

Software for Identifying Secondary Packaging of Medicines

Viliam Mojský¹, Eva Adámiková¹, Mykhailo Dobroselskyi¹, Simona Jaculjaková¹, Pavel Staša²

¹Department of communications, Faculty of Operations and Economics of Transport and Communications, University of Žilina, Univerzitná 1, 010 26 Žilina, Slovakia

²Department of economics and Control Systems, Faculty of Mining and Geology, VSB – Technical University of Ostrava, 70833 Ostrava, Czech Republic

Abstract The article deals with the creation of software for automatic identification of secondary packaging of medicines. In the first part of the analysis we focused on the possibilities of creating software considering available equipment. The second part followed achieved results. Here we focused on the knowledge and software needed to create our solution. Based on the findings of the analysis, we started working on the software. We have tailored the software solution to fit our needs. It has been customized for the laboratory equipment while meeting our requirements. We have successfully created the software and it will be used in further research activities in the ongoing project.

Keywords healthcare, RFID, logistics, automation, transport unit

JEL R49

1. Introduction

The article deals with research aimed at creating a software solution. The created software is designed to help research the possibilities of using RFID tags for automatic identification of secondary packaging of medicines. By secondary packaging of medicines is meant an outer packaging, such as package for one product, for a set of products or a whole transport unit. The software is intended to provide users with features that will serve to further research. Our research is based on the EUREKA Auto-ID Technology and the Internet of Things to Enhance the Quality of Health Services project, which aims to research the possibilities of using AIDC technologies in health services.

In the analysis, we dealt with the current state of the issue, with the equipment and devices in the laboratory and with the requirements on software solution. Then we evaluated the analysis and based on the results we have determined the methodology to be followed. After the analysis and methodology parts, we started developing our software solution. The results of the development are presented in the Results section.

2. Objective and analysis

The aim of our research was to create software that can be used to research the possibilities of labelling secondary packs of medicines with RFID labels. We had several requirements for the software that it had to meet. First of all, it

had to be fully compatible with the laboratory equipment and devices. The second requirement is that it had to have a clear user interface and it should be simple to use. Furthermore, we required the software to:

- Allow data to be written to the memory of RFID tag
- Allow data to be printed on the label
- Provide a database overview
- Retrieve data from labels
- Evaluate the retrieved data
- Graphically represent the printing and reading of labels
- Include database management functions

These are the basic requirements on our software solution. Before we started the development, we did an analysis to confirm the feasibility of our requirements or to adjust them if necessary.

2.1. Legislative regulation

As a part of the legislation on the labelling of medicinal products, a European Commission regulation has been issued to address and regulate it.

It is the COMMISSION DELEGATED REGULATION (EU) 2016/161 of 2 October 2015 supplementing Directive 2001/83/EC of the European parliament and of the Council by laying down detailed rules for the safety features appearing on the packaging of medicinal products for human use. The regulation, inter alia, regulates the labelling of medicinal products, according to which the following indi-

cations will be added to the packaging of medicinal products from February 2019:

- Product number GTIN
- Expiration date
- Batch number
- Serial number

These data will be presented on the package of the medicinal products in human readable form as well as in the GS1 DataMatrix code. They will be used to control the originality and safety of the medicinal product. During production, the specified product data is stored in the European Central Database and in the National Registry. The DataMatrix code is read when a medicine is being sold to the patient and it is verified in the National Registry. After confirming that it is a verified medicine, it will be sold to the patient [1].

The regulation deals with a new way of labelling medicines using the GS1 DataMatrix code, in which the required data is encoded. These data have become additional requirements on our software. Based on the analysis of the European Commission Regulation, we have added to the software requirements the data that are marked as mandatory in the Regulation.

2.2. Laboratory equipment

We created the program in cooperation with the Laboratory of Automatic Identification of Goods and Services administered by GS1 Slovakia. Within the laboratory, there are devices and models designed to simulate the use of automatic identification elements in practice. The laboratory is divided into several sections, where GS1 standards are demonstrated, ranging from 1D and 2D bar codes to radio frequency identification standards.

We decided to create our program in a laboratory section that deals with RFID technology and the associated GS1 standards. The following devices were available within this section:

- RFID Printer ZEBRA 110PAX4
- RFID antennas Motorola AN400
- RFID reader Motorola XR480
- Conveyor belt

The listed devices were linked together through a local network that was connected to the local server. We accessed devices through a computer that was connected to this local network. Figure 1. shows the part of the laboratory in which we worked.



Figure 1. RFID section of laboratory

The devices were arranged in a single model, and all were connected by a conveyor belt. At the beginning, there was the RFID printer with a motion sensor. The sensor captures the incoming object and signals the printer to print the label. When printed, the label is attached to the printer applicator, and the extendable arm applies it to the object passing under the printer. It may be any object that has a flat surface to allow the label to be applied. It must also fit the conveyor belt in dimension and weight. In our case, the test involved boxes and crates that symbolized secondary packs of medicines. The box labelled with the RFID tag is further guided to the RFID gateway that reads the label. We've determined reading as the last step in the program.

We found out from the laboratory analysis that the devices support network communication and that we will be able to use the existing local network. The RFID reader supports the LLRP protocol through which we can communicate with it. RFID printer is ZEBRA brand, so we will have to create print commands in ZPL. ZPL (Zebra Programming Language) is a programming language developed by Zebra for communication with its devices.

2.3. Used GS1 standards

Another topic we had to consider before developing the program was the standards used in program. GS1 standards have recommended use in their characteristics. Based on it, we have set the standards we have implemented in our solution.

We have decided for the GTIN standard because it is intended for labelling goods and transport units. In addition, in combination with a serial number, it creates an SGTIN code that can be written to the RFID tag via the EPC standard. We chose GTIN-13 (EAN-13), from the GTIN code family. We have added these standards to software requirements.

2.3.1. GTIN

GTIN, Global Trade Item Number, can be used to uniquely identify business items at different levels of packaging from individual pieces of goods to entire pallets. It is also possible to add a serial number to the code through which we can identify each piece produced in the production series. Assigning a serial number creates a serialized GTIN, or SGTIN. The GTIN code is unique to every company and none have the same. As a result, when reading the GTIN code in different parts of the Earth, we get a reference to the same manufacturer and to the same product. The GTIN code is divided into 4 types, namely GTIN-8, GTIN-12, GTIN-13 and GTIN-14 [2].

2.3.2. EAN - 13

The EAN-13 code, European Article Number, is one of the GTIN codes (GTIN-13) and it is one of the most commonly used bar codes. It is used for labelling consumer goods, group packages, or transport units. The code is composed of 13 digits, of which the first three digits determine the country of origin, the following nine digits determine the manufacturer and the item reference, the last digit is control digit [2].

2.3.3. EPC

EPC, Electronic Product Code, is a syntax for creating unique identifiers that are assigned to transport units, spaces and locations, business items, and other physical objects that are used in logistics and business chains. The EPC exists in two forms, in the printed form and in the digital form in the memory of RFID tag, which serves as its carrier. It can be expressed in several ways, e.g. in hexadecimal, where it is stored in the RFID tag memory, in decimal for users, or in binary that serves for translation between decimal and hexadecimal, and for writing into the RFID tag memory. EPC Tag Data Standard allows you to write data into RFID tag memory encoded in GS1 identification keys. The GS1 keys that can be written via EPC include, for example, GLN, SGTIN, GRAI, SSCC, ADI, and others [3,4].

3. Methodology

When deciding on how to develop software, we proceeded on the basis of the results of previous analyses, set goals and requirements. In the lab, we had a computer and server on which the program could be stored. Devices in the laboratory support network communication and are already connected. The printer is a ZEBRA brand and requires print commands written in ZPL language. Based on these findings, we decided to create the program as a web application. It will be divided into three parts:

- User Interface
- Middleware
- Database

The user interface serves as a connection between the program and the user. We designed it to provide all the required functionality and good user experience. We created the interface in the form of web application. We used HTML, CSS, PHP, SQL and JavaScript programming languages to create it. To create the front end, we used HTML in combination with CSS. To create the back end logic, we used PHP in combination with SQL. JavaScript was used to improve the dynamics and responsiveness of the interface.

Middleware manages the communication with laboratory devices. It performs data transmission from the database to the devices as well as receiving data from the devices and writing them to the database. For middleware we chose the RFID middleware Aton AMP OnID. It is a modular middleware available to the Department of Communications for teaching and research purposes. Middleware allows you to create schemas by adding modules. These modules are used to create program logic and functions and to link to the database and to other devices.

The database serves as a repository of data inserted by users and middleware. It also serves to send commands from the user to the middleware and thus performs the management function. When creating the database, we decided to use the XAMPP program, which allows creating a local server. Its advantage is the ease of exporting and importing stored databases, which allowed us to easily move the program between computers on which it was developed.

We decided to create the software solution using the spiral software development method. Before development, we have set our self milestones. When we reached them, we analysed the results achieved and determined the next direction of development. We developed all three parts of the program at the same time to ensure their compatibility and to avoid bugs. We developed the components according to the MVVM architecture, ie Model, View and ViewModel. This architecture defines the separation of the data from the user interface and from the logic that links them. It serves to ensure the independence of those parts, so that the change in the internal structure of one does not affect others. Its aim is to reduce the strict interconnection of its parts.

4. Results

Based on the results of the analysis and established methodology, we started to develop software. The following chapters describe the development of each part of our software. The last chapter shows the results of our development.

4.1. User interface development

At the beginning of the development of the user interface, we created a basic layout. Individual parts of the interface fit into this layout. After creating the design, we created a link to the database and the basic logic of the page transition and linking. Then we started developing its content. Based on the requests we've identified from the analysis, we've divided the user interface into six pages.

- Data insertion
- Database view
- Graphic presentation
- Repeat printing
- Data edit
- Control panel

We developed each page individually based on the requirements.

4.1.1. Data insertion

Data insertion serves the user to input data into the database. The inserted data will be used to print the labels and to validate the printing and writing to the memory of RFID tag. We have placed a simple form on the page through which the user can insert data into the database. The form consists of eight fields:

- Product name
- Serial number
- EAN-13
- Company prefix size
- Production date
- Expiry date
- Batch
- Number of items

We have set up the form logic so that the data is checked before it is sent to the database to meet the requirements of the used standards. After verifying, the Serial Number, EAN-13, and the Company prefix size are used to create the EPC code and the EPC data structure of bits. Number of items indicates the number of packages to be labelled. Based on its size, the Serial Number is also changed so that no packages have the same Serial Number. The data generated, along with all form data, is sent to the database where it is stored in three tables: the main table, the buffer, and the control table. After submitting the data to the database, a table appears below the form to display the user's inserted data.

4.1.2. Database view

We added the Database View page to the interface to give the user an overview of inserted data. The page consists of a table and two filters. The table selects data from the database where all the inserted data is stored. Through filters, the user can view data in the database and change the displayed results in the table. We set the table to display only some selected data from the database. When you click on a table row, the app displays the Details window to the user. The Details window displays all the product's data from the database. The details window is shown in Figure 2.

The image shows a 'Details' window with a close button (X) in the top right corner. It contains a table with the following data:

Product name:	Medical product
ID:	1
Production date:	2019-04-22
Expiry date:	2021-04-27
EAN-13:	8580000000009
SN:	1
Batch:	12AA
EPC Tag URI:	urn:epc:tag:sgtin-96:3.8580000.000000.1
SGTIN printed:	48.3.5.8580000.000000.1
EPC data structure:	8.3.3.24.20.38
EPC in hexadecimal:	3076 0BAE 8000 0000 0000 0001

At the bottom right of the window is an orange 'Close' button.

Figure 2. Details window

4.1.3. Graphic presentation

We created the graphic presentation page for the purpose of graphic presentation of the whole process. We split the page into two columns: Print Progress and Encoding check. Both columns have been assigned a custom table in the database. The first column, Print Progress, represents the print queue. We set it up so that it immediately fills with images of boxes as soon as the print order is entered. The number of boxes depends on the number of items entered. Subsequently, after the command is sent to the printer and applied to the box, the box image also changes from unmarked to labelled. We provided this functionality in middleware, where we send feedback to the database, after the label is printed and applied. The second column communicates with the RFID gateway and displays the items that the gateway has read. We set it to compare the read items with the printed ones. If they match, the package is marked with an RFID tag, which means it's okay. If they do not match, that means that the data from the read label does not match the inserted data. The user is notified with a red colour of the box. In both columns, the serial number is displayed on the boxes to identify them. In the first column it is read from the database. In the second column, the serial numbers are decoded from the data on the RFID tags. We have set the page dynamically to show the contents of the database tables in real time.

4.1.4. Repeat printing

The Repeat printing page serves the user to reprint the selected label. We placed a table, filter and a form on the page. The table contains all records of printed labels that can be sorted by filter. Once you have found the desired label, you need to enter its ID in the form and submit it. This initiates printing and the label is printed. We did not set this

feature in our goals, but we chose to implement it as a complementary feature, because we find it useful.

4.1.5. Data edit

Data edit is a page that allows users to edit data in the database. The page contains a table and two filters. Using filters, the user can choose which data is displayed in the table. Data editing is done directly in the table. After you finish editing, you'll need to confirm your changes in the last column of the table. In case of change in the GS1 standard data, the new data is firstly verified to meet the rules set by the standards. Changes will only be made after the changes have been verified and approved. This is also an additional feature.

4.1.6. Control panel

The Control panel page provides the user with basic tools for managing the database, tables and middleware. By using them, the user can abort printing, clear the graphic presentation, or delete the entire database. We have protected the database delete command with a password.

4.2. Middleware scheme development

When creating a control scheme in the middleware, we decided to divide it into three parts: to a part linked to the printer, the part connected to the RFID reader and the control part.

We set up the printer-linked part to perform a regular query on the database in which it determines whether there are data in the buffer table. If the table was empty, the rest of the branch remained inactive. If there were values in the table, the following modules were initialized. Data processing in modules was done in the following order. First, the data was extracted from the database. These were inserted into the print order template. The command was then sent to the printer. Finally, the printed line was deleted from the buffer table. And a feedback was sent on label printing. The print command template was created in ZPL. Inserting data into the template is done through variables. We defined the variables in the middleware and placed them in the appropriate positions in the ZPL template. Figure 3. displays blank ZPL template. The variables are defined by the % character at the beginning and end of the name of variable. Database queries are executed through SQL commands and a database module that provides a database connection.

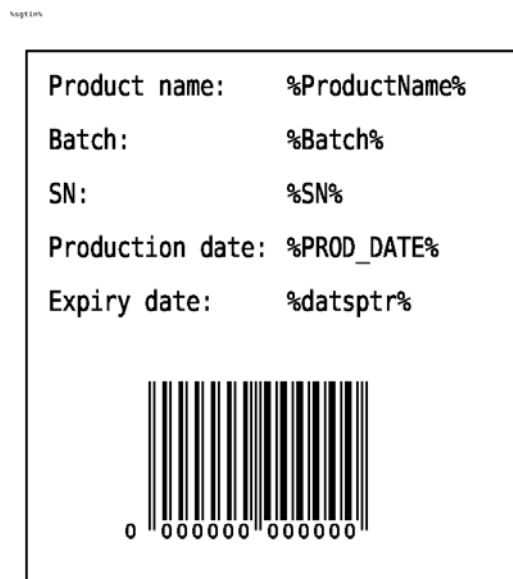


Figure 3. ZPL label layout

The RFID reader section provides communication between the reader and the database. After reading the label, the read data is sent to a programmable JavaScript module. In it we programmed a script that decodes the read data and retrieves the GTIN code and Serial Number from it. The read code, along with the serial number, is sent to the first gateway table in the database.

The control part resets the database and modules in the control scheme. We have configured it that it can be run both from user interface and from the middleware.

The advantage of the used middleware is its modularity. If necessary, we can add more functionality to it without disrupting current features. The entire middleware control scheme is shown in Figure 4.

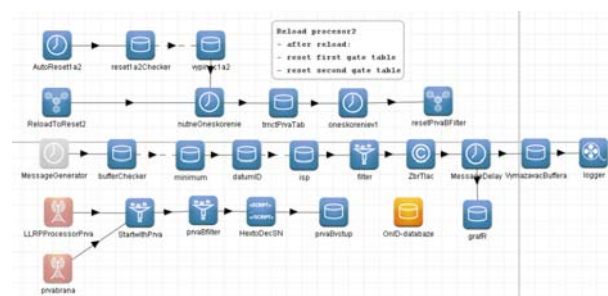


Figure 4. Riadiaca schema v midlveri

4.3. Database development

We developed the database together with other elements of the software solution. Since it communicates with both parts, we had to ensure that it was fully compatible with them. We created the database and all tables in the utf8_unicode_ci set. Five tables were created in the database: the main table, the buffer, the RFID gateway table, the graphical interface table, and the control table.

The main table stores all the data entered into it via the user interface. It serves as a backup of all data and at the

same time provides its data to user interface features. Table buffer serves as a buffer for print queue. When the data is entered into the database, it is filled and gradually emptied as the print runs. The graphical interface table serves as a control and management table for the graphical interface. Initially, inserted data is marked as pending for printing. Once printed, this information is overwritten and marked as printed. Subsequently, they are used for comparison with the read data from the RFID gateway for verification. The gateway RFID table stores data sent to it by the middleware. The control table is used to distribute instructions from the user to middleware.

4.4. Software solution test

All parts together form a complex solution that is designed to provide the required functionality. As a part of the development, we have decided to test the software we created to see if it met all of the requirements.

Before testing, we did a reset of all tables and modules in the middleware. We decided to do the testing by marking the 4 boxes with labels. We chose neutral Product name: Medical product. The serial number was determined to start from 1. We used artificially created EAN-13 code: 8580000000009. The Company prefix size was set to 7. The date of manufacture is always set to the current date. We set the expiry date to 2 years after production. Batch was also artificially created. We set the number of items to 4, according to the number of boxes. With this inputs we started testing our solution.

We started the testing on the Insertion page, where we filled out the form fields according to the specified inputs. The completed form is shown in Figure 5.

Product name	Medical product
SN	1
EAN-13	8580000000009
Company prefix size	7
Production date	23. 04. 2019
Expiry date	23. 04. 2021
Batch	12AA
Number of items	4

Submit
Delete

Figure 5. Filled input form

After submitting the form, the middleware initialized the printer and it started printing labels from the print queue. At the same time as the printing was initialized, the first column in the graphical environment was filled. Four images of

unmarked boxes were added to the column. Gradually, as the labels were printed and applied to real boxes, the image of the box on the page changed from unmarked to labelled. A label and a serial number was added to the image of the box. After labelling, the box was guided by a conveyor belt to the RFID gate. The gate read labels on the boxes and the data was sent to the database by middleware. The data were then selected from the table and compared in the second column of the graphical interface. If the loaded data matched the sent data, an RFID tag image was added to the top of the box. If the data did not match, the box was coloured red. As a part of the testing, we also placed a box from previous tests on the conveyor belt. The box had different data encoded in the label. The goal was to test the verification of read data. Both columns with the test data are shown in Figure 6.

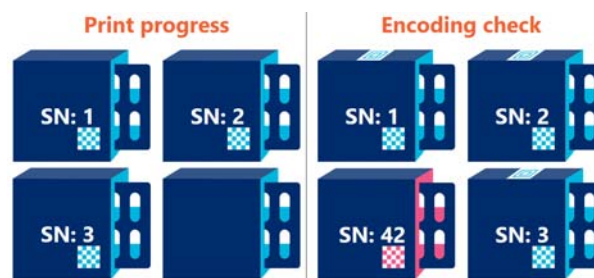


Figure 6. Results of testing

Reading and verification of the read data was the last step in the graphical environment. We also decided to test other user interface pages where we tested the filtering, editing and reprinting of data in the database. All performed tests were successful.

Based on the tests we have performed, we evaluate that the software solution we have developed met our goals and requirements.

5. Conclusions

Our research has succeeded in creating a program that meets the objectives and conditions. The created program has a simple and understandable user interface. The parts of the program are independent of each other, which we have achieved using the MVVM software development architecture. Using this program, it is possible to encode data into labels. After the labels have been printed, the program verifies data writing using RFID reader. In addition to printing and reading, the program also has additional features. It allows the user to view the data in the database, edit the stored data, perform reprinting and it also provides the user with some database management tools.

The results of our research can be used to research the possibilities of labelling secondary packaging of medicines using RFID labels. The advantage of the created software solution is its modularity. Additional functionality can be added to the program without disrupting its current features, which is useful characteristic in the case of further research.

ACKNOWLEDGEMENTS

This article was supported by Project EUREKA-E! 1118 U – health – Auto-ID technology and the Internet of Things to enhance the quality of health services.



This article was supported by Project 3/KS/2019 – IV – Návrh metodiky implementace RFID technologie pro specifické požadavky podniku.

- [2] Tengler, J., Kolarovszki, P., Kolarovszka, Z., 2017, Identification and Localization of Transport Units for Selected Company., In: Proceedings of the 16th International Scientific Conference Reliability and Statistics in Transportation and Communication (RELSTAT-2016), Procedia Engineering, pp. 491 – 500, 2016.
- [3] Kolarovszki, P., Tengler, J., 2015, Practical research in field of automatic identification in automotive., In: CLC 2015 – Carpathian logistics congress – conference proceedings, Tanger Ltd, pp. 92 – 97, 2015.
- [4] Kolarovszki, P., Vaculik, J., 2014, Middleware – Software Support in Items Identification by using the UHF RFID Technology. In: Mobile and Ubiquitous Systems: Computing, Networking and Services, Lecture Notes of the Institute for Computer Sciences Social Informatics and Telecommunications Engineering, pp. 358 – 369

REFERENCES

- [1] Online Available: <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0161>