providers supplying small and medium-sized enterprises have to cope with tight budgets and strict deadlines. As a consequence, there is little leeway for any effort not immediately contributing to the fulfillment of a customer's requirements. With meeting budgets and project deadlines being management's main concern, developers opportunistically adapt to this situation by concentrating on these goals as well. Usually only a small number of developers work on a particular project, as the service providers are small or medium-sized themselves. Also, as a result of cost-awareness and a lack of highly qualified staff, lateral hiring is common. Often, these people perceive themselves as "programmers" rather than software engineers, a view which seems to dominate public opinion on what IT projects are all about [De04].

Popular choices for the development of tailor-made information systems for SMEs are scripting languages like PHP or compiled languages like Java. In many cases components like database abstraction layers and template engines are used to support a design that follows a separation of concerns to a certain degree and to save time and money. While content management systems are suitable for the development of systems which are mainly concerned with the presentation of information or entities that fit into some sort of homogeneous structure, they are usually not capable of representing business processes.

3 Common challenges and pitfalls

When developing web-based information systems for SMEs, a number of problems are likely to occur, which in severe cases can even lead to project failure. Failure in this context can mean a number of things. The worst-case scenario is a project that is not finished on time or that exceeds its budget, but these are not the only outcomes that should be considered failures. Failure can also be interpreted as a poorly designed system that is not adaptable to future requirements or causes disproportionately high maintenance costs. While the potential problems described below are not exclusively particular to SMEs and the service providers they employ, they appear to us to be more common in these environments for a number of reasons mentioned below. The following sections describe the shortcomings that can occur during the different phases of development and their consequences with regard to the resulting system.

3.1 Organizational problems

Due to the lack of manpower at small or medium-sized service providers, the developers they employ are forced to accept various responsibilities, some of which can even be conflicting. Examples include analyzing business cases and designing systems while implementing and testing them. Frequently the people that are responsible for specifying and enforcing coding guidelines are the ones these rules are meant for, with the obvious result of the non-existence of such rules.

3.2 Requirements analysis

During the initial phase of a project, communication between the customer and the developer of the system is essential to ensure a common understanding of the requirements the project must meet. Usually customers need to be supported in this process, because they are not fully aware of the complexity of the desired system. This is a result of the lack in-depth knowledge on behalf of the customer's IT manager. With the range of problem domains being very broad, it is often difficult for all-round developers to derive a precise system specification from the common-language information provided by the customer. If the developer fails to fully comprehend the processes that need to be modeled and implemented, severe consequences for later project stages are often inevitable.

Many projects suffer from a poor requirements specification. Experience has shown that often the perceived importance of certain aspects of a system differs from the actual priorities of the customer. As a result the detailed specification of these aspects is deferred to later stages in development. Incompletely defined or constantly changing specifications can lead to a lot of workarounds being built into a system.

3.3 System design

As mentioned before, many developers lack the skills to derive a proper system design from a given specification or they are not able to use the appropriate design methodologies or tools. At the same time, management often fails to see design as a necessity and thus no or little time is spent on considerations concerning the way the system should be built. Especially questions regarding re-usability, separation of concerns and extensibility remain unanswered due to a lack of awareness of the importance of a structured approach to system design. While this doesn't necessarily endanger a projects general success, it can have a negative impact on future revenue by complicating maintenance and causing redundant development of different solutions to similar problems.

3.4 Implementation

Depending on the choice of technology, e.g. the programming language, special attention needs to be paid to aspects like safety, security, and separation of concerns to name a few. Scripting languages like PHP are a popular choice for the implementation of web-based information systems, but their lack of type safety and the option to mix HTML code and business logic offer many opportunities to write code that is counter-intuitive and therefore hard to maintain. The same applies to JavaServer Pages (JSP) as technology for the view layer. Even though there are means of separating view and business logic, it can be tempting to abuse the technology and move more than just display functionality into JSPs, making the resulting pages prone to errors introduced by front end designers. Without properly specified coding guidelines and principles or a lack of responsibility on behalf of developers, there is a high risk that modules developed by different individuals do not interoperate properly and "code smells" [Fo99], i.e. sections of little understood, half-heartedly maintained code, become abundant.

3.5 Maintenance

Maintenance of software systems is a much disdained portion of the lifecycle of software, even though some are of the opinion that it should hold a dominant place in the considerations of developers (cf. [Gl04]). Many maintenance problems originate in the early stages of a software project and have already been hinted at. A major obstacle to maintainability is a lack of separation of system concerns, e.g. mixing view and business logic components.

While documentation is not vital for the initial development of a system, ignoring it can make maintenance that is economically advantageous infeasible, because without it, expensive reverse engineering may be needed for the analysis of existing systems. Because it does not have an immediate value, it is often neglected for what are considered more "pressing issues" or is left in a semi-obsolete state.

4 Conclusion

While object-oriented methods and tools supporting a structured development process are mature technologies, many of their advantages still haven't reached developers working for small or medium-sized service providers. There are a number of potential reasons for this. Even though many developers are aware of the general ideas of object-orientation, they lack applicable skills needed to realize this potential. Therefore, while object-oriented technologies are often utilized, their full potential cannot unfold, because they are used improperly. Besides lack of skill there sometimes is an irrational fear of an increased effort due to object-oriented methods, e.g. it might seem more convenient to neglect basic principles of object-orientation, such as accessing class members directly instead of using accessor methods. Besides, employers are often reluctant to invest in tools supporting the use of these techniques or training, especially in times of recession and when there is no immediate gain to be expected.

In our opinion, further research is required to identify ways of overcoming these obstacles as opposed to refining the existing methodologies and rethink "some of our traditional beliefs in the software field" when they deviate from traditional software engineering [Gl04b]. We hope the experiences described in this paper will motivate scientists, who work in the field of web applications to address the issues specific to small and medium-sized enterprises and will instigate discussion on this subject.

References

[De04] Denning, P. J.: The Field of Programmers Myth. In Communications of the ACM, Vol. 47, No. 7, July 2004

[Gl04] Glass, R. L.: Learning to Distinguish a Solution from a Problem. In IEEE Software, May/June 2004

[Gl04b] Glass, R. L.: Some Heresy Regarding Software Engineering. In IEEE Software, July/August 2004

[Fo99] Fowler, M.: Refactoring – Improving the Design of Existing Code. Addison-Wesley, 2000.