CEZARY ORŁOWSKI
TOMASZ SITEK
Gdańsk University of Technology
KAMIL DOWGIELEWICZ
WŁODZIMIERZ WYSOCKI
Koszalin University of Technology
TOMASZ DERĘGOWSKI
Acxiom Global Service Center Polska

# THE STRUCTURE OF KNOWLEDGE RESOURCES SUPPORTING A MODEL OF IT TECHNOLOGIES ARCHITECTURE

Summary

*This paper introduces an abstract model for tools used in the software engineering process. The main goal of the research is to develop a universal model which allows a description of tools used in the process. The model supports a decision-making process and gives an option to choose the best tools for the situation. The model considers many factors including the company's structure, project environment and characteristics as well as available features. The model is supposed to be a support for decision-makers responsible for selecting an adequate solution for software engineering projects. Therefore, it is planned to build a knowledge-based system which could store in a formal way all knowledge regarding this problem area and also would help in its acquisition and processing.*

**Keywords:** software engineering, knowledge management, decision support, IBM Jazz

## 1. Introduction

The reality of today's IT market poses high demands on technologies for software development. On one hand, these technologies should support software development processes such as gathering requirements, measuring progress, quality management, or code production. On the other hand, they should support the management processes of production processes and of the project team, such as team development, preparing schedules and allocating tasks. Each of these processes requires a dedicated technology tailored to the needs of the given project, but also to the specific company implementing the project. This means that dedicated technologies have to meet the needs of a diverse group of producers and potential users. These teams consist, inter alia, of developers, testers, managers, analysts, and finally, customer representatives. More and more often, these teams are dispersed, often in different time zones, a fact that should also be included in the functionalities of the technology dedicated to the needs of teams and projects. Due to the fact that the structure of teams which carry out projects and the structure of projects are both complex, the selection of appropriate technology is such an important task.

The purpose of this paper is to present a method of matching technologies to the needs of teams and projects. Such a method should allow the choice of technologies by their initial description and comparison. The comparison should be made in terms of the requirements of the production organization looking for software suitable to their needs. In view of the wide range of technologies used, this paper suggests limiting their scope to support two selected processes: providing quality and managing projects as well as demonstrating their functionality. The presentation of the functionality of two representative technologies: Rational Quality Manager and Rational Team Concert, is intended to build reference models for these technologies. Reference models are crucial in the development of Open Source technologies, for which they are intended. Therefore, the presentation of both technologies, their features and a method of matching the functionalities of these Open Source technologies to the needs of companies, using a rule-based model, is the main content of this paper. Thus, both technologies will be presented briefly at the beginning, while the bulk of this work will be devoted to the description of a rule-based model for matching the functionalities of these technologies to the needs of production teams.

## 2. Support of the software development process – an analysis of the current status

Both technologies which will be presented for the construction of the reference model are based on the IBM Jazz technology platform. This platform enables the integration of technologies supporting software development into a coherent whole. The platform consists of five independent technologies [5] [6]:

- Quality Manager (QM) – dedicated to quality management, provides services primarily related to software testing,
- Rational Team Concert (RTC) – providing project management services,
- Functional Tester – allows the operation of automated tests recorded with the use of scripts,
- AppScan Tester – performing automated testing of web applications in terms of security,
- Requirements Composer – providing services related to the management of requirements.

Figure 1 shows the structure of the jazz.net platform and its division into technologies that can be accessed as services (the Jazz platform is based on SOA) [2]. The diagram presented in Figure 1 shows the functionalities of the Quality Manager technology. The diagram was prepared with the use of an analytical model called the Service-Oriented Modeling Framework [1]. According to SOMF, a service is a coherent set of functionalities.

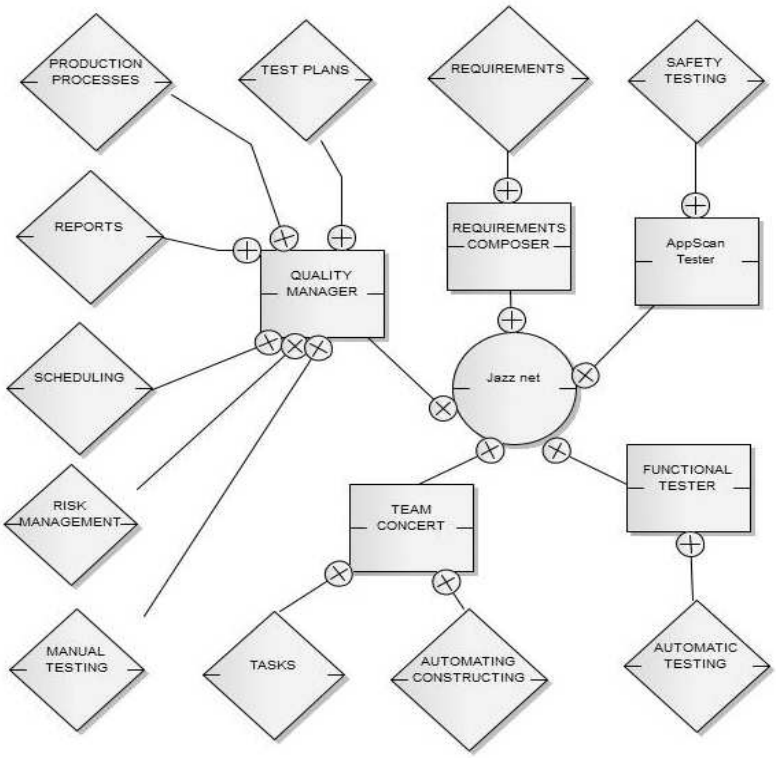


*Figure 1. Breakdown of a standard jazz.net platform*

Source: [2].

All other technologies available on the Jazz platform can be presented in the form of similar diagrams. Let us assume that A is a model set $A = \{x1, x2, ..., xn\}$, where $x1, ..., xn$ are the different features provided by the IBM Jazz platform, such as the ability to create a test plan, defining roles and users, and graphical reports. The assumption is that the technologies available in the Jazz platform, containing all the possible features which may prove useful in software production, constitute a complete set. We also assume that set B defines a group of functionalities of one of the other applications available on the market (e.g. Open Source) and is a subset of A. Therefore, for these sets the following will be true:

We also assume that this theorem is true for all Open Source applications available on the

$$B \subset A$$

market. Thus, if IBM Jazz provides all the possible functionalities, why might the decisions of the project manager focus on other technologies?

Using the optimal solution (in terms of the number of functionalities), which the authors claim the IBM Jazz platform is, may not always be possible. It is, commercially, a relatively expensive solution, and thus often impossible to implement in smaller firms. For such companies the cost of purchasing appropriate licenses appears to be an insurmountable barrier. In such cases, many of them look for cheaper technologies, often Open Source, which are not as ideal as the IBM technologies, but may prove to be sufficiently matched to specific applications. Often they may not be complete technologies. There may also be a need to use and integrate several different technologies.

It is assumed that the IBM Jazz platform is a complex, comprehensive solution which supports all possible aspects of the software development process. An attempt to model all its functionalities exceeds the scope of this paper. Such a model should be developed in isolation according to the needs of specific users: it should contain all those functionalities which could be potentially used, without evaluating whether they are useful and to what extent. A set of requirements prepared by the user will determine the usefulness of the given functionalities. In the requirements, the user will specify which functionalities of a given system are relevant to them, in terms of a particular case of use, a particular company or with a particular project.

Therefore, the model developed by the authors will include information about many technologies stored in the format specified by the model. Then, on the basis of requirements set by the user, it will compare the different technologies in terms of those functionalities which are valued by a particular client. In this way, it will indicate the technology or its functionalities which are closest to the client's needs. It is assumed that user requirements can be divided into two main groups:

- defining which organization processes are to be supported by the desired functionalities / a set of functionalities of a technology,
- defining which atomic functionalities the individual functionalities are to comprise of and how they are to support the organization processes.

Figure 2 presents a model which was based on the above requirements and describes the functionalities contained in the RTC and QM.

The individual RTC and QM functionalities which are relevant to the organization processes were marked in grey – software testing and project management. Although these technologies provide support for all organization processes, particular clients may be interested in selected functionalities of these technologies to support only some chosen organization processes. Each process is broken into atomic features (in the figure, they are the elements with a white background). Thus, for example, for the Work item tracking stage, the following atomic functionalities can distinguish [4]:

- track work item,
- assign work item to resource,
- set work item severity, type and status,
- automatically set creation date,
- add tag to work item,
- import work items from:
  - other bug tracking systems,

  - CSV file,
  - Excel file,
- create work items templates,
- search work item using queries,
- search for duplicates,
- track work item change using RSS.

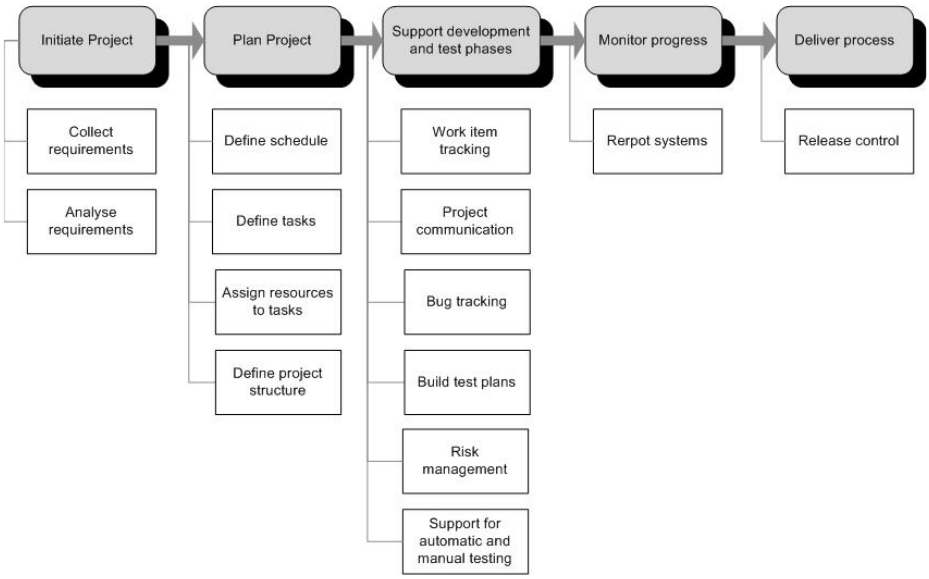


*Figure 2. A model of RTC and QM functionalities*

In addition to the client's requirements that refer to functionalities / atomic functionalities, the model also defines conditions existing in the organization which refer to software production, such as the number of developers, testers and analysts, and their knowledge – software and production methods which they are familiar with. Modules and applications currently purchased and supporting the production process are also defined.

Therefore, the model of matching technologies and their functionalities should include not only demands in terms of the production processes (of used technologies), but also the state of the organization. It should also include full knowledge about each of the tested technologies, as well as information about with which other technologies and their functionalities they can be integrated.

## 3. Rule-based model for selecting IT technologies

After presenting IT technologies and their functionalities, this chapter presents a rule-based model for selecting these technologies (RBMS-IT). Its purpose is to identify appropriate technologies to support the software development process based on criteria defined by the user.

The intention of the authors is to create a model which allows for the provision of decision support services. The decision-maker – usually the project manager – is faced with answering a number of key questions. These questions – temporarily called competence questions – can be formulated in different ways. Answers to such questions may, for example:

- allow the user to select a suitable alternative from the available pool of software – considering their key criterion (e.g. time, cost, functionality),
- compare two alternatives without implying which one is superior,
- indicate specific values (e.g. the cost of implementing a given application, the number of functionalities) in response to specific questions.

This can be achieved by using the concept of expert systems, namely intelligent mechanisms based on formal knowledge. The RBMS-IT model is based on the assumptions of this class of systems. It, therefore, allows the implementation of the following processes:

- collection and storage of knowledge about information technologies supporting software development and project management,
- inference about the selection of technologies on the basis of an evaluation of the relevance of the functionality of these technologies for the project.

The general concept of RBMS-IT is shown in Figure 3.

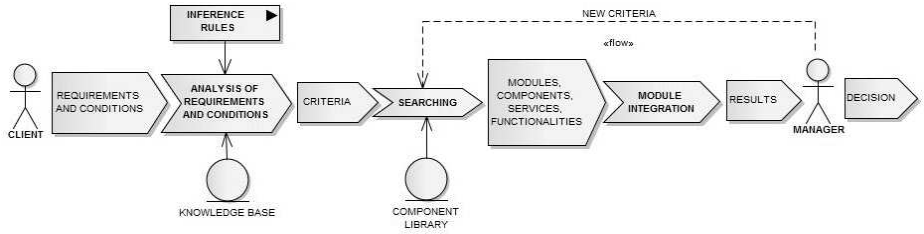


*Figure 3. The concept of an IT technology selection model*

The general principles of the operation of the RBMS-IT model are identical to the assumptions of the relevant expert systems. Therefore, a component approach applies here. Two key components constitute the model and they are described in turn as: knowledge resources and the inference mechanisms independent of them. They are presented in the following sections of this work.

### 3.1. Knowledge representation in the RBMS-IT model

Knowledge will be accumulated in knowledge bases. These knowledge bases should contain information about all the tools available on the market with a detailed description of which elements of the software development process are supported by those tools and to what extent. The formal knowledge records will be rules and facts.

Facts will be written in the form of so-called OAV triples, being a combination of Object, Attribute, Value. Their aim will be to store information about the functionalities and the atomic functionalities of individual applications. The *Work item tracking* functionality, quoted earlier, could be a component of the following facts:

- indicating which application has this functionality built-in

```
<rational_quality_manager, has_functionality,
work_item_tracking>
```

- showing which atomic functionalities a given functionality has:

```
<work_item_tracking, has_atomic_functionality, track_workitem>

<work_item_tracking, has_atomic_functionality,
assign_workitem_to_resource>
<work_item_tracking, has_atomic_functionality,
add_tag_to_workitem>
```

The source of facts in the proposed solution will primarily be our own research. In the course of the project, the authors are planning to examine available versions of RTC and QM as carefully as possible. On the basis of these activities, the first (known as "zero") provision of knowledge bases will be carried out. This should be understood as the identification, naming and defining of all the elements / functionalities of the model tools used in the process of software development. Having the most complete knowledge base will allow customer requirements to be more accurately matching to available solutions.

The "zero" provision is only the beginning of the process associated with the acquisition of knowledge. The RBMS-IT knowledge base is assumed to require continuous updating. Of course, it is difficult to mention all the elements of the production process, as it is known that they are not permanent and may change during the production of a new software development tool or during the creation of a new methodology for software development.

The second method of recording knowledge in RBMS-IT model bases is using rules. Rules will be written in the form of Horn clauses or implications. The rules will be used to present the relationship between specific products and desired functionalities [3].

```
suggested_application (IBM Rational_Software_Analyzer):-
    required_prevention_method (static_analysis),
  company_size (medium).

suggested_application (KnowledgePlan):-
```

```
    required_prevention_method (automated_quality
  predictions),
    company_size (big).

suggested_application (KnowledgePlan):-
    required_prevention_method (automated_quality
  predictions),
    company_size (medium).
```

### 3.2. Knowledge processing in the RBMS-IT model

The knowledge gathered in the model will be processed through inference processes. A question posed by the user will become the inference trigger. Depending on the type of question the inference engine does not work always carry out the task in the same way. The inference methods may be different depending on the problem.

It is planned that the RBMS-IT model should include several so-called evaluation schemes. Evaluation schemes are understood here as specific knowledge processing algorithms indicating the selection of the appropriate part in the given conditions. At the same time, they define the use of the inference engine. These evaluation schemes can be viewed as meta-knowledge, or knowledge referring to the use of resources gathered on the basis of the experience of selected experts. For a given decision problem, the RBMS-IT model can use multiple evaluation schemes leading to the same kind of response.

The methods which can decide about an individual model stem directly from the specific knowledge which is available. This refers primarily to its quality, namely the potential problems associated with obtaining and further processing the knowledge. Two main evaluation schemes taken into account at this stage are:

- the fuzzy model (with varying degrees of complexity: a simple model, self-organizing, self-tuning) – its use justifies lack of precision on the part of experts,
- inference with the use of degrees of certainty – when it appears that some wording supplied by experts cannot be taken to be certain; an expert may have doubts about the quality of their knowledge.

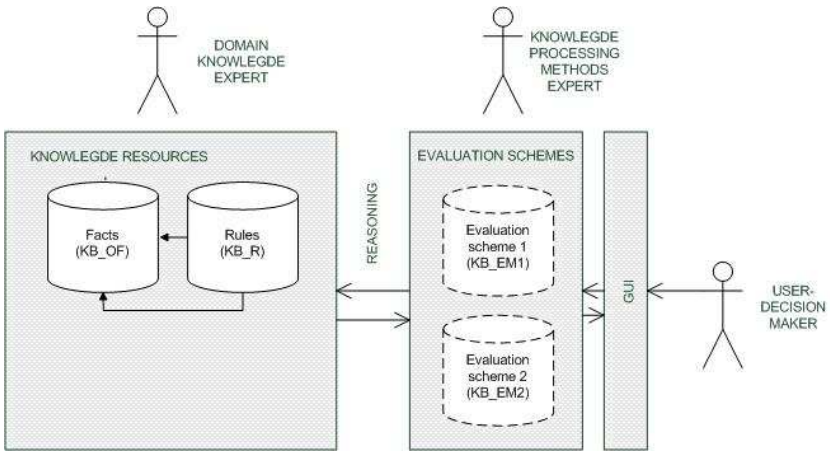


*Figure 4. The IT Tool Selection System as an evaluation scheme for the Knowledge Acquisition and Processing Model*

At this stage of research, it is not yet fully clear what kind of knowledge will be available from experts. Hence, it is not reasonable to analyze the functioning aspects of the inference engine in more detail

In the conclusion of this chapter, another aspect of this project is worthy of attention – dividing the experts. As shown in Fig. 4, it is necessary to distinguish two groups of experts:

- Domain Experts – their job is to make their knowledge and experience with IT projects and software selection available,
- Experts in formal methods of knowledge processing – this group does not need to have knowledge of a particular field; they constitute support for formal and technical aspects of knowledge acquisition and inference.

## 4. Conclusion

The paper presents the concept of a rule-based model for selecting information technologies. This concept is illustrated with the example of the selection of technologies supporting the software development process. The technologies supporting production have been extracted from the model system for supporting production. The authors have adopted a family of IBM Rational tools as a model. The selection of technologies is made with the help of rules of inference and knowledge stored in knowledge bases. The model supports the selection of appropriate Open Source tools to meet a client's needs. The choice of tools is a key stage on which the success of the project depends. The tools are selected at the beginning of the production process. The choice depends on the production methodology, the system architecture, and the team engaged in the production. It is difficult to change tools during the production process. The transfer of collected data to a new tool is usually impossible.

The model is presented in its initial phase – the idea and overall concept. The development of the model will be directed towards its implementation. The next steps in technology selection will

be specified and implemented. With reference to Figure 2, first the inference rules will be developed, as well as the knowledge base, relating to the selection of the necessary functionality tools. The following stage of the model's implementation will be to develop a knowledge base on existing Open Source tools and opportunities for integrating them. Then methods of acquiring tools and their integration will be developed.

The model will be tested by applying it to the selection of tools for certain sample projects. The projects will be selected from real projects both from the field of industry and academia, and from examples constructed to meet these needs. The sets of tools selected by RBMS-IT will be compared with sets chosen by project leaders.

**5. Literature**

[1] Bell, M. Introduction to Service-Oriented Modeling. Service-Oriented Modeling: Service Analysis, Design, and Architecture, Wiley & Sons, New Jersey, USA, 2008.
[2] *Jazz.net website* (2012, November 11) [Online]. Available at www.jazz.net.
[3] Jones, C. and Bonsignour, O. *The Economics of Software Quality*. Addison-Wesley 2011.
[4] Krishna S., *IBM Rational Team Concert 2 Essentials*, Packt Publishing, Birmingham, UK, 2011.
[5] Myers, G. J. *The Art of Software Testing*, 2nd ed. John Wiley & Sons Inc., New York, 2004.
[6] Robertson, S. and Robertson, J. *Mastering The Requirements Process*, Pearson Education Inc. 2006.

## STRUKTURY ZASOBÓW WIEDZY WSPIERAJACYCH MODEL ARCHITEKTURY TECHNOLOGII IT

Streszczenie

*W artykule przedstawiono koncepcję modelu dla narzędzi wykorzystywanych w procesie inżynierii oprogramowania. Celem badań jest opracowanie uniwersalnego modelu, który umożliwi opis narzędzi stosowanych podczas budowy software'u. Model powinien wspierać proces podejmowania decyzji i dawać możliwość wyboru najlepszego oprogramowania w danych realiach. Proponowane rozwiązanie będzie brało pod uwagę wiele czynników, w tym: dotyczące struktury firmy, środowiska i specyfiki projektu czy dostępnych funkcjonalności. Ma ono stanowić wsparcie dla decydentów odpowiedzialnych za wybór odpowiedniego rozwiązania dla projektów inżynierii oprogramowania. Dlatego też planowana jest także budowa bazy wiedzy, która będzie przechowywać sformalizowaną wiedzę dotyczącą tej domeny, w celu późniejszego jej przetwarzania.*

**Słowa kluczowe:** inżynieria oprogramowania, zarządzanie wiedzą, wsparcie decyzyjne, IBM Jazz

Cezary Orłowski
Tomasz Sitek
Department of Management and Economics
Gdańsk University of Technology
ul. Narutowicza 11/12, 80-233 Gdansk, Poland
e-mail: cor@zie.pg.gda.pl, tsitek@zie.pg.gda.pl

Kamil Dowgielewicz
Włodzimierz Wysocki
Department of Electronics and Computer Sciences
Koszalin University of Technology
ul. Śniadeckich 2, 75-453 Koszalin, Poland
e-mail: kamil.dowgielewicz@tu.koszalin.pl
e-mail: wlodek.wysocki@gmail.com

Tomasz Deręgowski
Acxiom Global Service Center Polska
ul. Wołoska 3, 02-675 Warszawa, Poland
e-mail: tomasz.deregowski@acxiom.com