Comparison of workflow software products

Krasimira Stoilova ,Todor Stoilov

***Abstract**: This research addresses problems, related to the assessment of software products, used for the design and exploitation of workflow management systems. The attention is drawn towards the assessment and comparison of such software suits. The lack of direct quantitative evaluations of the products insists to assess and compare the products. The problem solved is the minimization of the subjective influence of the experts in their personal evaluation findings. An idea to overcome this problem is to apply a common evaluation scheme, which is based on objective requirements towards the products.*
***Key words**: web services, workflow systems, automation, business processes*

INTRODUCTION

To implement automation techniques and control methods in the business processes it is necessary to apply modelling techniques for the non-technical and organizational systems and to extend the functionalities of the informational computer driven systems in the organizations. Over the last decade there has been increasing interest in information systems that are used to support, control, and/or monitor business processes. Typical examples of systems driven by implicit or explicit process models are Work Flow Management Systems (WfMS), Enterprise Resource Planning (ERP) systems and Customer Relationship Management (CRM) systems. These systems can be configured to support specific business processes. Several languages have been proposed to support process-orientation in the context of web services (BPEL4WS /Business Process Execution Language for Web Services/, BPML /Business Process Modelling Language/, WSCI, etc). The support of IBM, Microsoft, HP and SAP for a language like BPEL4WS [1] reinforces the fact that process-awareness has become one of the cornerstones of information systems development. Existing languages and tools focus on control-flow and combine this focus with mature support for data in the form of XML and database technology. As a result, control-flow and data-flow are well addressed in languages and systems: BPEL4WS [1], XPDL (XML Processing Description Language) [2]. The technologies in scope are those defined by standardization bodies and initiatives as BPMI (Business Process Management Initiative) [3], ebXML (Electronic Business using eXtensible Markup Language) [4], OASIS [5], WfMC [2] and W3C [6].

A smart technology analysis and comparison is required to make the right technological decisions, and in particular from two key aspects that impact the entire life-cycle of any eBusiness development: the choice of a choreography and orchestration language. These two languages are central to the specification and execution of all workflows:

- Choreography is concerned with global, multiparty, peer-to-peer collaborations where business entities interact in long-lived stateful and coordinated fashion regardless of any programming model or supporting platform used. Choreography languages (e.g. BPSS /Business Process Specification Schema/, WS-CDL /Web Services Choreography Description Language/, etc) cannot be directly executed and have to be translated to an orchestration language in order to be executed.
- Orchestration focuses on the behaviour of a single business entity - it is a hub and spoke model where a controller residing at a single location locally enforces the progress of a process by following its definition. Orchestration languages (e.g. BPML, BPEL /Business Process Execution Language/, XPDL, BPELJ, jPDL, etc) are executable languages and define a runtime environment for their execution.

Choreography and orchestration express the operational semantics of business entities involved in distributed services and complement each other. Choreographies translate global workflows between business entities while orchestrations translate local workflows to a business entity. Global workflows concern the exchange of messages between peers without any centralized control. Local workflows can be either external or internal to a given entity. External local workflows define the public external behaviour of a single entity and differ between entity's roles. Internal workflows are hidden from the outside and they implement external workflows. Workflows can be organized hierarchically in a way that a particular activity of a workflow could itself be realized by a more specific workflow.

Choreography and orchestration languages can be either graphical (e.g. BPMN, UML /Unified Modelling Language/, etc) or textual (BPSS, WS-CDL, BPML, BPEL, etc). Alternative languages exist for both choreography and orchestration. Some can be used for both, although their centre of gravity would be either around choreography or around orchestration.

Orchestration languages are typically high-level specialized programming languages although some languages or language extensions go much closer to general-purpose programming languages like JSR207 and jPDL that facilitates the programming of business workflows directly in Java, or BPELJ that allows integrating Java code (snippets) directly in BPELJ code. Although they can be initially classified, these languages refer to different concepts according to their own creators. They are named for instance execution language, modelling language, definition language, description language, etc. Understanding the exact differences between all these languages, their precise scope, their applicability to any project and evaluating which will emerge, is not an easy task.

Choosing the right language(s) is not the only challenge as many other technologies are also involved. For instance, the format of the messages (usually based on XML) exchanged between the workflow engines offers also a choice between various specifications, e.g. UBL, BPMS, RosettaNet interfaces, OAGIS interfaces, etc. Another example is the choice between communication protocols used between workflow engines, that can fulfil very different roles, e.g. SOAP, which is a synchronous access protocol based on XML, ASAP (asynchronous protocol built over SOAP), BTP (transaction protocol supporting atomic operations and running over SOAP), or Wf-XML (specialized protocol built over ASAP and providing management of workflow engines), etc.

The paper contains an analysis concerning the choice of software products, supporting workflow functionalities. A special emphasis is done on the comparison of existing Open Source software supporting the different technologies.

Methodology for comparison of the workflow software products

A suit of 134 software products are identified, concerning the workflow management domain [7]. These products address different area of system applications (scientific systems, business systems) and they have different level of maturity, functionality, usability. The evaluation process has to tackle a methodological problem which origins from the fact that different experts assess different products. The qualification of the experts, their experience, the variety of the workflow software products, and the lack of common evaluation methodology – all these factors can strongly influence the results of the product evaluation. A second methodological problem arises for the evaluation findings how to quantify the evaluation results to generate a common scale for products comparison. This scale is necessary to support the decision making process for finding good quality and prospective software products.

The paper presents a methodology for evaluation and comparison of the software products. The theoretical background is founded on consideration to minimize the noise influences, which take place in a control and management systems. The formal considerations are given below. The following notations are used:

A_i – assessment rate of software product *i*, *i=1,N*;

N – number of tested and evaluated products;

ε_i^l - evaluation error, performed by expert *l* during evaluation of product *i*, *i=1,N*;

M – number of experts, evaluating the software products;

ε_M - error, which origins from the methodological approach, applied for the assessment of the workflow software products.

The ideal case will be when the expert identify the quality of the software product just as its ideal assessment value A_i. Unfortunately, the background of the expert *j* influences the evaluation findings by his error of incompetence ε_i^j.

An addition noise influence ε_M comes from the methodology applied for the product assessment. Hence the real evaluation values about the quality of the product *i* is:

$$RA_i = A_i + \varepsilon^l{}_i + \varepsilon_M \tag{1}$$

The evaluation methodology has to minimize the influence of ε_i^l and ε_M by means that the working estimations RA_i have to tend towards the real value of the product quality A_i . Because the noise ε_i^l and ε_M are not measurable, during the evaluation process they have to be kept minimal if this is possible. The evaluation methodology is based on a common standard, concerning the quality of software products. Thus the standardization approach targets the minimization of the expert subjective influence to the evaluation findings. The evaluation methodology provides a common evaluation background for all experts. Thus the noise ε_M which arises from the methodological evaluation scheme will be equal to all evaluation findings according to (1).

The next improvement comes from the formal description (1). The idea is not to use the absolute values of the real assessment RA_i, but to make a relative comparison between the evaluated products. It means that if product *i* is assessed according to (1), hence the product j will have analogical assessment RA_j:

$$RA_j = A_j + \varepsilon_j^l + \varepsilon_M \ .$$

But for making a quality assessments of the products, the difference

$$\Delta_{i,j} = RA_i - RA_j$$

has to be considered. The benefit of using $\Delta_{i,j}$ instead of RA_i and RA_j comes from the difference:

$$\Delta_{i,j} = RA_i - RA_j = A_i + \varepsilon_i^l + \varepsilon_M - (A_j + \varepsilon_j^l + \varepsilon_M) = A_i - A_j + (\varepsilon_i^l - \varepsilon_j^l).$$

Thus having the differences between the products *i, j, k*, it follows:

$$\Delta_{i,j} = A_i - A_j + (\varepsilon_i^l - \varepsilon_j^l) \ , \ \ \Delta_{k,j} = A_k - A_j + (\varepsilon_k^l - \varepsilon_j^l). \tag{2}$$

If $\Delta_{i,j} > \Delta_{k,j}$ it can be strongly confirmed that the software product *i* is more qualified than product *k*. This result is influenced by the errors of the expert evaluations $\varepsilon_i^l, \varepsilon_j^l, \varepsilon_k^l$. However, assuming that during the evaluations the expert qualification rises, then the errors vanish:

$$\lim_{t\to\infty} \varepsilon_i^l(t) \Rightarrow 0, \qquad \forall\, i = 1, N, \qquad \forall\, l = 1, N \ . \tag{3}$$

An advantage of the classification scheme (2), based on relative assessments $\Delta_{i,j}$, comes from the fact that the error ε_M , originated from the evaluation scheme, disappears. This is important having in mind that absolute evaluation scheme is difficult to design. Additional benefit of the scheme (2) in comparison with the assessment scheme (1) comes for the expert evaluation background. For the scheme (1) the final evaluation RA_i is directly influenced by the expert incompetence ε_i^l. For the scheme (2) the final evaluation $\Delta_{i,k}$ is influenced by the subtraction of the two incompetences ε_i^l *and* ε_k^l. It means that if the incompetence of the evaluator *l* is the same for the different products, the integral error $\varepsilon_i^l - \varepsilon_k^l$ vanish, which is beneficial for the evaluation process.

For the case when two experts *l* and *m* have to assess different products, the evaluation scheme $\Delta_{i,j}$ is influenced by the incompetence of the both experts ε_i^l *and* ε_k^m. But these kinds of incompetence are subtracted for the overall evaluation to $\Delta_{i,j}$. Thus, for the final evaluation rating $\Delta_{i,j}$ according to scheme (2), the incompetence of the evaluators influence slower the final result when the errors are from the same sign, in comparison to the absolute evaluation scheme, residing on (1). Consequently, the relative assessment of the products, following (2), has three general benefits:

- the error ε_M from the evaluation methodology is suppressed;
- the evaluation findings are influenced by the difference of the evaluator's incompetence, not from their absolute values;
- for the experts, according to their real work during the test of the software products, the absolute incompetence vanish:

$$\lim_{t\to\infty}\varepsilon_i^l(t) \Rightarrow 0, \qquad \forall i = 1,N, \qquad \forall l = 1,M \ .$$

- Following these theoretical findings for the assessment of the software products, the evaluation can be performed in the following order:
- Design of a common evaluation template for the assessment of the quality of the software products;
- To derive appropriate qualitative scheme for the quality of the products. Particularly, a quantification scheme has to be applied for the estimation of the relative assessments RA_i, *i=1,N* for each product. The results of these estimations have been presented as pie- chart diagrams;

The common evaluation template for assessing the software product can be designed based on ISO/IEC 9126 standard for the quality of the software product. The template has to contain the evaluation categories:

1. General categories (G): Workflow software overview with sub-categories
   - G1.1. Workflow software presentation
   - G1.2. Workflow software description
   - G1.3. Category of the software product
   - G1.4. Supported interfaces
   - G1.5. Supported standards. Confirming standards and exchange formats.
2. Functional categories (F): Principle functions
   - F1. Modelling process definition
   - F2. Simulation, debug
   - F3. Execution workflow engine
   - F4 Workflow client application
   - F5 Integration with other workflow engines. Supported standards
   - F6. Administration and monitoring
   - FA. Auxiliary functions: statistics, registration, country area information, help functionalities.
3. Reliability.
4. Usability.

5. Efficiency.
6. Maintainability.
7. Portability.
8. External metrics.

The different categories can be additionally decomposed to give hints to the evaluators by means to decrease the values of the incompetence ε_i^l .

Evaluation Findings

The evaluations are divided into six general criteria, related to the main software quality categories: functionality, reliability, usability, efficiency, maintainability and portability. The evaluations of the functionalities of the products have qualitative nature. They explain and summary the standardization background of the product, its features to cooperate with other software products, the possibility to model, simulate, manage and administer the workflow management processes. The evaluation findings are result from installation, configuration and trial test. The evaluation is given by direct test with the product. An integral evaluation has been performed by giving expert ranking for every software quality subcriteria. Four-level scale has been chosen: week, good, strong, can't assess/not applicable, which formalize the expert opinion for the appropriate quality subcriteria. A particular evaluation finding for the main quality criteria "functionality" is presented in fig.1 for the workflow product Active BPEL Engine.

	Active BPEL Engine				
Characteristics ISO/IEC 9126	**weak**	**good**	**strong**	**can't assess**	**score result**
Functionality	20%	20%	60%		**4,8**
Reliability			75%	25%	**4,5**
Usability		50%	12,5%	37,5%	**2,75**
Efficiency		67%		33%	**2,68**
Maintainability		60%	20%	20%	**3,6**
Portability		20%	60%	20%	**4,4**

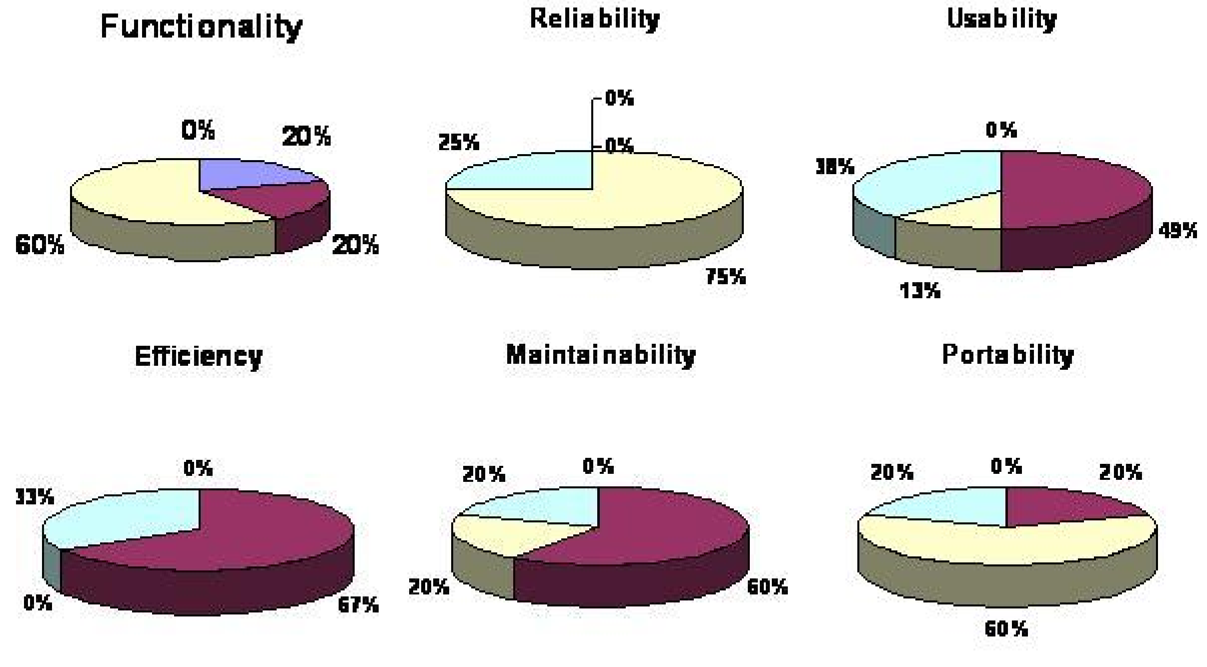

Fig.1. Evaluation findings for the product Active BPEL Engine

CONCLUSIONS

The methodologies used for the evaluation of software products and information resources are strongly influenced by subjective reasons. These influences origin from the evaluation methodology, the choice of the criteria, the incompetence of the experts, the users' expectations of the software products. To minimize these subjective influences, the paper gives preferences to a standardization approach performing a comparison of the products according to the recommendations of the standard ISO/IEC9126 for assessing the quality of the software product. Using the main categorization scheme for quality assessment of the software product, an evaluation template is developed. Thus, using a common evaluation scheme, the evaluations minimize the drawback that different evaluators have to evaluate different software products. The common evaluation scheme and the derived template are prerequisites for minimizing the evaluation errors. A comparative evaluation scheme is developed. It allows minimization of the evaluation errors, originating from the methodological drawbacks of the evaluation scheme and from the personal incompetence of the different evaluators. A relative assessment and comparison is worked out. The quality of the products are assessed by absolute evaluation, marked like $\boldsymbol{RA_i}$. Pie-chart diagrams reflect these quality assessments.

The evaluation was used for the development of the FP6 project: 027178 Virtual Internet Service Provider (VISP), funded by the European Commission.

REFERENCES

[1] Andrews T., F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte, I. Trickovic, and S. Weerawarana, Business Process Execution Language for Web Services Version 1.1, 2003. Technical report, Accessed at http://xml.coverpages.org/ BPELv11-May052003Final.pdf
[2] WfMC, Workflow Process Definition Interface –XML Process Definition Language. Technical Report Document Number WFMC-TC-1025, Workflow Management Coalition, 2002, http://www.wfmc.org/standards/docs.htm
[3] Using BPMN to Model a BPEL Process; Stefan A. White; March 2005
[4 http://ebxml.org/specs/ebBPSS.pdf
[5] OASIS: Web Services Business Process Execution Language; Committee Draft; Version 2.0; December 2005; available at: http://www.oasis-open.org/committees/download.php/16024/wsbpel-specification-draft-Dec-22-2005.htm
[6] W3C: Web Services Addressing; W3C Member Submission; August 2004; available at: http://www.w3.org/Submission/ws-addressing/
[7] Project FP6-027178 VISP. D2.2. Workflow software analysis and comparison, February 2006.

ABOUT THE AUTHOR

Assoc.Prof. Krasimira Stoilova, D.Sc.,PhD, Institute of Computer and Communication Systems, Bulgarian Academy of Sciences, Phone: +359 2 979 27 74, E-mail: k.stoilova@hsi.iccs.bas.bg
Prof. Todor Stoilov, D.Sc., PhD, Institute of Computer and Communication Systems, Bulgarian Academy of Sciences, Phone: +359 2 873 78 20, E-mail: todor@hsi.iccs.bas.bg