Approaches Regarding Business Logic Modeling in Service Oriented Architecture

Alexandra FLOREA, Anca ANDREESCU, Vlad DIACONITA, Adina UTA
Academy of Economic Studies, Bucharest, Romania
alexandra.florea@ie.ase.ro, anca.andreescu@ase.ro diaconita.vlad@ie.ase.ro, adina.uta@ie.ase.ro

As part of the Service Oriented Computing (SOC), Service Oriented Architecture (SOA) is a technology that has been developing for almost a decade and during this time there have been published many studies, papers and surveys that are referring to the advantages of projects using it. In this article we discuss some ways of using SOA in the business environment, as a result of the need to reengineer the internal business processes with the scope of moving forward towards providing and using standardized services and achieving enterprise interoperability.
Keywords: *Business Rules, Business Processes, SOA, BPM, BRM, Semantic Web, Semantic Interoperability*

1 Introduction

In developed societies more than 60% of the work force is dedicated to the delivery of services [1]. From business perspective, the importance of services for modern enterprises is such that it has led to the SOC paradigm [2] [3] and its technological counterpart SOA. In the last year, the buzz around Service-Oriented Architecture (SOA) has diminished as it has entered the mainstream, and at the same time more companies are engaging such projects.
According to some authors [4], SOA has reached the top ten in effective strategies or visions, competing with other strategies such as: virtualization, business intelligence, standardized application platforms, application harmonization, mobility solutions, and collaboration. Still, according to John Ladley [5], only 17% of the firms who answered his survey are running SOA, and only 5% have a robust enterprise view with associated data governance. The average company spends about 85% of its IT budget to keep existing operations and only 15% of its budget on changing the business, innovation, or new capabilities for the business. Companies that invest in SOA are those that know it's wise to spend in order to create capabilities that innovate, differentiate, and change the business in a way that creates desired business outcomes. For those, SOA is the preferred architectural approach, paired with cloud computing and analytics, a blueprint for making both the business and IT more efficient such that a company can spend 40% of its IT budget on creating new business capabilities.

2 Service technologies

The adoption of service oriented concepts has very different motivations for the two categories of intended audience, respectively IT and business, as they see quite a different set of potential benefits or disadvantages.
Looking from the IT perspective, SOA could be viewed as the next step of good engineering practices, going from class reuse to service reuse irrespective of implementation language or host platform. Among the benefits of this approach not only to the IT field but indirectly to the business itself, Scott Glen [6] mentions the decrease of development effort and an improvement in IT project delivery, greater flexibility in business planning provided by the use of interfaces to isolate specific operational systems and an easier introduction of new business system due to the removal of point-to-point connections.
Service-oriented architecture (SOA) represents a significant step towards realizing more dynamic and less expensive integration solutions. Using distinct services to

encapsulate functionality that can later be discovered and used is certainly an important step in the transition from a patchwork of proprietary products and legacy software to a flexible yet robust architecture.
SOA and ESB (Enterprise Service Bus) use the following standards and protocols: J2CA, JMS, SOAP, HTTP, WSDL, JDBC, XML/XSD, XSLT, XPATH, WS-* protocols (like WS-Addressing, WS-Security and WS-Reliable Messaging), BPEL (Business Process Execution Language) and SCA (Service Component Architecture).
An ESB should enable transport to the service consumer and guide have a set of design patterns and best practices to provide the necessary infrastructure to support using a single tool other components that are not using only web services or message driven use cases, but also non-XML based endpoints such as ASCII files, ETL (Extract, Transform, Load) procedures and enterprise applications. ESBs should be flexible at dealing with high volumes, diverse service mediation cases and to scale across multiple projects. Large volumes of stateless messages can usually be processed by mediation only, but sometimes, an orchestration step is required.
Information as a Service models focus on data integration and on optimizing the exposure of information contained in data from multiple disparate sources. Data services can behave as shared components in a SOA Integration solution, making use of other elements in tooling, service integration, process integration and connectivity as a way to simplify complex information-centric integration patterns. SOA Integration and SOA Governance need to work together, governance is essential to ensuring that your SOA has been implemented as intended they give the right control and visibility needed for successful SOA implementations [7].
SOA and BPM can optimize business processes. When a business process model is constructed optimization can begin when runtime feedback is received by the business analyst. Improvements are identified and the models are updated through an iterative business-integration cycle. SOA and BPM should be integrated to give users applications enabling them to share data helping optimize the connection between a business processes and how those processes are translated to the integration[8].

3 Business perspectives for integration

From a business perspective the key word in dealing with SOA is definitely flexibility. Companies must be able to keep pace with the rapid changing conditions of the business environment. In the same time the trend in IT architectures leads toward an integrated model by building business processes that span multiple operational systems and by enabling interoperability between legacy systems and newly developed systems.
With SOA companies have the flexibility needed for implementing such a model and also a framework for using business process modeling as the key technology in handling the ever present business changes. There are different graphical tools that analysts can use in order to define and model the business process and the results of their modeling work can be expressed in the Business Process Execution Language (BPEL). BPEL is an XML vocabulary which can be used in other integration tools that create a link between a business process and the business service designed to realize the tasks of the business process, thus realizing integrated solutions that fulfill the business requirements. Among the advantages of such an approach the following stand out: business-driven development of IT solutions, enterprise solutions, reusable business components and of course business flexibility and agility. In this approach those that set the trend are the business analysts that define the software processes by aligning them to the business needs and thus assuring the business's capability to adapt to change.
SOA Integration implies using Service Oriented Architecture (SOA) to deal with the integration needs of an enterprise. To achieve this, it's not sufficient to have an ESB or BPEL coupled with adapters. A good SOA Integration can help people, business

processes and computer applications to run more efficient and bring advantage to a business by focusing on services that can be shared and reused across the enterprise. A service has reuse potential if it provides capabilities that are not specific to any one business process and it is useful to the automation of more than one business process [9].
Utility services are frequently process-agnostic because they are intentionally designed to not encapsulate business logic. Business services need to be well designed to avoid being tied to parent business logic. Platform dependencies limitations can be avoided by making the service capable of encapsulating logic from different application environments. Not linking services to specific processes and proprietary implementation eases building an inventory of services that can be reused and composed when new requirements appear. SOA provides a framework for enabling services, data, and events, and connecting them to better align with business requirements. It supports key integration patterns that allow IT managers and IT architects to aggregate, orchestrate, and mediate these services increasing the responsiveness to the constant changes in business requirements [10].

4 Business Logic Implementation in SOA

Business logic is the defining element for a business being in the process of modeling and automation, and it includes both business rules and workflow (process), which describes the transfer of documents or data from one participant (person or software system) to another. Business Rules refers to the multitude of policies, procedures or definitions that govern how an organization works together with its interaction with customers or partners. These may be external rules, coming from legal regulations that must be observed by all organizations acting in a certain field, or internal rules which define the organization's business politics and aim to ensure competitive advantages in the market. Starting from the previous observations, it is obvious the important role that business rules play within the development process of a software system.
The success of business rules' adoption in a software project largely depends on the ability to separate and independently design business rules and business processes in order to reuse them. Design solutions that have succeeded to address this separation problem propose that the distinction between business rules and processes should start from the role that each one plays within the underlying business, in that business rules produce knowledge and business processes run business activities [11]. Recent years have been marked by an increased interest for a new type of software products, called Business Rules Management Systems (BRMS). These systems externalize business rules and provide facilities for a centralized business rules management (BRM), frequently by using a business rules engine. They also offer solutions for compelling problems facing any business: business rules changes in response to increasingly rapid pace of change and the short time required for the implementation of these changes in the software system.
Historically, research and practical solutions in business process management (BPM) preceded the appearance of business rules approaches (the latter tried to impose rules as a central element in the software development process) Ronald Ross [12], one of the promoters of business rules approaches, emphasized the importance of focusing on business rules because "[…] business processes are not so simple. In fact, they are quite complex and therefore difficult to change". This is because processes highly depend on rules.
In situations that involve simple rules, process engines will likely implement the rules directly, but in order to handle more complex situations, most process engines have the ability to call other components that implement rules.
It's a fact that SOA is centered on building services that are especially designed for interoperability from the very start. But SOA is also about raising the abstraction levels of

interfaces: interfaces that must support the business, not the system [13]. The combination between business rules and web services offers an adequate approach for applications integration and sharing of distributed information. Business rules adoption, together with a service-oriented architecture allows the integration of strategic corporate applications between multiple business units. For example the same business logic that has been explicitly defined in a BRMS may be shared in a SOA with other applications that need it. These applications communicate via XML with the Business Rules Services [14].

According to [15] using BRM and BPM together in a SOA represent the methodological and technical requirement for industrializing business processes and being agile. Applications constructed using BPM software may include web services, rules contained in a rules engine and control and flow business logic, part of the process, coded or rendered using BPM software. The SOA and Business Process Management (BPM) developers and researchers focus on the technology to build and operate service networks and to automate the business processes that take place inside them. In the paper [16], the authors envision service networks modeling as the means to gain better alignment between the business and IT perspectives in enterprises. The enhanced alignment is achieved by (1) providing an overview of inter- and intra-enterprise business relationships in terms of service providing, (2) supporting decision making on service networks in terms of business relationships between participants, and (3) facilitating the propagation of changes from service networks to the underpinning software service infrastructures and vice versa.

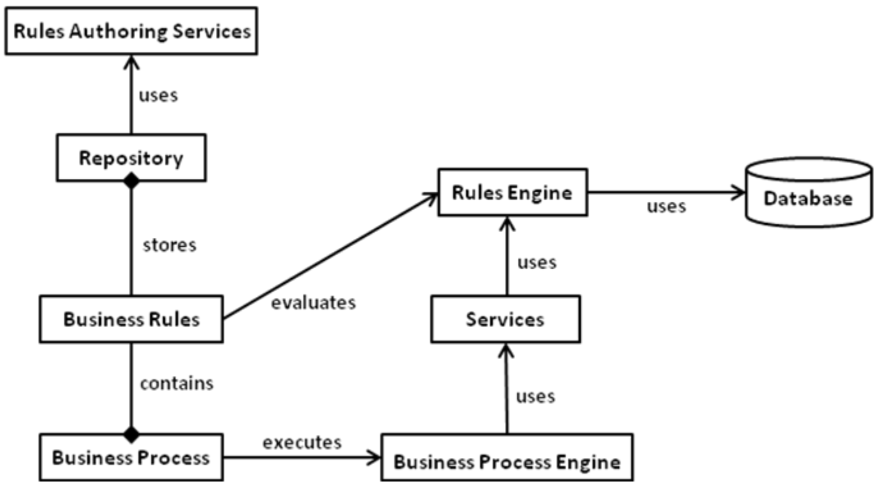

Fig. 1. Usage of business rules and process engines in SOA [adapted after [17]]

Figure 1 describes the common usage of business rules and process engines in the SOA implementation.

The problem is many consider the business architecture as simply a step in developing the informatics architecture, thus managers rarely participate in enterprise architecture or business architecture development, therefore minimizing the effectiveness of the resulting business architecture. The solution is that the stakeholders realize the importance of the business architecture, it's an asset of the firm and it can greatly impact one's results.

Table 1. Value-added to the integrated solution by each technology

	Value-added
BRM	• standardization and transparency of business policies and principles • business rules externalization from the core applications • business users can access and manage business rules
BPM	• standardization and automation of business processes • flexibility in business activities ordering • ability to call business rules services
SOA	• separates specific types of logic included in processes and decisions • business rules becomes services that can be reused across systems • minimize the impact of expected changes

Each one of the above technologies brings an added-value to the final integrated solution, as summarized in Table 1.

5 Semantic interoperability in SOA

One of the most important architectural features of SOA is the semantic interoperability. This is one of the cornerstones of service-oriented approach because it ensures fair and consistent data exchange between clients and service providers involved. Lack of semantic interoperability leads to erroneous interpretations of the message sent and to data corruption, due to different interpretations that can be granted to the same concepts by the two parties. In essence, semantics is the foundation of SOA, without it data being nothing more than meaningless binary strings.

Despite its great importance, in the present semantics is not given due attention, it represents a secondary concern when design and implementation activities occur. This is mostly due to the fact that developers tend to take semantic interoperability for granted especially because the semantic interpretation, mapping and transformation are so ingrained with home grown applications, Enterprise Application Integration (EAI) and Enterprise Information Integration (EII) [18].

When there is an exchange of messages through a service, there are a number of mappings of names, values and structures that appear in these messages. These mappings are accomplished through message transformation rules and make the necessary semantic correspondences for a common understanding of the message by the issuer and receiver. When performing such mappings, no explanation of how the source and target refer to real entities is given. Basically the message transformation rules that are written and used should be taken as such because there is no accounting for the reason why the corresponding mappings are sustainable. This way of looking at and addressing semantics, without in depth explanations can be called neutral.

This way of addressing the problem of semantic interoperability is a pragmatic one, which can be used without problems in many current situations as it is reasonable to believe that usually, the partners involved in the exchange of information through services, have the same interpretation of their description. Often, this neutrality is based on a relatively realistic assumption, namely that the attributes that shape the architecture of services is set in a natural way, once and for all. Thus, because by default the same services using ontology, developers' role is merely to neutralize a number of differences in names and representation of elements that are part of a unique, universal conceptualization [19].

This approach cannot be widely used, although in some cases it is reasonable, because it starts from the assumption that there is an implicit consensus between the parties involved, a consensus that goes beyond infrastructure and is established in the social system which includes the information system. This hypothesis is unreliable considering that SOA is used for developing systems for large, geographical distributed organizations and therefore have representatives from different social environments. Basically, under neutral approach, we can assume that two attributes refer to the same item if they have the same name. This would be true if all parties have the same interpretation of labels used. For example, when the infrastructure of services is developed, used and managed in an organization where there are rules and naming policies that can be applied and controlled. But when discussing about various environments, designers cannot make assumptions about the interpretations that are given to the concepts used. In fact, they must work with information providers which are only required to give a brief description of the types of services they make available, especially from a functional perspective, so that no information about the coding and interpretation of data is available. Semantics, which is how the services actually carried out what their description promises, is encapsulated in the "black box" of service implementation [19].

The basic idea to remember is that no single interpretation of a service can be made relying only on a description of the service. In working with services no trustworthy semantic assumptions can be made. Without semantic interoperability, there can be no assurance that the data encapsulated in the messages exchanged through the service, is interpreted by the parties that interact as the same concepts, relationships or entities, so there are chances that they will be misinterpreted and ultimately bring harm to the business.

There are many templates proposed to achieve semantic interoperability, and according to [18] they can be grouped into five broad categories: point to point type semantic integration, hub and spoke type semantic integration, master data management (MDM), models of industrial information and semantic Web. The semantic Web goes beyond the borders of applications, organizations and industries. It makes connections between data models and elements of a common ontology and uses RDF (Resource Description Framework) and WOL (Web Ontology Language) for allowing data to be shared and reused online.

6 Business rules in the semantic Web

Due to the growing volume of information available on the Web, it is difficult to fully automate their retrieval, much less by the human operator [20]. This is also one of the objectives of the Web Consortium, which seeks to identify ways, based on XML technologies, for solving the problem of computer-based information processing in cyberspace. A viable solution appeared to be the semantic web, which is a "consistent and logical web of all resources on available on the Web, with emphasis on machine data interpretation and not on their representation" [21].

Semantic Web architecture is functional one, because its development is based on incremental languages' specification, starting from the lower level (the metadata) through the upper levels (logic level) [22]. Languages available on each level can meet the requirements imposed by different types of applications: 1) *metadata level* provides the overall framework for expressing simple semantic assertions. The model includes concepts such as resource and property, in order to express meta-information. The language is specified via RDF and the various DCMI metadata vocabularies (Dublin Core Metadata Initiative), RSS (Rich / RDF Site Summary), FOAF (Friend Of A Friend); 2) *scheme level* enables simple specification of the ontology in order to define a hierarchical description of the concepts and properties, 3) *logic level* introduces more complex ontology

languages, capable of sophisticated ontology model.
So the Semantic Web is applying the idea of knowledge-based systems and ontologies for the Web. But when their manipulation is possible through computers, ontologies can be viewed as metadata which explicitly represent the semantics of data in such a way that enables direct computer processing.
While de semantic web is centered on data integration, SOA deals with operations and services in addition to the data. The intersection of these two approaches is found in the form of semantic web services [23]. Semantic services can be considered a component of the semantic web as they use markups, which make data machine-readable in a detailed and sophisticated way. Besides these, markup languages for business rules allow the specification of rules as independent and modular units in a declarative manner and also their publication and interchange between different systems or tools. Wagner [24] predicted that they will play an important role in the Web for business- to-customer (B2C) and business – to - business (B2B) interactions. The scope of rule markup languages can be extended from the two areas mentioned above and their contribution can have a positive impact on other domains, such as e-government. And since now SOA is an important paradigm in developing system applications and ensuring interoperability between these, the implementation of business rules in a SOA using markup languages is expected to led to a bigger interoperability level, as we will further discuss.
Studies about introducing rules on the Web have increased consistently since the year 2000 with the presentation of an initiative called RuleML. Rule Markup Language (RuleML) is a markup language that was proposed by the Rule Markup Initiative as the standard language for publishing and distributing rules specific to Web based applications [25]. In fact, RuleML specifications represent a modular family of sublanguages for the Web, which root allows access to the language as a whole and whose members allow the identification of customized subsets of the language. Each sublanguage has its own definition of an XML schema, identified by a URI. At the first level of the modular structure, RuleML family distinguish between derivation rules, queries and integrity constraints, but also production rules and reaction rules. The language's most developed branch groups derivation (deduction) rules, which in turn are based on a core language called Datalog and two major side branches named Hornlog and FOL (First Order Logic). Being now at its 1.0 version, RuleML is implemented using XML schemas, XSL transformations (XSLT) and reasoning engines. RuleML is also extensible, as examples we can mention its combination with WOL that led to the formation of the Semantic Web Rule Language (SWRL) and its object oriented extension called OO RuleML [26].
Since RuleML should facilitate interoperability between systems that use rules, the possibility of converting it (using XSLT) to other Semantic Web standards (like RDF or WOL) or specific rules engine languages (such as Jess) is particularly important.
RuleML intends to cover a wide range of business rules types. In this way, it could be used to specify queries and inferences in Web ontologies, or to make correlations between Web ontologies and dynamic behavior of workflows, services and agents within the Web environment [27].
Particularly important in a software system is the possibility to define implications or inference rules, especially when they relate to each other and require chaining mechanisms in order to execute rules in the correct order. RuleML Datalog implication rules confront facts with rules with the purpose of producing new knowledge. As an example, we will consider the next two implications (or inference) rules which are specified in RuleML:
Rule 1: A customer enters the Gold category if he gathered, through his orders and behavior, a score of at least 50 points.

```
<Implies>
  <if>
```

```
        <Atom>
          <Rel>points</Rel>
          <Var>client</Var>
          <Data>min 50</Data>
         </Atom>
      </if>
      <then>
         <Atom>
          <Rel>Gold</Rel>
          <Var>client</Var>
         </Atom>
      </then>
    </Implies>
```

Rule 2: The discount for a customer is 30% if he placed an order of at least 5000 monetary units and falls into the Gold category.

```
    <Implies>
      <if>
        <And>
          <Atom>
            <Rel>Gold</Rel>
            <Var>client</Var>
          </Atom>
          <Atom>
            <Rel>value</Rel>
            <Var>order</Var>
            <Data>min 5000</Data>
          </Atom>
        </And>
      </if>
    <then>
        <Atom>
          <Rel>discount</Rel>
          <Var>client</Var>
          <Var>order</Var>
          <Data>30 procent</Data>
        </Atom>
    </then>
  </Implies>
```

The main idea between these rules' definitions is that data must take the form of RuleML facts, which will be subsequently mapped to the inference rules. In order to be able to execute the above two rules, the following two facts have been considered:

Fact 1: The client named Tom has gathered a score of at least 50 points.

```
<Atom>
      <Rel>points</Rel>
      <Data>Tom</Data >
      <Data>min 50</Data>
</Atom>
```

Fact 2: The value of the order with the code OR20157, placed by Tom, has exceeded 5000 monetary units.

```
<Atom>
       <Rel>value</Rel>
       <Data>OR20157</Data>
       <Data>min 5000</Data>
</Atom>
```

Regarding the structure of RuleML language, the superior positioning of tag <Implies> can be observed. This contains the tags <if> and <then> representing the rule's condition and respectively the rule's action. Within these tags, relationships between the elements of the analyzed domain are defined in the form of atomic formulas, as indicated by the tag <Atom>.

As shown in figure 1, the reasoning process starts from Rule 1 and Fact 1 that are used together for generating a first derivation (Fact 3): the rule's atom between the **if** tags matches the fact atom, binding `<Var>customer</Var>` to `<Data>Tom</Data>`. Then this binding is used to instantiate the same variable in the rule's **then** tags and a new `<Atom>` is derived expressing that:

Fact 3: Tom is a Gold customer.

```
<Atom>
     <Rel>Gold</Rel>
     <Data>Tom</data>
</Atom>
```

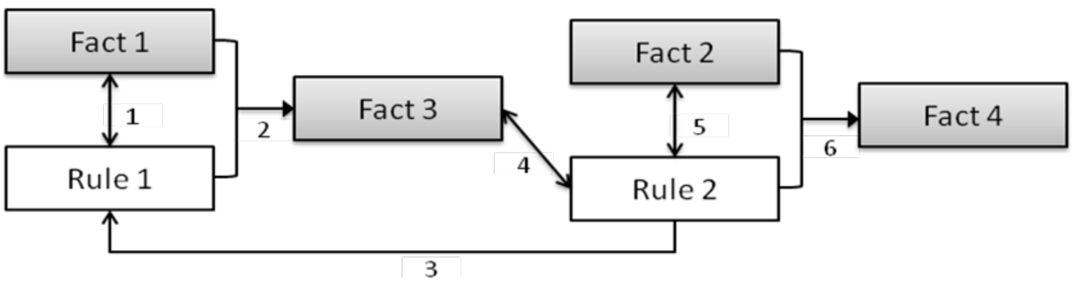

Fig. 2. Steps of the reasoning process for the exemplified RuleML rules

Rule 2 is composed of two atoms connected by the `<And>` tags. First of them, `<Rel>Gold</Rel>`, chains to Rule 1, which has evaluated Tom as being a Gold customer. Then, the `<Rel>value</Rel>` atom matches the second atom from Fact 2, binding `<Var>order</Var>` to `<Data>OR20157</Data>`. Therefore, because the bindings in Rule 2 have succeeded, a new fact is generated:

Fact 4: For the order number OR20157, client Tom will have a 30% discount.

```
<Atom>
     <Rel>discount</Rel>
     <Data>Tom</Data>
     <Data>OR20157</Data>
     <Data>30 procent</Data>
</Atom>
```

Being based on XML, RuleML inherits all its benefits, including platform independence and interoperability. But it is a low level language, making it difficult to be adopted by business people. As an attempt to increase RuleML's level of abstraction, open source tool TRANSLATOR [28] was created and it automatically translates the statements written in natural language similar to the so called Attempto Controlled English, in RuleML rules.

7 Conclusions

Like other technologies such as structured design and analysis, databases, information engineering, object oriented development, frameworks and patterns, SOA will continue to be well-established as a best practice. The adoption of SOA together with BRM and/or BPM will allow organizations to integrate and deploy new applications more easily due to SOA's significant interoperability advantages.

BPM should not be viewed only for the creation and customization of applications. Over time, business process logic remained encoded only at the level of the applications, described in the program lines, making it difficult and costly to change. These problems have lead to changes being made by duplicating parts of the business functionality and by manual workarounds. New technologies such as SOA and BPM, used together, are making possible readjustments in the IT budget to encourage innovation and developing fresh capabilities for the business.

Compared with rule engine languages or proprietary languages of Business Rules Management Systems, the foremost advantages of RuleML in a service oriented environment are its markup nature, which leads to interoperability capabilities, the ability to define a wide range of rules categories, business rules externalization from core applications, but also the possibility to be translated in other web standards or languages as mentioned above (capability defined as rule interoperation between industry standards). However, its major drawbacks have delayed large scale adoption, as RuleML is too technical for allowing business people to define and modify rules and, in addition, few tool support and practical applications are available. Nevertheless, for software applications that relay on semantic web and/or SOA, the implementation of business rules using markup languages might be the "real" solution when dealing with rules in the Web.

References

[1] P. Maglio, S Srinivasan, J. T. Kreulen, and J Spohrer, "Service systems, service scientists, and innovation", *Communications Of The ACM - Services Science*, Issue 49(7), pages 81–85, 2006.

[2] M. P. Papazoglou, P. Traverso, S. Dustdar and F. Leymann, "Service-oriented computing: State of the art and research challenges", *IEEE Computer*, Issue 40(11), Pages 38–45, 2007.

[3] M. P. Papazoglou, P. Traverso, S. Dustdar and F. Leymann, Service-Oriented Computing: A Research Roadmap, in "*International Journal Of Cooperative Information Systems*", vol. 17, n. 2,2008, pp. 223-255.

[4] K. Holley and A. Arsanjani, "*100 SOA Questions Asked and Answered*", Prentice Hall, 1 edition, ISBN 978-

0137080205, pages 220-221,November 22 2010.

[5] J. Ladley, "SOA Survey Results", *Enterprise Data Journal*, November 2010.

[6] S. Glen and J. Alexander, "A practical application of SOA", IBM white paper, 2007

[7] G. Sabau, A.R. Bologa, R. Bologa, M. Muntean, "Collaborative Network for the development of an Informational System in the SOA context for the University Management", *International Conference on Computer Technology and Development* (ICTD2009), Malaysia 2009, IEEE Computer Society, pp 307-311, ISBN 978-0-7695-3892-1.

[8] A.R. Bologa and A. M. I. Florea**,** "Approaches in Business Oriented Management of Web Services", The Ninth International Conference On Informatics In Economy, *The Proceedings of the Ninth International Conference On Informatics In Economy*, Bucuresti, Mai 2009, pg 233-239, ISBN : 978-606-505-178-2.

[9] T. Erl, "*SOA, principles of service design*", Prentice Hall, ISBN 978-0-13-234482-1, 2007.

[10] D. Hansen, **"***Blueprint for Successful SOA Integration"*, Oracle Paper, March 2008.

[11] T. Morgan, "*Business Rules and Information Systems: Aligning IT with Business Goals*", Addison Wesley, 2002.

[12] R. Ross, "*Principles of the Business Rules Approach*", Addison-Wesley, 2003.

[13] I.Graham, "*Business Rules Management and Service Oriented Architecture: A Pattern Language*", Wiley, 2007.

[14] M. Mircea and A. Andronescu, "Using Business Rules in Business Intelligence", *Journal of Applied Quantitative Methods*, Volume 4, Issue 3, September, 2009.

[15] W. M. Tem, "*Role of Business Rules in SOA*", Annecy, France, 2006, available at: http://www.visual-rules.com/fileadmin/pdf-en/white-paper/business-rules-processes-soa.pdf

[16] D. Olha, K. Dimka and L. Frank, "Service Networks Modeling: An SOA & BPM Standpoint", *Journal Of Universal Computer Science*, Volume 16, Issue 13, Pages: 1668-1693, 2010.

[17] B. Lublinsky, D. Le Tien, *Implementation of business rules and business processes in SOA*, March, 2007, available at: http://www.infoq.com/articles/business-rules-processes.

[18] S. M., W. D., Zurek B., Kahan E., Achieve semantic interoperability in SOA, *IBM Technical Library*, June 2006.

[19] G. Vetere, M. Lenzerini, Models for semantic interoperability in service oriented architecture, *IBM Systems Journal*, Vol. 44, Issue 4, 2005, pp 887 – 903.

[20] S.C. Buraga, '*Semantic Web – fundamente şi aplicaţii*", Matrix Rom, Bucharest, 2004.

[21] T. Berners-Lee, J. Hendler, O. Lassila, "The Semantic Web", *Scientific American Magazine*, May 2001.

[22] S.C. Buraga, "Considerations Regarding the Use of Semantic Web Technologies in the Context of E-business Applications", *Informatica Economica Magazine*, Bucharest, number 3, 2005.

[23] Semantic Web Services, available at: http://en.wikipedia.org/wiki/Semantic_Web_Services

[24] G. Wagner, "How to Design a General Rule Markup Language", presented at the workshop "*XML Technologies for the Semantic Web*" (XSW2002), Humboldt University, Berlin, Germania, June 2002

[25] The Rule Markup Initiative - RuleML, available at: http://ruleml.org/

[26] H. Boley, "*The RuleML family of web rule languages*", the fourth international workshop "Principles and practice of semantic web reasoning", Budva, Montenegru, June 2006.

[27] M. A. Cibrán, "*Connecting High-Level Business Rules with Object-Oriented*

Applications: An approach using Aspect-Oriented Programming and Model-Driven Engineering", PhD thesis, Vrije University, Brussel, June 2007.

[28] D. Hirtle, *TRANSLATOR: "A Translator from Language to Rules"*, in the proceeding of Canadian Symposium on Text Analysis (CaSTA'06*)*, NewBrunswick, Canada, 2006.

Alexandra Maria Ioana FLOREA is a pre-assistant lecturer at the Economic Informatics Department of the Faculty of Cybernetics, Statistics and Economic Informatics of the Academy of Economic Studies. She has graduated from the faculty at which she is now teaching in 2007 and also from the Faculty of Marketing in 2008. Since then she is a PHD candidate, studying to obtain her PhD in the field of economic informatics. At present her fields of interest include integrated information systems, information system analysis and design methodologies and database management systems.

Anca Ioana ANDREESCU is university lecturer in Economic Informatics Department, Academy of Economic Studies of Bucharest. She published over 15 articles in journals and magazines in computer science, informatics and business management fields, over 20 papers presented at national and international conferences, symposiums and workshops and she was member in over nine research projects. In January 2009, she finished the doctoral stage, the title of her PhD thesis being: *The Development of Software Systems for Business Management*. Her interest domains related to computer science are: business rules approaches, requirements engineering and software development methodologies.

Vlad DIACONIȚA is an Assistant Lecturer at the Economic Informatics Department at the Faculty of Cybernetics, Statistics and Economic Informatics from the Academy of Economic Studies of Bucharest. He has graduated the faculty at which he is now teaching in 2005 and has a PhD in the field of Cybernetics and Statistics. He is the co-author of 2 books in the domain of economic informatics, 3 articles in ISI journals, 4 articles in Scopus journals, 5 articles in ISI proceedings, 6 papers in B+ journals and 6 papers in the proceedings of international conferences. He participated as team member in 3 research projects that have been financed from national research programs. He is a member of the IEEE Computer Society and the INFOREC professional association. Domains of competence: Database systems, Data warehouses, OLAP and Business Intelligence, Integrated Systems, SOA.

Adina Ileana UȚĂ is university professor in Economic Informatics Department, Academy of Economic Studies of Bucharest. She published over 30 articles in journals and magazines in computer science, informatics and business management fields, over 50 papers presented at national and international conferences, symposiums and workshops and she was member or director in over 40 research projects. In 2000 she finished the doctoral stage, the title of her PhD thesis being: *The automation of analysis and forecast activities related to final energy consumption*. Her interest domains related to computer science are: software development methodologies and development systems.