4

Developing Software for School Administration and Management

Incorporating Flexibility

M. Bajec, M. Krisper and R. Rupnik
University of Ljubljana, Faculty of Computer & Information Science, Trzaska 25, 1000 Ljubljana,Slovenia
e-mail: marko.bajec@fri.uni-lj.si

Key words: Software Flexibility, Business Rules Technologies

Abstract: Flexibility is one of the most important characteristics of software systems for computerised school management and administration, in particular if the software has to be used in several institutions. Given that school institutions are diverse in many aspects and have their own specific needs, several problems are faced when developing unified software. In this paper we describe an approach to information system planning and development, which we believe can help to gain the required software flexibility. The main purpose of the paper is not to examine technical issues on information technologies but to emphasise the possibilities that have to be considered when developing software in support of management and administration in schools.

1. INTRODUCTION

Studies on the use of Computerised Information Systems in schools have proved the correlation between the existing variance in the type and extent of School Information System usage and the ability to adapt software to institution-specific needs (Mahnic, 1997; Wild & Fung, 1996). Flexibility is an example of a software characteristic that makes transition to new software smoother and easier. Of course, there are several other indispensable features that can affect the decision whether or not to use or adapt the system. In our opinion, system developers have to be aware of all the requirements in order

to develop adequate systems. Business users, in our case school managers, also have to be aware of the possibilities offered by contemporary technologies and approaches in information system development. A few years ago it would not have been expected that a system would be able to take instructions only by voice, without using a keyboard or mouse – this was simply not possible. New technologies, however, made this dream a reality.

We will use this opportunity, therefore, to introduce a rather unknown approach to information system development. Since our domain is information science, and we have some experience in planning and developing software for school administration and management (Rupnik *et al.*, 1997; Mahnic, 1997), we believe that school management and administration is a good example of an application domain that can fit several modern development concepts. One of these is a Business Rules Approach that primarily focuses on software flexibility issues.

2. WHY FLEXIBILITY?

The need for new approaches and technologies emerged from the problems that we experienced with the development of an information system for higher-education institutions in Slovenia (Rupnik *et al.*, 1997) and later with the system maintenance. The project was funded by the Ministry of Education and Sport and took place at the beginning of 1995. The objective was to develop an information system that supports centralised processing of enrolment applications, which is a common process for all Slovenian universities and independent colleges. As there were many institutions (with many different study programmes), each of which had followed its own enrolment policies and rules, we had a tremendously difficult job developing a rigorous, and at the same time flexible, enrolment policy model. Even today, five years from the launch of the Enrolment System, we still have to make changes required by faculty management.

Similar findings were discovered when developing a students records information system (Mahnic & Vilfan, 1995) for the University of Ljubljana, the largest university in Slovenia. It has 26 member institutions (twenty faculties, three academies and three colleges), more than 40,000 students, over 2,600 teaching and research personnel, and an administrative staff of about 1,250. The purpose of the development project was to support entrance examinations, enrolment, examination records, alumni records, various analyses and statistical surveys. All the applications were written in co-operation between the Faculty of Computer and Information Science and the University Computing Centre, with the support of the European Union

Tempus program (project IEP 1852 "Computerisation of Administration and Management in Higher Education", 1991-94). As Mahnic (1997) states, in Slovenia faculties have substantial autonomy within their universities, and they often have their own policy regarding the use of information technology. Thus the development of an integrated university information system was not only a difficult technical task, but required a substantial organisational effort. Among the other findings discovered through close examination of the experiences of other institutions and initiatives in foreign countries (McDonough, 1992; Powell, 1991; Frackmann, 1991, 1992; Schutte, 1991), was that the system has to offer a certain level of flexibility in order to handle all the differences among the member institutions (in organisation, administration, etc.). Even though some specific solutions required by particular institutions were not a part of an overall agreement, the project team was forced to consider them in order to retain user satisfaction. Again, the question was, how to achieve the required level of flexibility?

2.1 Business Rules and Flexibility

Recently, much effort has been put into developing applications that are flexible and easier to change and adapt. Unfortunately, most of today's applications do not apply to these characteristics, as changes require the modification of low-level program code. Of course, not all changes are equally difficult. If a customer wants to have an additional dialogue box incorporated into the user interface, we can (usually) do that without any substantial effort. But what if, for example, a process of entrance examinations has been changed due to some additional rules that have been put into operation? Or what if a current system allows a student to select four course offerings for the coming semester, but now management would like to allow students to select four course offerings plus two alternative choices, in case the student cannot be assigned to a primary selection? From an organisational point of view, these changes are rather trivial, but in terms of a software change, they are difficult and time consuming. In fact, this is commonly the case: the rules change constantly at a policy level, while we cannot keep up with the software that is used to implement them.

These kinds of rules are known as "Business Rules". Although the name is rather confusing[1], it has become a widely accepted term within information science. Since there is no common definition that would clearly explain the concept of a business rule, various inconsistencies can be noticed

[1] The adjective "business" only causes confusion, as it forces us to think that business rules can apply to the development of business applications only. In fact, they apply to all kinds of applications.

when reading papers on business rules or using tools that claim to support them. For the purposes of this paper, a business rule will represent a statement that defines or constrains some aspect of the organisation's behaviour. Here are some examples:

- A candidate can apply to a maximum of three study programmes at the same or different institutions.
- Each study programme is composed of several courses.
- A date must be specified for an examination and cannot be changed after it has been published.
- A student must register for an exam at least three workdays before the examination date. After that, any registration is rejected.
- If a candidate is not a full-time student, her/his registration to an exam is automatically cancelled, unless she/he has preliminarily paid for the examination.
- If a student has failed an examination more than three times, the board of examiners must be convened. In addition, the student must pay for the examination.

According to various discussions (GUIDE, 1995; Barnes & Kelly, 1997; Ross, 1997; Hurwitz, 1997), business rules have a significant impact on software flexibility and scalability. If not presented properly, for instance if buried in the program code, they can be very difficult to manage and maintain. The most common problems that arise as a result are:

- Every change of business rules requires programming.
- Business rules are distributed across the application logic; thus the place where the change has to be made is hard to find.
- Business rules are dependent and interrelated chunks of logic. Therefore they have to be modified carefully, considering the possible effect on the other rules.
- It is very difficult to control business rules, as there is no common place where they are stored.
- Since the need for changes to rules usually arises from organisation requirements with which developers are not necessarily familiar, there is a risk the requirements will be misunderstood.

These are only a few of the problems that stimulated the development of a new strategy that is primarily focused on business rules. The main idea of this approach is to conceptually, logically and physically separate business rules from the other parts of application, data and functionality, while

making them easy to access, view, modify and manage. Achieving those goals results in improved application flexibility and scalability.

Before discussing the concept of a business rules approach and its associated technologies, we will first examine how the rules are managed within the traditional application development life cycle and traditional applications.

3. TRADITIONAL APPROACHES TO FLEXIBILITY AND ADAPTABILITY

In order to make applications adaptable and flexible, developers have been using several different approaches for a long time. The most common traditional methods include parameterisation and using database mechanisms.

3.1 Parameterisation

One method of adding flexibility and adaptability to an application is to parameterise the application and its components. These parameters may be then set in a configuration file or in a database, and can be managed through a configuration utility. In doing so, the application can be adapted to different environments and situations just by parameter settings, without any programming effort.

The parameterisation technique proves useful when used to provide parameters for the business rules. Although the business rules remain hidden in the application logic, they can be modified through parameters, without any need for changing the program code. In addition, end users can make business rules modifications, if they are provided with simple and user-friendly configuration utilities.

However, this approach presupposes that the development team can foresee all kinds of changes that are likely to be required. Moreover, it presupposes that the development team able to programme and parameterise all additional cases. This requires the parameterisation of all logical decisions (decision logic has to be used or bypassed, and the variables used have to be stored as modifiable parameters). Consequently, this is an extreme burden on the application developer. In addition, application testing suffers from a combinatorial explosion effect that sometimes requires an additional application in order to configure the parameters correctly. These parameters themselves will often be interrelated, requiring assumptions and rules to be encoded in the parameter-modifying application, with similar problems.

http://www.springer.com/978-0-7923-7493-0

Pathways to Institutional Improvement with Information Technology in Educational Management
IFIP TC3/WG3.7 Fourth International Working Conference on Information Technology in Educational Management July 27–31, 2000, Auckland, New Zealand
Nolan, C.J.P.; Fung, A.; Brown, M. (Eds.)
2001, X, 174 p., Hardcover
ISBN: 978-0-7923-7493-0