Notification Systems: Basic Concepts and an Example

Milena Stanković[1], Milan Rajković[2]

Abstract – **In this paper we discuss software systems for user notification, underlying functional and architecture requirements. We also present an example of a notification system, which is developed and deployed at the Faculty of Electronic Engineering in Niš.**

Keywords – **Notification systems, Internet SMS gateways**

I. Introduction

Notification systems are software products used for notifying and alerting users about the events they have subscribed for. Nowadays, they are usually a part of more general information systems, but they can also exist independently if they are used for advertising purposes.

The importance of these systems has been spot by many software vendors. One of them is *Microsoft*, which offers Web Services called *.Net Alerts* that can be integrated into the third party applications. The main disadvantage of these high level solutions for developers is their price and the lack of flexibility, which is why most programmers prefer to implement the whole system from the scratch.

In order to enable efficient alerting, these applications provide multiple alternative ways for sending messages. That is why these systems require using and integrating different technologies. The most frequently used alerting methods are sending E-mail, SMS, ICQ or Microsoft Messenger messages. Of course, users can choose the way they receive the notifications.

In this paper we focus on functional and architecture requirements these systems have to fulfill. In the second part, a notification system called *StudentAlerts* is presented. It is developed and deployed at the Faculty of Electronic Engineering in Niš.

II. Functional requirements

In this section we will present common functional requirements for a notification system.

There are three typical roles that can be identified in the system. The first is *client*, which represents a person who subscribes for a notification(we also use the term *alert*) on a specific *event*. The client can choose from all events that currently exist, and has to specify the notification method (E-mail, SMS etc.). Figure 1 shows a use case for this scenario.

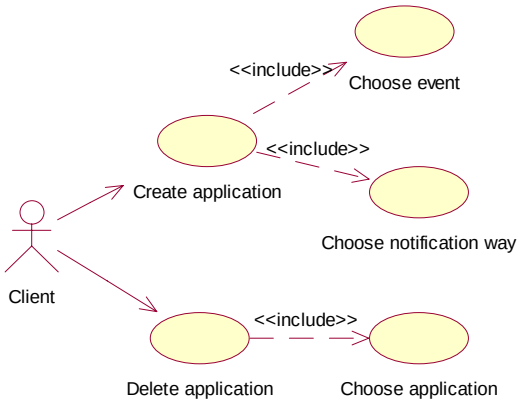

Fig. 1. Use case: subscribing for an event

The second role is *Sender*, which represents a person that creates and sends alerts. There can be many types of alerts, but the most common are personal and group alerts. Latter ones are used for notifying all clients that has subscribed for a specific event. *Sender* also creates new events and delete the old ones. Use case which explains the process of creating and sending alerts is presented in figure 2.

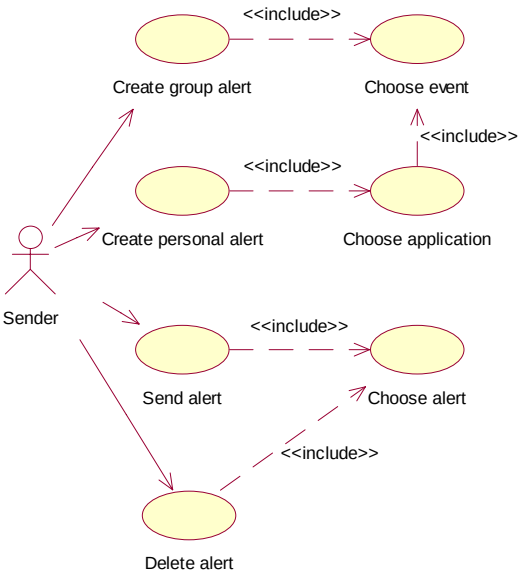

Fig. 2. Use case: creating and sending alerts

The third role is *Administrator*, which represents a person that adds *users* or changes their data. Of course, *users* are *clients* and *senders*. Administrator is also responsible for setting user privileges. Use case which explains the process of managing users is presented in figure 3.

[1] Milena Stanković is with the faculty of Electronic Engineering, Beogradska 14, 18000 Niš, Serbia and Montenegro, E-mail: mstankovic@elfak.ni.ac.yu

[2] Milan Rajković is with the faculty of Electronic Engineering, Beogradska 14, 18000 Niš, Serbia and Montenegro, E-mail: mrajkovic@elfak.ni.ac.yu

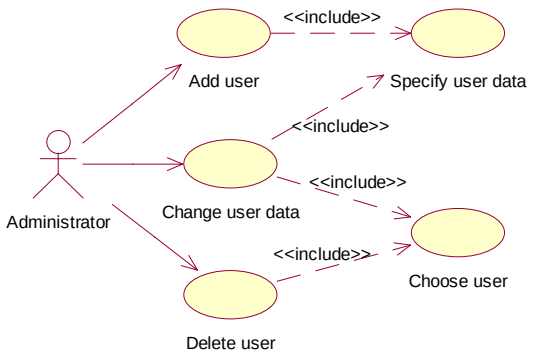

Fig. 3. Use case: managing users

III. Architecture requirements

In this section we will present the most important architecture requirements that notification systems have to carry out.

In order to provide multiple ways for sending alerts, notification systems are implemented as a mixture of different technologies. As we have already mentioned, typical ways for sending messages are SMS, E-mail, ICQ etc. We will here discuss in more details how SMS messages can be sent from a notification application to a client, this being the most complicated and frequently used way.

Basically, there are two different approaches for sending SMS messages from an application. The first one involves using an Internet SMS gateway, which is an interface to an SMS center (SMSC). Many telecommunication companies provide access to their SMSCs via SMPP, SMTP, HTTP or FTP protocols, allowing programmers to include SMS features into their applications. Of course, these companies charge for the service, and the price differs depending on the target mobile network. Figure 4 shows how SMS messages are sent using an Internet SMS gateway. [1]

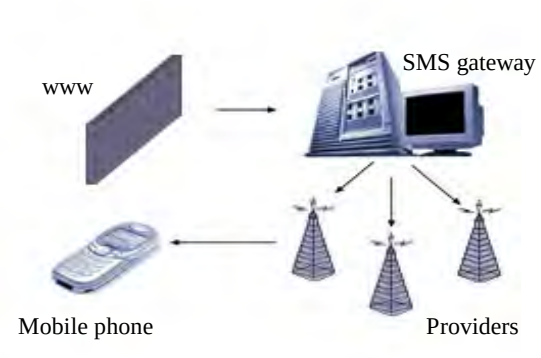

Fig. 4. Sending messages using SMS Gateway

The second approach is a local solution and it involves using a GSM modem or a mobile phone connected to the COM port. There are free software solutions that support many mobile phones, providing a simple, high level interface for programmers. One of these is a server *VisualGSM Lite*, which was developed by company *Visualtron.* [2]

Sending E-mail messages is very simple from most of programming languages. In the next chapter we will show how E-mail messages are sent from a Delphi application. Sending ICQ messages can also be accomplished using ICQ API for developers, which can be found at www.icq.com.

IV. Notification system *StudentAlerts*

In this chapter we will describe an example of a notification system which was developed at the Faculty of Electronic Engineering in Niš. This system called *StudentAlerts* is intended for alerting and notifying students about events they have subscribed for.

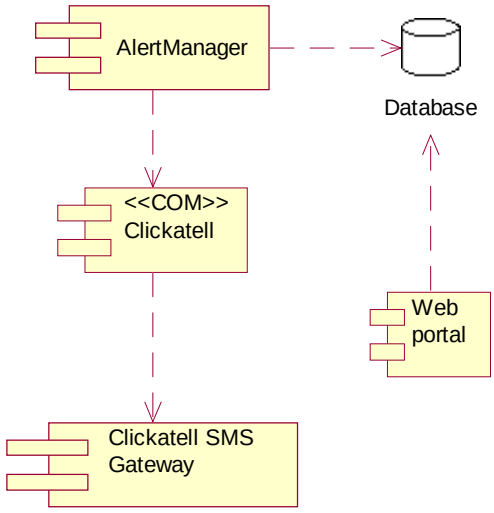

Fig. 5. Basic components of the system *StudentAlerts*

Figure 5 shows the basic components of the system. Students apply for events using Web portal, which is a PHP application. They can change their personal data, like their cell phone number or E-mail address, or they can create new application or delete the old ones. In order to logon to the Web portal, students have usernames and passwords, which can also be modified on the Web site.

All the information important for the application is saved into MySQL database. This is a relational data base, which can be used as a freeware product. Besides, MySQL is very fast and reliable medium for keeping application data. Unfortunately, it has a few problems, like poor user interface and impossibility to automatically maintain the referential integrity. [3]

For setting new events or sending notifications as for maintaining the student data, application *AlertManager* is used. It is created using Delphi7, and has a two-tier , client-server architecture which connects to MySQL database server using *dbExpress* drivers.

Currently, we provide only two ways of alerting: via SMS or E-mail, but we plan to enable alternative ways of notification. For sending E-mails we use *Indy* components, made by company *Nevrona*, which implement SMTP protocol and provide simple interface for developers.

In the system we use *Clickatell*'s SMS gateway [1]. It provides access to its SMS centre using SMPP, SMTP, HTTP, FTP and XML interface. Moreover, programmers of Windows applications can use a COM component that implements HTTP interface and represents a simple and safe way for accessing the SMSC. This option is used in application *AlertManager*, and this is shown in Figure 5.

V. Application *AlertManager*

Application *AlertManager* is a part of the system *StudentAlerts*, and it is used for sending notifications and administration. As we have already mentioned, it is implemented using Delphi 7, with *dbExpress* drivers for accessing *MySql* database via unidirectional cursors. [4]

For sending SMS messages we use *Clickatell* COM component. However, application is designed in a way which enables an easy integration of alternative ways of sending SMS messages (for example, using a different Internet gateway or a local solution like *VisualGSM*). The class diagram which shows how this is accomplished is shown in Figure 6.

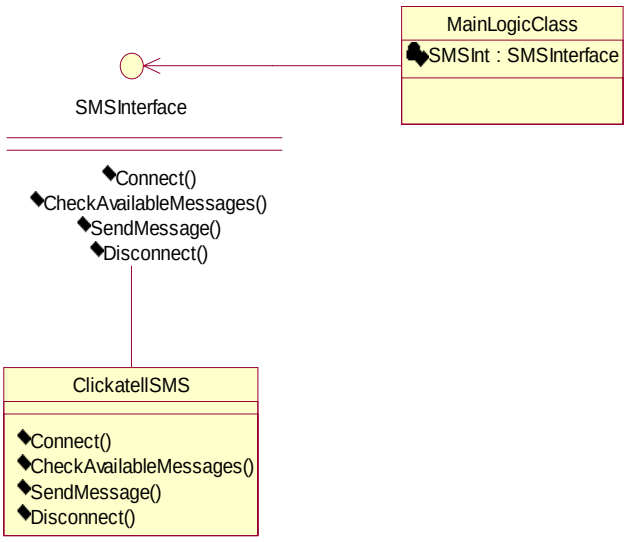

Fig. 6. Classes used for sending SMS messages

Methods for sending SMS messages are grouped into *SMSIneterface*. For every alternative way of sending SMS messages it is necessary to create a separate class that implements *SMSInterface*. Because the client code calls only the interface methods, it is possible to switch between the implementations even during the run time. In this way, a user can choose which provider or gateway he wants to use.

Currently, there is only one way of sending SMS messages, meaning that there is one class that implements *SMSInterface*. Its name is *ClickatellSMS*, and it creates a class *TSMS* which is just a Delphi container for *Clickatell's* COM component.

Application *AlertManager* has very intuitive, *office like* interface and it is very easy to use. It is intended for professors and administrative workers, and its interface varies depending on user privileges.

Figure 7 shows the main application window. On the left side of the main form there is a menu containing categories, while on the right side there are lists showing the elements of the chosen category. Double click on any element of a list opens a detail dialog, which enables changing and saving data. Above every list, there is a toolbar, which is used for adding or deleting the list elements. The toolbar buttons differ depending on the current category.

Categories *Teachers*, *Students* and *Events*, are used for manipulation of data belonging to the respective entities. The most complicated for handling are events and this case is shown on figure 7. Events and student applications are grouped into the master-detail structure. Considering that for one application there can be multiple notifications, applications and notifications are also in the master-detail relation. Toolbar shown on the figure is connected to the currently selected list, although the toolbar actions are a part of a pop-up menu which belongs to the list.

Category *Sent messages* shows all successfully sent messages, where the time of delivery and the codes received from the Clickatell's gateway can be seen in dialog shown on the figure 7. Category *Messages for sending* shows all the messages that have not been sent which are all messages with status *not delivered* or *error on delivery*.

Conclusion

In this paper we presented some basic concepts behind notification systems, and we described common functional and architectural requirements. In the second part, we described a system for student notification, developed at the Faculty of Electronic Engineering in Niš.

It was also shown here that implementation of a notification system involves integration of different technologies. SMS messages are only one example of this. The way this issue is resolved influences the quality of the system itself. That is why it is very important to carefully choose providers or gateways in order to get a reliable application.

References

[1] Clickatell Mobile Solutions, http://www.clickatell.com/brochure/index.php

[2] Visualtron company, http://www.visulatron.com

[3] MySQL Reference Manual, http://www.mysql.com/documentation/manual.php

[4] Marco Cantu, *Mastering Delphi 6*, Sybex, 2001

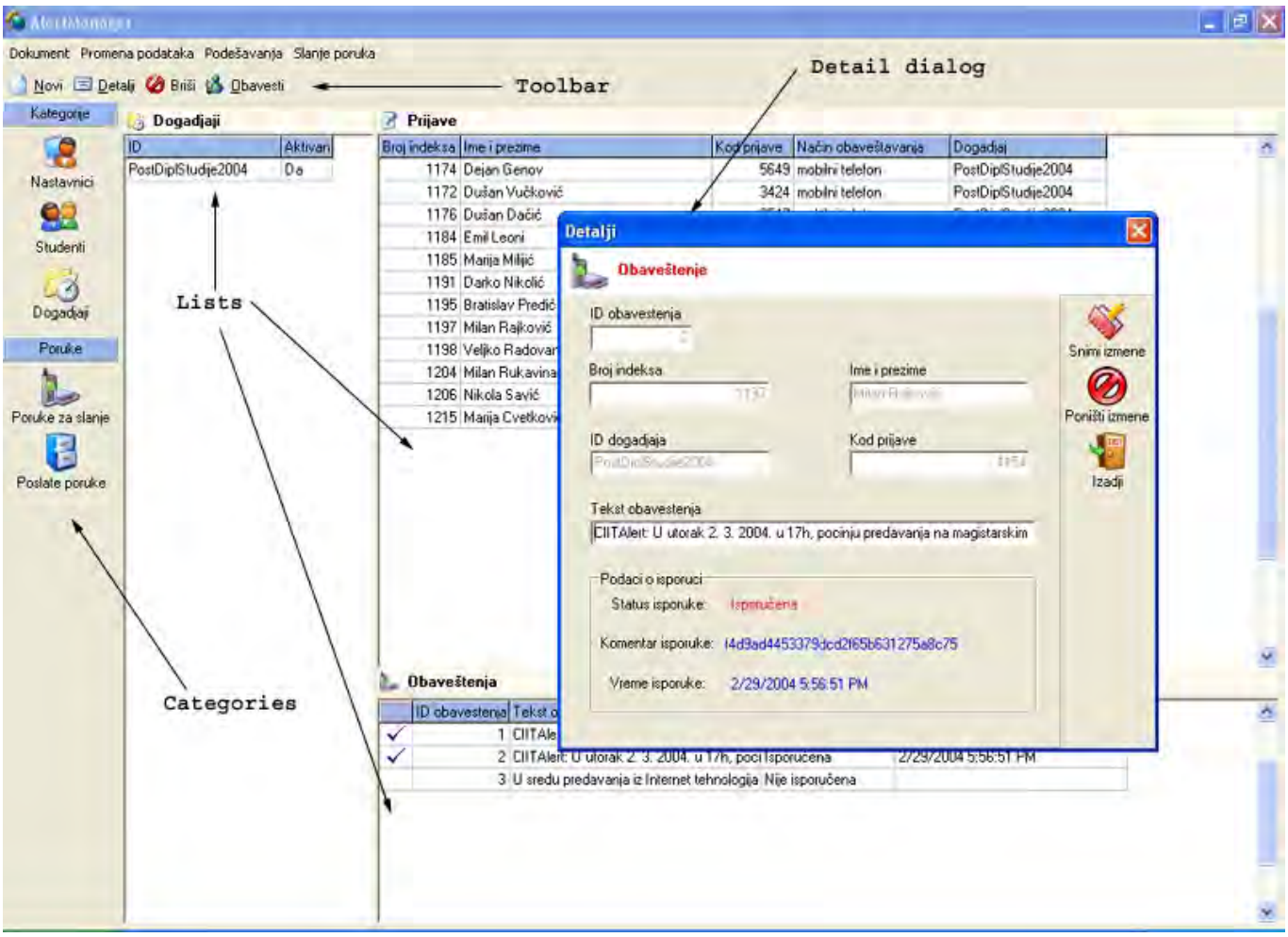

Fig. 7. Application *AlertManager*