Entry into Virtual University Space through Web-Based e-Application

Mária Bieliková, Jaroslav Kuruc and Vladimír Marko
Institute of Informatics and Software Engineering
Faculty of Informatics and Information Technologies, Slovak University of Technology
Ilkovičova 3, 842 16 Bratislava, Slovakia
{bielik, kuruc, marko}@fiit.stuba.sk

Abstract

In this paper we describe a web-based information system, which serves for a first contact between a university and a prospecting student (applicant) in the admission process. We describe advantages of an involvement of the applicant into the virtual space of the university. Our e-Application system covers the whole admission process. The objective is to make the communication between the university and the applicant more effective and consequently to achieve considerable time and resource savings at the both parts. The system is designed not as a closed system for the university staff only. It enables an applicant to submit his application electronically and to observe the application's status in the scope of the admission process. The university staff uses the system as an evidence of the applicants, as a support of the applications processing, admission process organization including management of the entrance exam. Due to the current Slovak legislation the applicant has to supplement electronic application by a printed form of the application signed by the applicant (the printed form is generated by the e-Application system). We describe successful deployment of the e-Application system at the Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava to cover 2004/2005 bachelor's admissions and experience gathered through its operation.

1. Introduction

Nowadays, many universities employ information systems whose purpose is to support activities essential for providing the education. Besides e-learning systems, there exist information systems for maintaining students' records, their achievements and necessary decision support related to the educational process. Mentioned support deals with the study period. However, the first contact with a prospecting student is provided before starting the study – during the admission process.

Involvement of an applicant into a virtual space of the university has several advantages. Main benefit lays in effective communication between the university and the applicant. Moreover, the system for support of admissions can serve as an educational tool, which is able to bring prospected students into the university information space and to allow them better understand the field of their prospective study. Automating some tasks would bring also considerable time and resource savings (especially processing applications and admissions evaluation). In order to achieve mentioned facilities we have developed a web-based information system, which covers the whole university admission process.

We have successfully deployed the system at the Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava to cover 2004/2005 admissions for bachelor degree programs. In this paper we describe the admission process based on the usage of an electronic application form (e-Application) together with supporting web-based information system.

The paper is organized as follows. In Section 2 we describe university admission process adapted for an operation within the Internet environment. Next, Section 3 presents developed software system that covers the admission process. We describe its architecture as well as tools and technologies used. In Section 4 we discuss experience gathered through the e-Application system active usage. The paper concludes with summary and discussion related to the benefits of described e-Application system.

2. E-admission Process

E-admission process is based on standard university admission procedure. It is extended in such a way that majority of the admission process steps are accomplished by the medium of Internet. The e-admission process is in full conformance with the Slovak legislation, and our university and faculty study regulations [5]. It consists of the following basic steps:

- applicant registration,
- application submission,
- application processing,
- applicant performance evaluation (including entry exam evaluation),
- admission decision and publishing final results.

Communication between an applicant and the university staff is provided by means of developed e-Application system (for displaying a status of the admission process and up-to-date information) and e-mail service (for alerting a change of the e-Application status).

2.1. Applicant Registration

One of our main goals related to the implementation of electronic admission process is to let an applicant to observe her standings throughout the admission process. Therefore, it is needed to collect applicants' credentials together with additional information, and create her account to offer a personalized view ensuring privacy. The applicant is prompted to enter her basic personal information, as well as a login name, password and contact e-mail address. She can also enter a secret question and an answer to it. The question is used in case she forgets the password. The applicant is also asked to approve a declaration related to the processing applicant's personal information by the university.

The e-mail address is used for activating the account. It is also used later for a notification on significant events within the admission process. The events triggering sending a message include:

- *User registration.* The message contains an URI link, which must be followed to activate the account.
- *Lost password recovery.* The user may request issuing a new password, which is sent to the user's e-mail address. The operation is authorized by the correct answer to a secret question determined during the user registration.
- *e-Application state change.* The user receives a notification when the state change occurs (e.g., a member of administrative staff confirms receiving paper application form, gives a notice of a shortcoming in the e-Application or the final admission results are published).

2.2. Application Submission

E-admission process originates from our former business processes. Therefore the applicant may submit at most one e-Application for degree programs on particular level (bachelor, master or doctoral) operated by particular faculty of the university. He selects a degree program operated by the faculty that he is applying for. More degree programs can be selected together with expressing the preference for each selected degree program. This approach corresponds to our current practice as the administrative staff of each faculty processes applications independently and applications for all degree programs operated by the faculty are processed in the aggregate.

The e-Application form is designed following the paper application form being used in the Slovak Republic presently. We considered well known principles of user interface design [4] and divided the form into several parts: degree programs the applicant is applying for, personal information, educational background, present educational achievements, awards and supplements. Fig. 1 shows the content area of a screen of the e-Application form with the awards tab displayed. Selected rated contests are listed. Other awards can be filled in the edit box bellow.

The application form requires filling quite a lot of data, which can be inconvenient to do at once. The applicant is not required to fill and submit whole form at once. She may return back to the system and fill up semi-finished application as many times as needed (see the *Not submitted* state of an e-Application in Fig. 2).

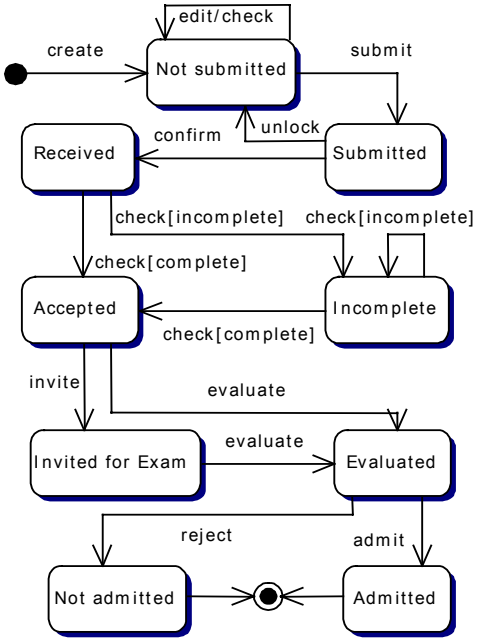

Figure 2. e-Application states

When an applicant completes the e-Application and is prepared to submit it, she may do so at any time before a stated application submission deadline passes. To ensure consistency and completeness of each submitted e-Application it is automatically checked for missing information and/or inconsistencies. The check can be also invoked at any time by the applicant. It serves the applicant in two ways: to make sure that all required information is already entered, or to easily recognize missing application parts (in case of phased e-Application filling).

Once the e-Application is submitted it cannot be modified by the applicant (unless a member of administrative staff unlocks the e-Application; this is advantageous in case when the applicant filled in mistaken data

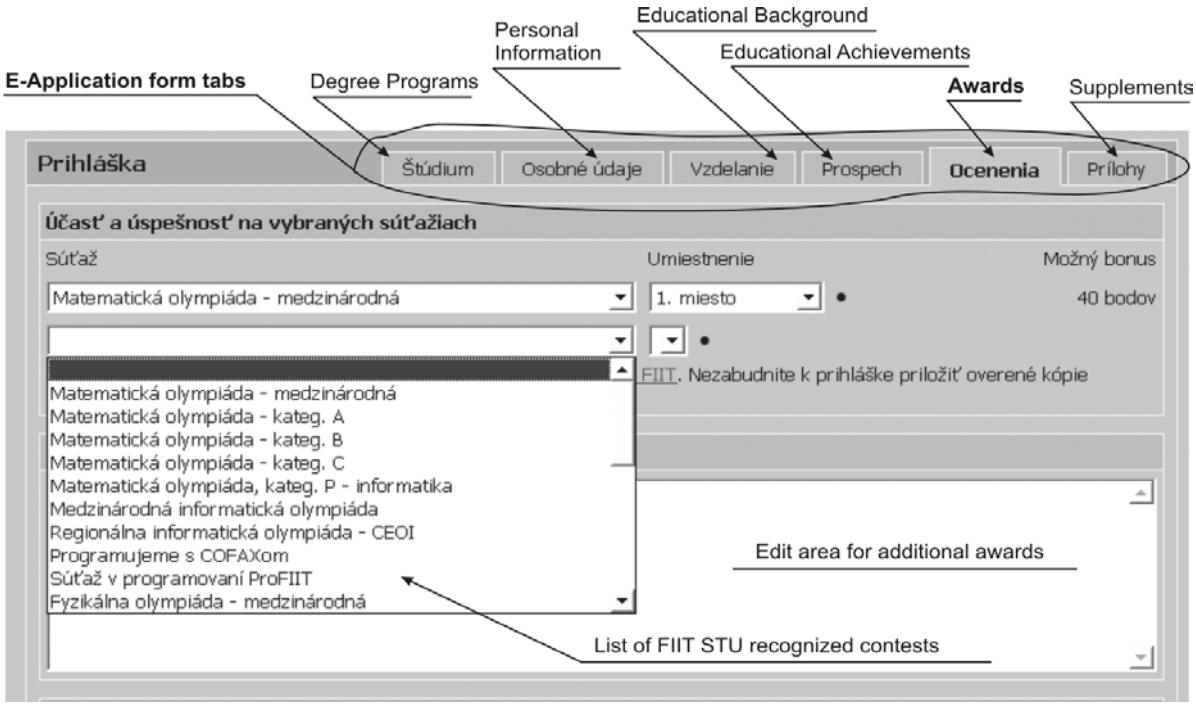

Figure 1. A part of the e-Application system screen (in Slovak)

related to educational achievements and high school staff declines to approve it).

Due to the current legislation each e-Application must be supplemented by delivering a printed form of the application signed by the applicant. Accordingly after a submission of the e-Application the applicant is required to print the application form (generated by the system in pdf format) in order to have educational achievements approved by the high school staff. The paper application form has to be signed and application processing fee must be paid. The applicant should send the paper application form along with required supplements (e.g., educational achievements certificates) to the faculty, which provides degree programs the applicant is applied for.

2.3. Application Processing

After receiving a paper application form the administrative staff confirms in the e-Application system its reception. This step is indicated by an e-mail to the applicant (and represented by a separate state – the *Receive* state, see Fig. 2). The confirmation does not involve checking of the application's completeness. Its main purpose is to provide a feedback to the applicant as quickly as possible.

Incoming applications are queued and checked later (according the workload of the staff). The administrative staff checks both received paper application forms and e-Application forms submitted on-line and marks eventual problems that could not been detected automatically. These include missing mandatory attachments in the paper application, wrong payments, etc. In case of complete application the administrative employee approves the application as accepted (the e-Application enters the *Accepted* state, see Fig. 2). The e-Application system provides a feedback to the applicant by announcing a message with information whether his application has been accepted or there is any problem. He is notified by means of an e-mail message.

The system supports also a submission of only regular paper application forms. E-Applications for the only paper applications are created by the administrative staff. Naturally, in this case the feedback to applicants provided by the e-Application system is not accomplished.

2.4. Evaluation and Results Publishing

Applicants, whose application has been accepted, can be invited to an entrance exam. However, there is also a possibility to permit an applicant not to take admission exam and be evaluated solely based on her history. The applicant selects in her application the exam subjects from the set predefined by the university (or faculty).

To ensure transparent evaluation of tests and the applicants' anonymity to examiners, the e-Application system provides an anonymous identifier for each applicant. The identifiers are printed on answer sheets instead of applicants' names. The e-Application user interface does not provide means for finding out correct pairs – the identifier and corresponding applicant's name.

Admission exam results (points achieved in tests for particular subjects) are entered into the e-Application

system. To eliminate typing errors, results can be entered repeatedly (good practice is to enter results twice by different people). The system automatically highlights conflicting results entered for particular test and the applicant.

The system computes overall assessment for each applicant (taking high school achievements and admission exam results into account). Resulting rankings of applicants constitute principal input for the admission decision process. The results and a final decision are published, so each applicant can see her own results, whether she is admitted or not, and the ranking within the list of applicants. Applicants who have not submitted e-Application may receive during the entry exam access data to generated account enabling them to log into the e-Application system to check the results. The applicants who have provided a mobile phone number are notified also by a SMS message.

3. Computer Support of Admission Process

Developed software system for computer support of university admission process covers the entire admission process as described in previous section. The system is designed not as a closed system for the university administrative staff only. It enables an applicant to submit her university application electronically and to observe the application's status in the scope of the admission process. The university staff uses the system as evidence and monitoring of submitted application forms, and as a support of the admission process organization including the management of entrance exam.

3.1. Requirements and Design Overview

While designing the system we have considered numerous requirements. The key requirements include:

- offering access to prospective applicants coming from different geographic areas,
- considering specific characteristics of high schools environment in our country,
- providing effective communication between an applicant and university staff (effective from the point of acquiring information and requisite university staff workload),
- supporting both applicants and the staff.

The first two requirements imply exploiting Wide Area Networks (WAN) to enable simultaneous client access from all regions of the country as well as from abroad. Our most prospective applicants come from high schools. As high school labs are in many cases the only place where they can access information technology infrastructure (including the Internet), we decided to take an advantage of hardware and software providing by the labs in majority of high schools.

The system's task is not only to collect application forms and publish admission results, but also to allow a communication between the university staff involved in admission process and applicants in the first place. Therefore it employs a front end for applicants as well as an interface for staff to review and manage incoming applications and to resolve eventual issues by collaboration with the applicants.

We have identified following basic user roles:

- *Applicant.* Accesses the system to create and submit an application and to view the application's status and final results of admissions.
- *Member of administrative staff.* Manages the applications processing, entry exam and resolves conflicts.
- *Manager.* Views the e-Applications and statistics (as a base for decision processes).
- *Administrator.* May carry out administrative tasks, such as clearing message queues, or providing for backup.

Fig 3 depicts selected basic use cases executed by users assigned to the *Applicant* role and the *Member of administrative staff* role.

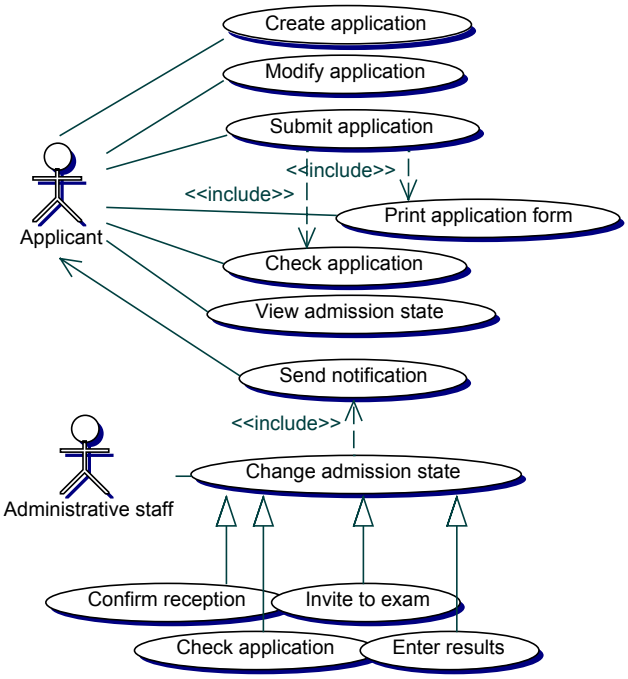

Figure 3. e-Application system use case view

Admission process steps (Section 2) are mapped to corresponding requirements. Basic actions performed by an applicant include creating, editing and submitting an application. The application may be independently checked for inconsistencies or missing data. However, the *Check application* use case is included in the *Submit*

application use case because it is provided at least in the application submission step. The *Submit application* use case includes also printing of the application form. The application form can be also printed anytime after its submission. The applicant can view the admission state including an input by the administrative staff in the form of text notices.

It is in the nature of most actions performed by members of administrative staff that they change the e-Application state. The actions include an application reception confirmation, indication of inconsistent or missing data, inviting applicants to the entrance exam and publishing admission results. Naturally, a member of administrative staff can create and edit e-Applications (corresponding use cases are for the sake of simplicity not shown in the use case diagram on Fig. 3). A change of the application state is announced by sending a notification message to the applicant (the administrative staff can block this feature).

The e-Application system is designed as a client-server web-based application. This is in full conformance with the requirement of the system geographic availability. Applicants' web access brings undoubted advantages. This approach allows applicants to access the system without a need for any special client software installed besides a thin client in the form of a web browser.

The e-Application form constitutes rather complex user interface including input validation, data forms layout on a screen and conditional data items display. Therefore designing usable user interface required significant effort. As the great part of use cases related to the staff is almost identical to the applicants' use cases we decided to share the user interface used by the staff with applicants' user interface. This decision has led to requiring the web-based access also to the staff. It allowed saving resources that would be spent otherwise on creating a thick client application for the staff.

All server components are collocated on a single common server. However, main components may be deployed on different hardware resources. Staff accesses the system by means of a web browser through the faculty LAN and applicants use the Internet as a means for accessing the e-Application system.

3.2. Server Architecture

The e-Application server is designed as an n-tier system. The structure of the system is depicted in Fig. 4. The reasoning behind choosing the n-tier architecture model [1] was to decouple business logic including a domain model from the application logic. This approach not only enforces defining clear interface between the tiers but also enables eventual reuse or extending of existing domain model. Additionally, such separation enables to combine several architectural styles.

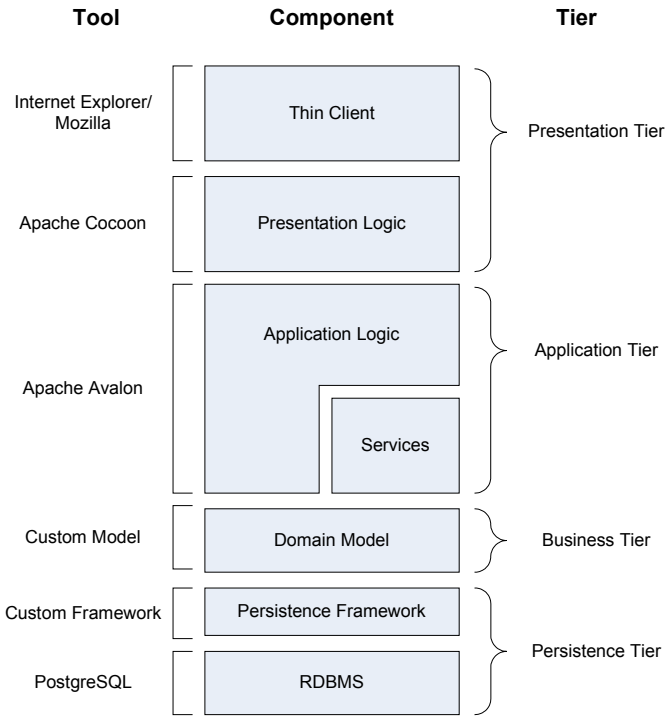

Figure 4. N-tier structure of the system

The persistence tier stores domain model objects. In our case, we have developed a custom persistence framework, which integrates with domain model entities in order to have them persisted by using RDBMS system.

The domain model is complemented by business methods and services, and thus forming the business tier. This platform comprises internal data representation and transformations, which are not yet composed to make up any particular application.

The application tier is built upon business logic and carries out tasks specific for particular application (e.g., executing correct sequence of steps to submit an application form). User's authorization to access particular function or data is also performed at this level. We have considered the option to move the authorization checking to the business logic, but such move would limit reusability of the business logic. Besides crippled reusability some authorization checks must still be executed at the level of the application or even presentation tier, especially for actions that do not necessarily call the business tier in order to complete (e.g., the authorization to access certain static content; authorization checks are also used to display a menu containing only items accessible to a user who is currently logged in). Therefore it turned out that it is justifiable to put the authorization in both application and presentation tiers.

Presentation tier's task is to provide users with human readable form of data through suitable user interface. As mentioned in previous sections, users of all user roles

communicate with the e-Application system using a web-based user interface for existing thin clients.

3.3. Tools and Technologies

Choosing implementation platform was among the most crucial decisions. We have considered variety of options including common scripting languages such as PHP. We finally opted for using Java platform. The rationale behind this decision was among others:

- availability of libraries and components (form support, portable document format generation, internationalization, cryptography),
- existence of proved best practices framework for building web-based applications,
- platform dependability (including dependable application containers and reusable components),
- performance comparable with scripting languages.

Parts of the persistence and presentation tiers, as well as the entire application and business tiers, are written in Java. At the base of the application lies the PostgreSQL RDBMS system, which provides a relational persistent store for domain model objects (business objects).

The domain model is developed as custom structure of Java classes inheriting from common persistence enforcing interface. Classes comprising the domain model should implement functionality required to store and load themselves from persistent store.

Actual storing and loading is controlled by custom built persistence framework, which provides added value functions such as an entity lookup and caching. Actual business object instances are used directly for communication with upper tiers because the server architecture is designed in a way that the business and application tiers are not separated by network boundaries. Consequently objects are passed by a reference between the tiers and no network overhead applies.

The application tier is built upon the Apache Avalon lightweight component framework. It consists partly of actions being invoked from the presentation layer and from services, which run independently of the presentation tier and act on their own. The application tier is decomposed into loosely coupled components. Component framework imposes the Dependency Injection design pattern [3] over the components, and thus the usage relationships are constituted and maintained dynamically and the lifecycle of a component is managed by the framework according to existing usage relations or dependencies between the components. The Avalon framework also provides naming and configuration management services for the application.

The presentation tier is implemented using the Apache Cocoon publishing framework, which provides an XML based platform for transforming data into a form suitable for displaying by a thin client. One of the main advantages of Cocoon is that it introduces the Filters and Pipes architectural pattern [1] to create documents by generating XML event stream from an input document (e.g., file, XML document created on-the-fly), passing the stream through series of transformers (e.g., custom defined modules transforming one XML event stream to another or an XSL transformation sheet), and finally producing final text or binary document using a serializer (e.g., HTML or PDF serializer). The execution of a pipeline may be the HTTP request of certain URI within the application context.

Besides above mentioned components, Cocoon offers many more building block types allowing among other uses also influencing the flow of data and control, effectively forking and aggregating pipelines. Cocoon itself is built on top of the Avalon framework and therefore parts of the presentation layer integrate nicely with the rest of developed system. In particular, we have used following Cocoon features and components:

- Centralized sitemap document defining pipelines within application context.
- Cocoon Forms. XML based forms engine.
- FOP (Formatting Objects Processor). XML-FO based PDF transformer and serializer.
- Cron. A utility for scheduling jobs to be executed automatically sometime in the future.

We found the Cocoon Forms component to be especially useful for representing web forms. It allows defining widgets of the form independently from the final form presentation. However, Cocoon was designed as a web publishing framework and therefore lacks means for efficient controlling the flow between locations within the application (i.e., URIs of pages in the application URI context). There exist efforts to enable comprehensible definition of a page flow represented especially by the FlowScript approach. The Java script server-side language is endorsed by the Cocoon development group, but we found it limited and immature at the time of development, lacking in terms of integration with the rest of application. Moreover, it could not be debugged efficiently. Therefore we decided to go with classic state machine model of controlling page flow. In order to decouple application actions from actual presentation and its sitemap, we have extended Cocoon to include definitions of forwards as possible state transitions triggered by actions' outcome to match capabilities of existing MVC frameworks. In our implementation the definitions of forwards are located directly in the sitemap.

Cocoon itself runs as a servlet within the Tomcat servlet container. Tomcat is considered as an industrial standard in the area. In production system, Tomcat is used in conjunction with the Apache HTTP document server to exploit security features such as SSL.

3.4. Data model

The e-Application system business tier consists of variety of classes representing entities of the domain model and operations upon the domain model objects. The domain model is reflected as a logical data model of persistent RDBMS store. The core of the domain model (excluding operations) is shown in Fig. 5.

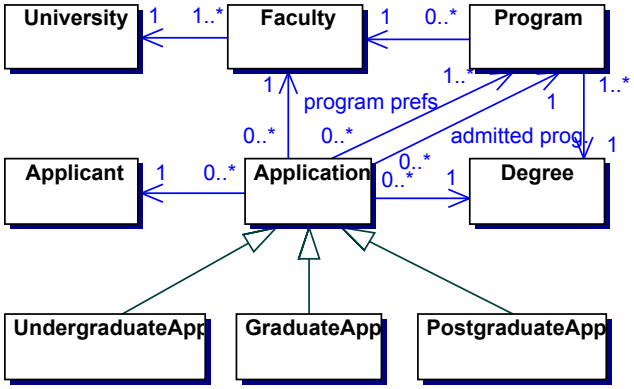

Figure 5. Basic entities of e-Application system

The figure captures only the crucial data entities found in the system. The *Applicant* entity is an abstraction of a person, who applies for usage of the system. It allows to group applications owned by the same person as well as to reuse once entered data in subsequent applications.

It is obvious that an applicant may own several applications to different faculties and different degrees. In our case, we recognize independent application forms for bachelor, master and doctoral degree programs, because we need to collect specific data in each type of the form. Each application contains the preference of selected degree programs and finally may or may not be admitted to one of those.

3.5. External Interfaces

The E-Application software system offers also means for interfacing with external systems. There is a long practice of using the legacy application management system at our university. This legacy system further interfaces with the student management information system by being able to transfer personal data of admitted students. Therefore our priority regarding external interfaces was to support exporting data to mentioned legacy systems (in fact interfacing with the legacy application management system is sufficient). As both systems use dated FoxPro platform we had to develop a specialized application aimed at exporting data from the new e-Application system to the legacy application system (application form data) and vice versa (e.g., list of high schools in our country maintained in the legacy system).

We have also designed the general purpose export interface to export various views of data in the form of Microsoft Excel Workbooks. To accomplish this task we used the HSSF serializer provided with the Cocoon framework. It enables a transformation of the XML encoded application data into the XLS format readable by common spreadsheets. This interface was employed also to export data required to notify applicants of whether they were admitted or not by means of SMS messages. The messages have been sent by custom built application external to the application web server.

4. Experiences

Described e-Application system has been first time deployed to cover 2004/2005 bachelor's admissions at the Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava. The system was developed incrementally in three development cycles that have corresponded to its deployment. The increments included implementation of the following use cases:

1. e-Application form related use cases available for applicants (especially applicant registration, create, modify, check and submit an application, print the application form),
2. e-Application processing related use cases for members of administrative staff (the same as above plus confirm application reception, send notification, entry exam invitation) together with basic statistic displaying for management,
3. evaluation and results publishing use cases important for a completion of the admission process.

Once developed and tested increment has been deployed in accordance with the admission process schedule. Submitting the e-Application online has not been mandatory for applicants in the first year of using the e-Application system. We have allowed also a submission of regular paper forms. Another particularity of our 2004/2005 bachelor's admission process is that applicants were allowed to select one subject for the entrance exam test (physics or informatics), and all applicants with accepted application were invited to the entry exam.

We accepted 478 e-Applications in 2004/2005 admission process (35,75% of all accepted applications). Moreover, there were about one hundred semi-finished applications (few of them even received in the paper form but never submitted in the e-Application system).

We responded to 71 applicants on their e-mail help request. Most common problem was related to providing data about educational achievements. In order to retain the style and content of the paper application presently in common use we provided in the user interface the same

number of rows for high school subjects (24). In many high schools the actual number of subjects is greater then predefined maximum in the application. This does not cause any problem from the admission process point of view because it is recommended to fill in just principal study subjects and those related to the degree study programs applicant is applied for. However, the applicants – high school students – have behaved differently from the prevalent style used in former only paper based admission process. Whereas before the students heavily consulted the application filling with their teachers, in case of the electronic application they consulted the e-Application system and used communication facilities provided.

The other group of problems included forgetting data necessary for the access the e-Application (password, login name or answer to a secret question) and problems related to unexpected behavior of the system in certain versions of MS Internet Explorer.

Our e-Application system was successfully used in its first year of service. However, its operation at the present time heavily depends on presence of system developers. User interface does not cover whole functionality and some operations (mainly related to the evaluation and import/export facilities) require an expert user.

5. Conclusions

In this paper we presented a web-based system for electronic support of university admissions. Various universities provide e-application forms. It becomes a standard practice in many countries to apply for a degree program online. Because existing systems are not open and available to third parties we are not able to compare capabilities of individual systems. We do not have knowledge about the present usage of an electronic application in the Slovak Republic.

A limitation of the e-Admission process used is a necessity to complement e-Application with a paper application. In spite of the fact that we devoted considerable effort in order to simplify the process as much as possible, we cannot fully avoid situation where the paper application form is not consistent with the electronic one (the pdf form of the application is marked by special identifier generated by the system and its security level is augmented by password security with just printing allowed). On the other hand, each application should be supplemented by several paper documents such as educational certificates (or acknowledgment of educational achievements by the high school), award certificates (if applicable), so supplementing also a paper application form does not pose problem. The question is what data are actual and valid: those in the information system or those on the paper application (in case of an overlooked inconsistency). Without a proper authentication of the applicant we should consider as a primary data source the paper application form.

The e-Application system usage proved expected benefits to applicants, members of administrative staff and management. The system helps an applicant to fill in the application form. Help is available to general public and 24 hours a day, 7 days a week and allows the applicant complete access at any time day or night. This help considerable decreases the number of applications with some problems (mainly inconsistency or completeness). The more e-Applications are created by applicants the lesser work is required by members of administrative staff. We can conclude that the admission process with developed application is more effective. Moreover, transparency of the admission process with our system increases. We plan to use the e-Application system in the 2005/06 admissions also for master degree programs.

Acknowledgements

This work has been supported by the Grant Agency of Slovak Republic grant No. VG1/ 0162/03.

References

[1] Buschmann, F., R. Meunier, H. Rohnert, et.al, *Pattern-Oriented Software Architecture*, Volume 1, A System of Patterns, Wiley, 1996.

[2] Clements, P., F. Bachmann, L. Bass, et al., *Documenting Software Architectures: Views and Beyond*, Addison-Wesley, 2002.

[3] Fowler, M., *Inversion of Control Containers and the Dependency Injection Pattern*, MartinFowler.com, Available at http://martinfowler.com/articles/injection.html, Jan 2004.

[4] Schneiderman, B., The Designing the User Interface. Addison Wesley, 3rd Edition, 1998.

[5] STU FIIT, 2004/2005 admission regulations, Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava, 2003, available online at http://www.fiit.stuba.sk.