The Basic Process for the Implementation of Virtual Simulators into the Private Security Industry Using a Support Tool for Automated SQL Script Generation

PETR SVOBODA, JAKUB RAK, DUŠAN VIČAR, MICHAELA ZELENÁ
Tomas Bata University in Zlín
Nám. T. G. Masaryka 5555, 760 01
CZECH REPUBLIC
psvoboda@utb.cz, jrak@utb.cz, dvicar@utb.cz, m_zelena@utb.cz

Abstract: - This article is focused on the simplified implementation of virtual simulators into the private security industry. The first part of the article presents the basic points of the problem being solved. The second part describes an analyst for the private security industry needed to facilitate the implementation process. The third part is dedicated to the specifications of the requirements for the actual implementation. The fourth part describes a basic algorithm for the implementation of virtual simulators for the needs of the private security industry. Finally, the functionality of the proposed tool for facilitation of the implementation with the automated SQL script generation is presented.

Key-Words: - Excel, implementation, private security industry, software engineering, SQL, virtual simulation.

1 Introduction

The Virtual simulators are useful tools for training members of the armed forces all around the world. The simulator enables participants to acquire knowledge and skills in a virtual environment; it also allows a large variety of scenarios while the risk associated with training is minimized. The training of employees of the private security industry is currently being conducted in a standard way that can be referred to as live simulation. There exist virtual military simulators such as Virtual Battlespace 3 but they need to be redesigned and customized to enable training by means of up to date virtual simulators. It means that requirements for the redesign and customisation will have to be specified to software developers. [1, 2, 3]

A document entitled the Software Requirements Specifications, also known as the Requirement Document, is the output from the analysis of customers' requirements. This document can be characterized as an official summary of requirements that the software developers should implement. This analysis is usually done by an IT specialist – an analyst working as a SW developer.

The previously mentioned specifications can be divided as follows:

1. User requirements – natural language phrases supplemented with diagrams that describe services expected from the system and limitations under which it must work.
2. System requirements – a detailed description of the functions, services and operational limitations of the software system. The document containing the system requirements (sometimes called the functional specification) should exactly define what is to be implemented. It can be included into a contract between the customer and software developer. [3]

For both types of requirements, there are two forms of the specification and these are:

1. Specifications in natural language – the output is almost an unlimited and unstructured text that can be comprehensive, intuitive and universal, as well as vague and unclear.
2. Structured specifications – expressing requirements while maintaining comprehensibility, structure, uniformity and abilities to express. Templates are usually used for this approach. Also, the structured specifications are often supplemented with tables and models depicting, for example, the functionality algorithm of the requested software and the relationships of its individual objects. [3, 4]

2 Analyst for the private security industry

Above all, the algorithm proposed below is based on adjusting the existing analysis of data

requirements. It is precisely the analysis of data requirements that has been identified as an activity the processing of which can largely be transferred from the developers to the customers, which means directly to the employees in the private security industry (PSI). For these purposes, the term "PSI analyst" is used in this article to designate a person from the PSI field who performs the comprehensive processing of software requirements. Fig. 1 indicates the task of the PSI analyst when processing software requirements processing software requirements.

The approach described above has the following advantages:

- Accelerating the process of specifying software requirements for adjusting the simulator by a person experienced in PSI problems.
- Cost savings owing to the use of own resources instead of resources of developers. [3]

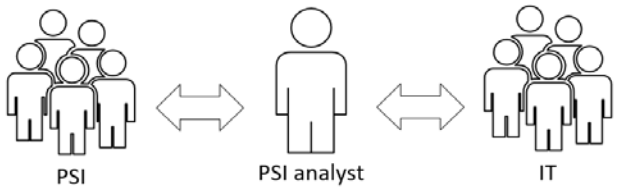

Fig. 1. PSI task analyst.

In this case the analytical activity is carried out by the customer, who has in his or her ranks a person whose work tasks include analytical activities. In compliance with specific procedures, such a person is capable of creating the Requirement Document for specific implementations provided he or she has experience in the field of the private commercial industry and information technologies. The objective of the procedure proposed below is to help customers to develop a comprehensive, sufficiently detailed and comprehensible Requirement Document for the subsequent implementation into the selected simulator. [3]

3 Specifications of requirements related to the implementation

The purpose of the procedure proposed below is to define the content and creation of the Requirement Document for the following areas:

1. Implementing new scenarios for training and adjusting the existing ones.
2. Additional adjustment of existing simulators including the addition of new objects, their attributes and relations, or editing the existing objects.
3. Implementing new actions that can be performed by the selected types of objects. [3]

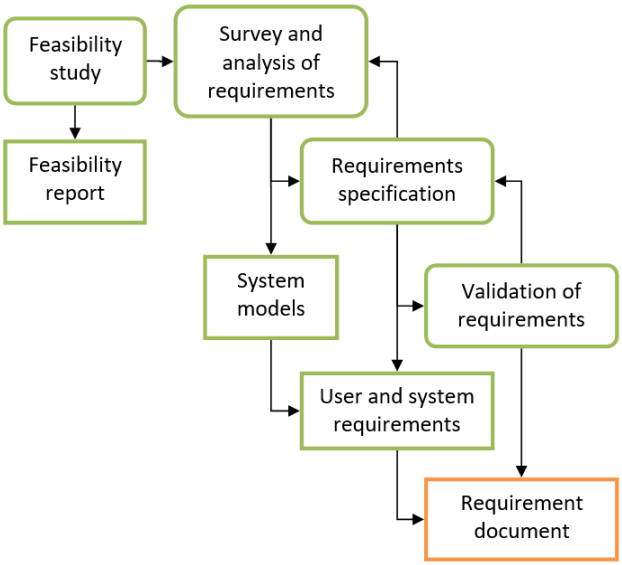

Fig. 2. Scheme of requirements of the engineering process in the PSI [6]

Focusing on the proposed algorithm, it would be possible to base this on methods similar to those used in the agile software development, especially the principle of incremental development, i.e. the developing the system in phases. By means of gradual implementations, the chosen simulator is being improved and completed to the point where it can be used for most of the scenarios intended for training in the private security industry.

The engagement of the previously mentioned PSI analyst enables a change in approach towards requirements engineering. Fig. 2 depicts the process of requirements engineering during which individual sub processes are usually processed by an IT analyst. In the figure the processes in the province of the PSI analyst within the PSI are bordered in green while those under the direction of both the PSI and developers are bordered in orange. Processes preceding the creation of the Software Requirements Specifications Document are without exception in the hands of the PSI analyst, and both parties then process the actual document. The result of the process carried out by the PSI analyst is the first version, which is then handed to developers for consultation. The PSI analyst then incorporates any

observations and this process is repeated until the document is completed. [3]

4 The basic algorithm for implementing virtual simulators to the private security industry

The he basic algorithm for implementing virtual simulators to the private security industry is depicted in Fig. 3.

The described algorithm is based on the PSI needs and its aim is to perform training in a virtual simulator. Prepared training scenarios are received by the PSI analyst who checks the completeness of the tool. If the simulator lacks some of the features important for training by means of these training scenarios, the Software Requirements Specifications Document with specifications of the individual implementations is prepared. Consequently, the document is taken over by the IT sector and the requirements are implemented into the simulator. Upon the completion of the tool for training using the said scenario, the actual training in the simulator is performed.

In the extended algorithm (Fig. 4), the PSI analyst proactively searches for new requirements for implementation based on training scenarios. When a new requirement is detected, it is thoroughly analysed, validated and specified at two levels – the level of user requirements and the level of system requirements and system models. Upon finalisation of these phases the PSI analyst compiles the complete version of the Requirements Specifications Document and submits it to developers (IT). They revise the document and in the case of any deficiencies it is returned to the PSI analyst who again incorporates the changes in the field of user and system requirements and system models. After incorporating the changes, the PSI analyst submits a new version of the document for a repeated revision. When the document is complete it is implemented into the actual simulator; failing that, the process of incorporating new requirements is repeated. [3]

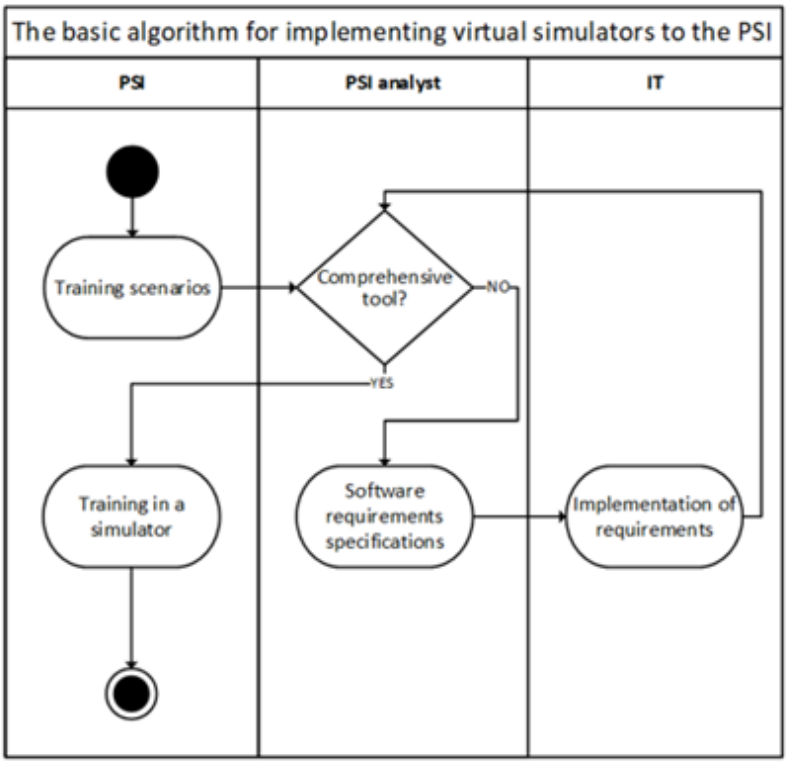

Fig. 3 The basic algorithm for implementing virtual simulators to the PSI [3]

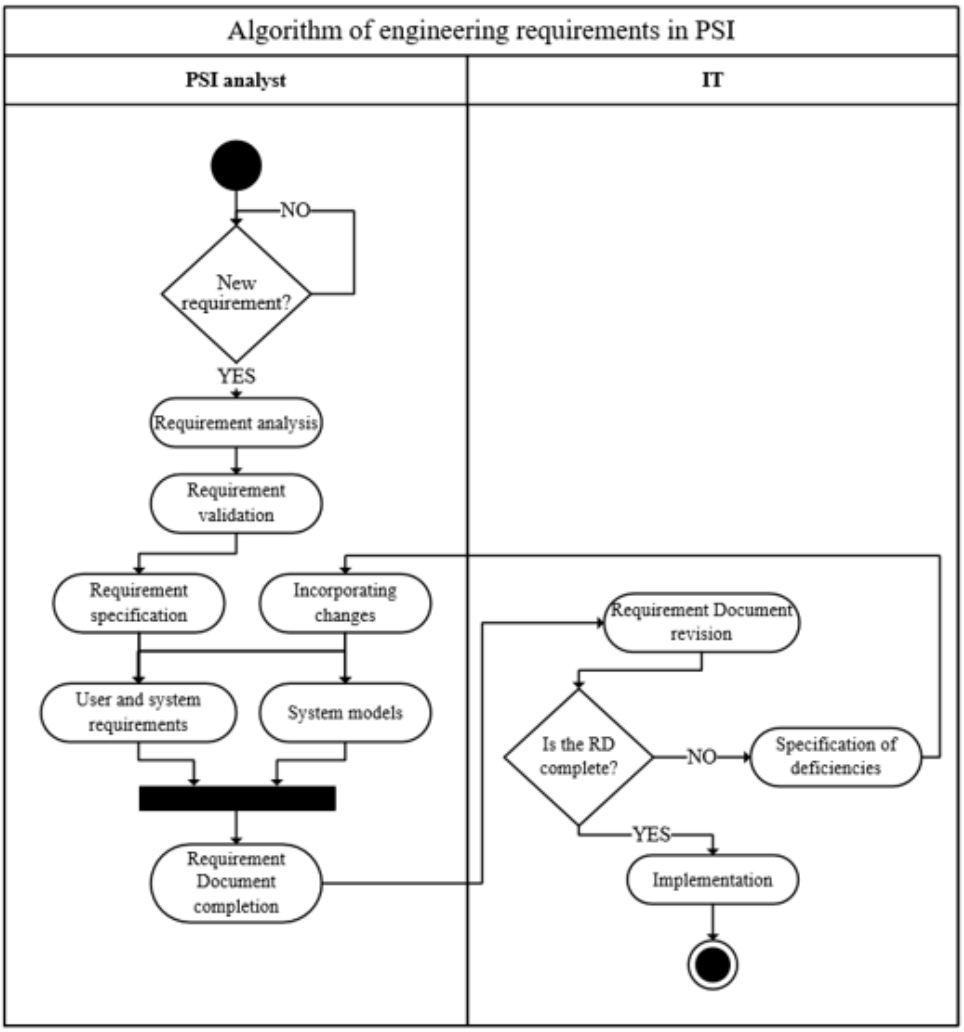

Fig. 4 Algorithm of engineering requirements in PSI

For the purpose of reviewing and completing the Requirements Specifications Documents, version control is proposed. It means that the version number is included in the title of the document in the following form:

document_title_v#.extension,

where:

Document_title is the title of the document without diacritical marks,

v# is an abbreviation of the word "version" and # is the document version number,

extension is the standard extension of the document.

For instance:

requirements_document_v1.docx

It is important to note that multiple revisions can be performed. The documents processed by the PSI analyst are designated by odd numbers while those reviewed by IT specialists are designated by even numbers.

In The final document without any additional revisions is the resulting Requirement Document, which is ready for the actual implementation. [3]

Due to the focus on various possible types of implementation, these were divided into 3 categories according to the type of implemented objects:

- Implementation of the object type – i.e. implementation of the missing type of objects into the given simulator, which is needed for the future simulation in the PSI.
- Implementation of the scenario – i.e. implementation of any scenario for training of PSI employees.

- Implementation of the action – i.e. implementation of the missing action, which can be performed by object types in the simulator.

These three types of implementation are closely related. In the event of the scenario implementation, it is often necessary to implement a new type of object because it has been detected by research of the current situation that many significant objects for the private security industry are missing in current simulators.

4 Support tool for the script generation

Outputs from the designed tool are also included in the Requirement Document. Based on the internal structure of the simulator, for which the implementations are created, these outputs generate the appropriate source code to facilitate the implementation process. The general principle of the tool functionality is shown in Fig. 5. [3]

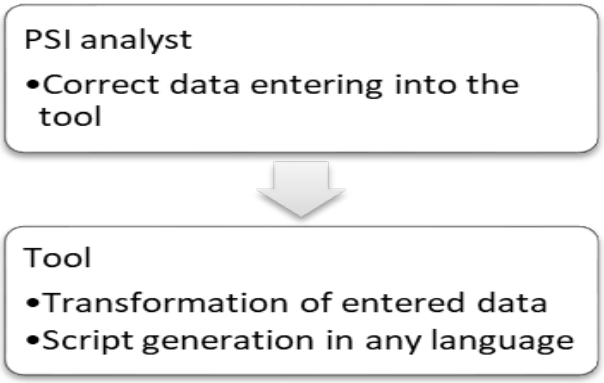

Fig. 5 Activities of the PSI analyst when working with a tool for facilitation of implementation

Within this research, a tool has been designed for each type of implementation the output of which is a source code that affects the database structure of the example simulator. The tool was designed in accordance with the objectives of the dissertation and then with respect to:

- Easy additional adjustments for specific needs.
- User-friendliness.
- Low dependence on the platform used.

For this purpose a tool was designed in MS Office Excel, which is available in the Czech Republic in the most commonly used Microsoft Windows platforms and it is also available for Mac OS and Mac OS X. The tool can also be run in LibreOffice – a free option to the paid MS Office, which also runs under the Linux operating system.

By filling designated cells, this tool generates outputs in the form of executable SQL scripts in accordance with Fig. 5. Although the described tool is used to generate SQL scripts, it can be easily customized, which means that the generated outputs can be modified.

In order to generate SQL scripts the database structure is divided into three categories according to the purpose, namely codebooks, tables and junction tables, see Fig. 6; the affiliation to the category is clearly stated in the title as follows:

"category_title",

where:

- *category* – is expressed as "cis" for codebooks, "tb" for tables and "cn" for junction tables.
- *title* – is aptly selected title expressing focus of the table.

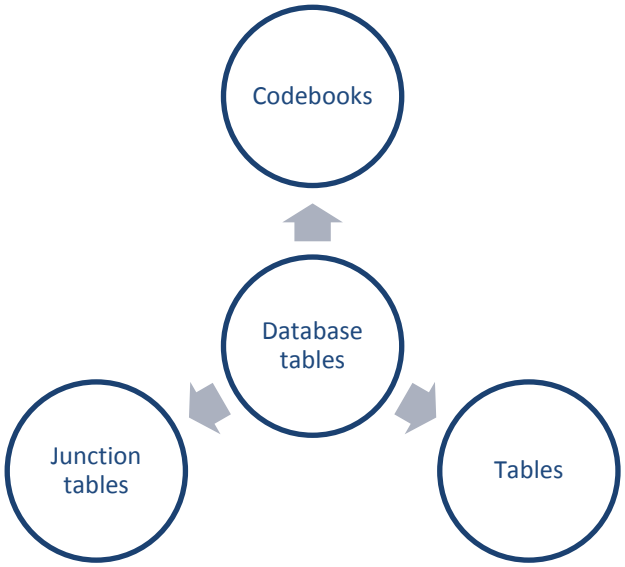

Fig. 6: Division of database tables

The actual outputs are automatically generated scripts, which allow maintaining correct syntax and applicable forms; in the cases mentioned bellow, the text in the generated output indicating commands is written in capital letters, and other text is standard.

3.1 Requirements for the object type implementation

Tool X1_object_types is used to define attributes of object types. The focus of this tool is based on the

fact that attributes of object types are the elemental constructs of data models, which must be specified when objects are to be added. Each object attribute must be defined prior to saving it in the database using the title and its data type. The tool allows:

- Adding specific attributes for individual categories of object types.
- Assigning specific attributes to individual object types of the category.
- Adding default attributes for all categories of object types.
- Assigning default attributes to individual object types of all categories.

On Fig. 7 the dependence of X1 tool on X0 tool is shown.

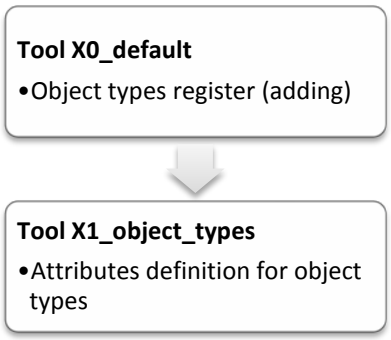

Fig. 7: Dependence of X1_object_types tool for the implementation of the object type on X0_ default tool

Requirements for the implementation of a scenario

The tool designed to facilitate the implementation of a new scenario enables:

- Selecting a number of occurrences for individual object types in the scenario.
- Assigning attributes to individual objects.
- Setting the default (random) or predefined values of these attributes.

The tool is based on data from X0_default and X1_object_types tools see Fig. 8. Once the tool has been run, it loads objects types from the default tool X0_default.

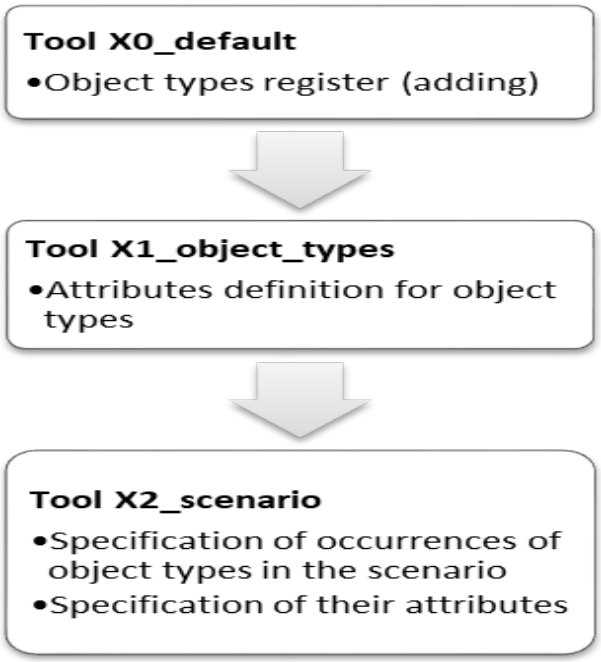

Fig. 8: Dependence of tool X2_scenario for the scenario implementation

Requirements for the implementation of a new action

The tool designed to facilitate the implementation of a new action enables:

- Adding a new action.
- Assigning actions to individual object types.

The process for using the tool designed to implement an action is initiated by loading the object types from the default tool X0_default into the tool X3_action of the cn_action_objects sheet. On the same sheet, categories of individual objects are loaded from the tool X0_default depending on the titles of the category sheets see Fig. 9.

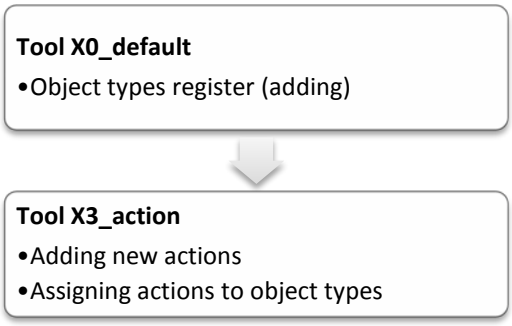

Fig. 9: Dependence of the tool X3_action for the implementation of the action

4 Conclusion

The research presented introduces the basic process of implementing virtual simulators into the private security industry. The newly created position of the PSI analyst facilitates the implementation of virtual simulators in the PSI and it simplifies the process of specifying requirements. This enables the use of simulators for the training of employees in the private security industry. In addition, a support tool for SQL script generation, which facilitates individual implementations, is presented in the article. Future research will be focused on detailed clarifications of individual tools and a way of generating scripts, together with the methodology for compiling the requirement documents of each individual implementation.

Acknowledgements. This paper is supported by the Internal Grant Agency at Tomas Bata University in Zlín, projects No. IGA/FLKR/2017/003; and No IGA/FLKŘ/2018/001.

References:

[1] SVOBODA, Petr a ŠEVČÍK, Jiří. *The Use of Simulation in Education of Security Technologies, Systems and Management.* In: Proceedings of the 2014 International Conference on Applied Mathematics, Computational Science & Engineering (ACMSE 2014), Varna, Bulgaria, 2014. ISBN 978-1-61804-246-0.

[2] P. SVOBODA, L. LUKAS , J . RAK , D. VICAR. *The Virtual Training of Hazardous Substances Transportation*. Proceedings of 19th International Scientific Conference. Transport Means. 2015. Kaunas 2015. ISSN 1822-296X (print), ISSN 2351-7034 (online).

[3] P.SVOBODA, J. RAK, D. VICAR, M. ZELENA. *The Basic Process for the Implementation of Virtual Simulators into the Private Security Industry Using a Support Tool for Automated SQL Script Generation.* In: Applied Physics, System Science and Computers III: Proceedings of the 3nd International Conference on Applied Physics, System Science and Computers (APSAC2018), September 25-28, 2018, Dubrovnik, Croatia, 2018. Springer. ISSN 978-3-319-75605-9.

[4] SOMMERVILLE, Ian. *Software engineering*. Tenth edition. Boston: Pearson, 2016. ISBN 978-0133943030.

[5] ISO/IEC/IEEE 29148:2011(E). *Systems and software engineering — Life cycle processes — Requirements engineering*. First edition. Piscataway: IEEE, 2011.

[6] PRESSMAN, Roger a Bruce MAXIM. *Software engineering: a practitioner's approach*. Eighth edition. New York: McGraw-Hill Education, 2015. ISBN 978-1-259-25315-7.