Cloud-based Virtual Organization Engineering

Liviu-Gabriel CRETU
Alexandru Ioan Cuza University, Iasi, Romania
liviugabriel.cretu@gmail.com

Nowadays we may notice that SOA arrived to its maturity stage and Cloud Computing brings the next paradigm-shift regarding the software delivery business model. In such a context, we consider that there is a need for frameworks to guide the creation, execution and management of virtual organizations (VO) based on services from different Clouds. This paper will introduce the main components of such a framework that will innovatively combine the principles of event-driven SOA, REST and ISO/IEC 42010:2007 multiple views and viewpoints in order to provide the required methodology for Cloud-based virtual organization (Cloud-VO) engineering. The framework will consider the resource concept found in software architectures like REST or RDF as the basic building block of Cloud-VO. and will make use of resources' URIs to create the Cloud-VO's resource allocation matrix. While the matrix is used to declare activity-resources relationships, the resource catalogue concept will be introduced as a way to describe the resource in one place, using as many viewpoints as needed, and then to reuse that description for the creation or simulation of different VOs.

Keywords: *Virtual Organizations, Enterprise Engineering Framework, Cloud Computing, REST*

1 Introduction

For many years, a lot of work has been done around the virtual organization concept (VO) generating two main streams of research: agent-based systems (individuals, agents, goals, individual and group behavior, rules) [1], [2], [3] and service-based systems (systemic approach on relationships between objectives, events, entities, nodes, services, and the required coordination and management frameworks) [4], [5], [6], [7]. From the formation methodology point of view, there are two types of VOs [8]: 1) *emergent VO* – a request is sent to a virtual market, there is an auction process taking place and, finally, a broker will decide the structure of the consortium that will actually process the request; 2) *designed VO* – once the opportunity has been identified, the broker will proceed with a top-down design process in order to select the required services and to form the VO.

On the other hand, the technology evolved and recent years have generated a new software delivery paradigm known as Cloud Computing. It encompasses Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). Among these types, SaaS is a software delivery model, which provides access to business functionality remotely (usually over the internet) as a service [9].

In this paper we will identify the relationship between the two elements: the organizational concept (virtual organization) and the technology (Cloud Computing). We consider that virtual organizations should be seen as service-based socio-technical systems and that they should be engineered (top-down approach as opposed to ad-hoc formation) following the cybernetics principles and the economic laws (business objectives, cost, profit). Moreover, in the context of Cloud Computing paradigm, the nature of commodity-like capabilities delivered by cloud services and the inherent challenges in this business model drive the need for Cloud-based VO engineering as the process of designing the VO system such that to use cloud resources in order to respond to business opportunities. There are two main assumptions that emerge from the above statements: 1) structural analysis is the more appropriate model for research in this field (focus on the concepts used to describe a Cloud-based VO and the building-blocks of this model) and 2) there are multiple perspectives that may be used to analyze such a system (the service models, the information processed within VO, the work-flow of activities, the cost of the process, the agents that approve/execute the activities etc.).

2 Virtual Organizations

The basic definition of virtual organizations (VO) can be fairly simple: organizations and individuals that dynamically inter-connect in order to share and use resources by means of temporary alliances. However, complex problems arise on different abstraction levels when one starts to analyze concepts like inter-connections, resource sharing and alliances. The complexity is generated by the multi-disciplinary approach needed to design, operate and manage VOs as socio-technical systems.

The VO term has its roots back to the early 1990's when Raymond Miles and Charles Snow [10] first described the *agent-broker network organization* (dynamic network). Later on, the idea has been transformed into a new organization design paradigm by popular works of Dvidow, Hammer, Cunningham, Adam [11], [12], [13]. All these theses share the same vision of an organization system with the following distinctive characteristics: vertical disaggregation, internal and external brokering, full-disclosure information systems, and market substitutes for administrative mechanisms. It is a vision that gives to information technology (IT) the key role in creating the links between various resources shared throughout the VO. The Networks became one of the practical examples of the applying the well-known model of competitive advantage [14] in the real business world. From a distinct strategic position, the broker uses the instrument of the network as an effective organizational form to create value in the industry (in new niches, demand pockets) and to capture value for the individual company. Following this approach, the Network shares also the basic characteristics of any organization, as defined by Galbraith in '77 [15]: (1) groups of people, (2) common goal, (3) division of labor, (4) integration by information based processes.

One of the most distinctive building blocks of VOs vs. other organization forms is the "switching principle" defined by Mowshowitz in 1999 [17]. This principle basically states that the broker or the final customer can dynamically re-allocate resources to design virtual activities. A common example of applying the switching principle is the order payment process when the user or the system (by means of a set of pre-defined business rules created by the broker) may select one type of payment, from a list of available methods, based on the process execution's context variables. The switching principle generates yet another viewpoint over the virtual organizations as systems created on the basis of resource selection from the (electronic) market: the need to calculate the market transaction cost [17] as the cost for searching the right partner or specifying the transaction.

From a methodological perspective, we can summarize the literature on virtual organizations formation by two possible approaches:

1. *emergent virtual organizations* - a demand is placed on a virtual market, there is a bidding process and, finally, a broker decides the structure of the consortium that will be created to process the request;
2. *designed virtual organizations* - once a collaboration opportunity is detected, a member playing the role of the broker will launch a top-down design to create the virtual organization;

Since the network is the basic organization form for VOs, the next natural question would be: do the network topologies have any influence on the VO structure? By analyzing the relevant literature, Katzy et al. [19] identified three VO types that seem to be acknowledged by many authors: supply-chain VO in manufacturing industries, star (main contractor) VO in construction industries, and peer-to-peer VO in creative and knowledge industries. These types are based on basic network topologies. In a supply-chain topology, it is the business process that is designed and governs the partners' interaction. In a star topology, partners interact with one central hub or strategic centre, while partners in peer-to-peer topology have multiple relationships between all nodes without hierarchy.

3 Frameworks for Virtual Organizations

Regarding the frameworks that should guide the formation, implementation and the management of virtual organizations, current research is directed almost exclusively to identify the optimal model for coordinating the services / autonomous agents. In this respect, we can identify three main models regarding the conceptual framework and the coordination mechanisms:

- agent-based models and rules;
- models based on services and service oriented architectures;
- models based on semantic Web technologies;

The main problem identified in a VO system is how to ensure cooperative behavior in scenarios populated with heterogeneous agents and led by their own interests. Castelfranchi summarizes relevant literature and identifies two main areas of research [20]: 1) imposing restrictive facilities for the actions of agents, and thus being impossible for them to deviate from the desired behavior (the approach severely limits the autonomy of agents), 2) restricting the environment, in which agents interact, through the use of business rules and leaving the freedom for the agents to follow or violate them. Usually, the first case deals with the relationship between tools for workflow management and agent-oriented systems while in the second case, the concept of electronic institutions is introduced as a virtual replica of the institutions that govern the real world.

In a recent article, McGinnis and colleagues published a framework for designing virtual organizations seen as a result of interconnections that take place in a society of agents [21]. More details on this may be found in other two papers: a) a voting protocol for the agents that make up the VO [22], b) the formal representation of contracts [23]. Moreover, other works analyze the norms that may be applied to the behavior of agents through the so-called electronic institutions (EI). An electronic institution is considered a key component in the supervision of agent-based virtual organizations. A set of rules are declared by the EI to govern the public behavior of agents. In this regard, Sierra et al. [24] propose a framework for defining and applying such rules. The authors aim to combine the Islander (a pragmatic modeling language for electronic institutions) with a methodology for the development of intelligent agents (Prometheus). Oliveira and Lopes have also develop a framework [25] based on the use of a rules engine to apply a set of rules (the normative system) in a context called "institutional reality" (body of facts that exist into the engine's working memory at a certain moment). The agents will then always act within this kind of context. The authors identify three types of rules: constitutive rules, institutional and operational. Similar approaches that propose the use of rules as a restrictive environment for agents' behavior can be found in [26].

There are a number of works which try to demonstrate that the creation of virtual organization can only occur through the integration of ontologies and semantic technologies in service-oriented systems. Thus, in [27], [28] and [29] we can find an ontology-oriented service-based VO modeling framework addressing the inherent inter-operability problems that can arise in heterogeneous service-oriented environments. The authors present a simplified architecture that facilitates the dynamic reconfiguration of services based on requests expressed by customers. A request is sent to the system and is served by an ad-hoc organization of heterogeneous services that have been previously registered in a semantically-harmonized environment based on Semantic Web technologies.

Camarinha-Matos and Afsarmanesh [8] describe a framework for the creation of an OV in a breeding environment: (1) characterization of the opportunity for collaboration; (2) creation of the VO draft plan; (3) search and selection of partners; (4) negotiations; (5) detailed plan of the VO; (6) contracting; (7) launching. From an architectural perspective, Boukadi proposes a framework based on a multi-layer SOA and the concept of community [31]. The authors seem to be among the firsts to identify the need for a VO management system and the proposed multi-layer architecture consists of four manager roles (community, reputation, resources and decision making), which coordinates the operational services layer. Inter-operability is reached by a layer of semantic services and domain ontologies.

4 The Cloud-based virtual organization

In recent years, a new paradigm known as Cloud computing has been added to software engineering landscape encompassing Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). Among these types, SaaS is a software delivery model, which provides access to business functionality remotely (usually over the internet) as a service [9]. Thus, Software as a Service introduces both a new business model and a new software architecture model. The very essence of this architectural paradigm shift is the ability to embed tools and techniques to capture *common and variable* features of various business models within the software at run-time instead of design-time. From the manager's perspective, the following three characteristics are essential to any enterprise Cloud [35]:

1. Configurations are dynamic and automated (or semi-automated) in varying and unpredictable ways, and possibly even include event-driven conditions.
2. Systems management technologies are scalable so that they are manageable in aggregate conditions (e.g., integration of business constraints with infrastructure constraints).
3. A Cloud is secure and has the necessary information assurance capabilities.

SaaS seems to be the most familiar type of Cloud Services to everyday Web users. The application services layer host applications that fit the SaaS model. These are applications that run in a Cloud and are provided on demand as services to users. And we should not only mention the widely used free services for general use like Google Docs. There are enterprise targeted hosted software offerings available on the Internet that handle payroll processing, human resource management, collaboration, customer relationship management, business partner relationship management, to name only a few of them. Popular examples of these offerings include IBM Lotus Live, IBM Lotus Sametime, Unyte, Salesforce.com, Sugar CRM, and WebEx. In all cases, applications delivered via the SaaS model benefit consumers by relieving them from installing and maintaining the software, and can be used through licensing models that support pay-per-use concepts.

We define the Cloud-based VO (Cloud-VO) as a business process made of activities that may allocate resources from different Clouds in order to respond to business events. By this approach we are committing to the designed VO type as opposed to the emergent VO described earlier in the paper. The process is designed by a broker that will indentify the required activities and the Cloud services that could optimally respond to the opportunity that triggered the VO formation. The process is coordinated by a work-flow engine and a set of business rules. There are two types of business rules: Cloud service provider rules and the VO activities specific rules. The business process itself does not have any associated business objectives. The objectives will have to be specified for each activity and thus leaving the optimization of the whole business process to take place gradually, as long as each activity is analyzed and optimized. Figure 1 shows such a VO that allocates resources from Salesforce.com, Google Apps Cloud, and Amazon Cloud Services together with a specific DHL service. All these services are orchestrated by the business process engine.

For each activity there is a number of well-defined elements that constitute the building blocks used to analyze and design that activity: 1) the event that triggers the activity (When); 2) the service that will be executed (How); 3) the result produced when the activity completes (What); 4) other events that may be triggered during the activity execution (What); 5) the organizational role that is going to be in charge for the activity completion (Who – the human or software agent); 6) the business objective used to measure and analyze the activity in order to find out ways for optimization. All these basic elements may be analyzed from different perspectives, depending on the number of stakeholders taken into account: business, information, application, cost, time, rules etc.

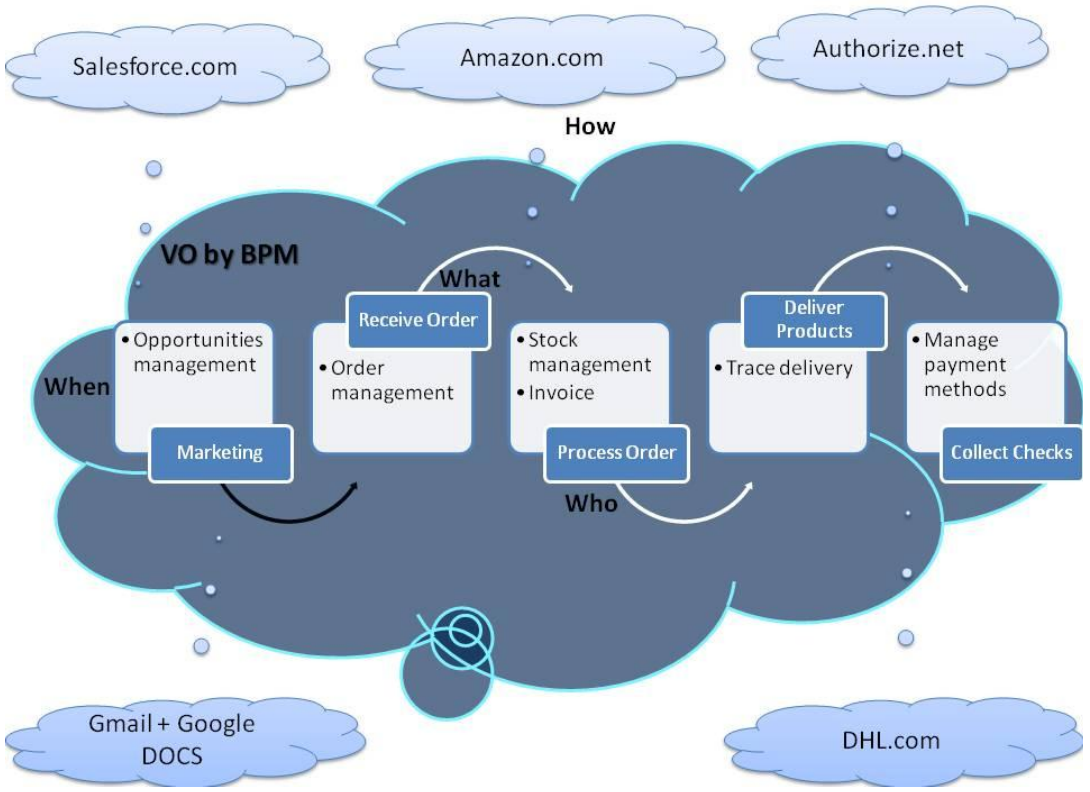

Fig. 1. The Cloud-based Virtual Organization as a business process running Cloud Services

In order to design each of the Cloud-based VO activities, we will take into consideration the principle of views and viewpoints separation in the software architecture development, recommended by ISO/IEC 42010:2007 - Recommended Practice for Architectural Description of Software-intensive Systems [36]. SO/IEC 42010:2007 addresses the activities of the creation, analysis and sustainment of architectures of software-intensive systems, and the recording of such architectures in terms of architectural descriptions. ISO/IEC 42010:2007 establishes a conceptual framework for architectural description and defines the content of an architectural description. It specifies requirements on the contents of an architecture description. An architecture description (AD) expresses the architecture of a system. An AD is a document, repository or collection of artifacts used to define and document architectures. According to this standard, every system is considered in the context of its environment. The environment of a system is understood through the identification of the stakeholders (e.g. client for the system, users, operators, developers, suppliers, regulators) of the system and their system concerns (e.g. data structure, behavior, data access, control, cost, safety, security). Identifying the stakeholders and concerns helps the architect to get a detailed understanding of the context in which the system must be developed, used and operated. In order to take into consideration both the stakeholders and the many concerns of a system, the standard introduces two fundamental basic constructs of the system's architecture: viewpoints and views.

A *viewpoint* is a way of looking at a system. A viewpoint captures the conventions for constructing, interpreting and analyzing a particular kind of view. Viewpoint conventions include languages, notations, model types, modeling methods, analysis techniques, design rules and any associated methods.

A *view* is what you see when looking from the chosen viewpoint. A view is a collection of models representing the architecture of the whole system relative to a set of architectural concerns. Separation of concerns is a useful technique for managing complexity. A view is part of a particular architecture description for a system of interest. For example, a structural view of a system might include a model showing components and their interfaces and a model of their dependencies and inheritance relationships. A performance view might consist of models for resource utilization, timing schedules and cause-effect diagrams. The idea of a view is that it addresses a specific set of concerns about a system using well-defined notations and models.

Using this approach, we can describe an activity by filling in all the cells of a matrix like the one shown in Table 1. The matrix is using the views (columns) and viewpoints (rows) described above as core elements in Cloud-based VO engineering, but each designer (VO broker) may define his own extensions.

Table 1. Views and viewpoints for Cloud-based VO activities

	When (Event)	**How (Service)**	**What (Structure)**	**Who (Role)**	**Objective**
Viewpoints	The event that triggers the activity	The service that will be executed by the activity	The information structure(s) that will be delivered once the activity finishes	Organization roles accountable for the activity	The declared business objective for this activity
Business	Event name and description	The description of the service that will do the job (manual, semi-automated, automated)	The business document(s)	The role name	Business objective (cost, time, profit)
Information	Event structure	IN/OUT parameters	The business document formal structure (UML, XML, DER)	The user description	Logging info structure and location
Application	The actual service that takes the event and dispatches it to the enterprise system (usually, the ESB)	The service - fully automated. If the activity is semi-automated or manual, the service that will show the GUI to the user	The service that will deal with the persistency transaction	The authentication system	Logging & monitoring services

Cost	Event processing and storage costs	Cost per service execution	Information storage & management costs	The human costs	Computed cost of the activity
Time	Arriving time	Execution time	Time when transaction ended	Time when documents have been signed	Computed time till the activity completion
Rules	To apply to the event	To apply to service execution (pre and post-processing)	To apply to the structure	Automation rules	Rules to instruct the objective computation

5 A framework for Cloud-based virtual organization engineering

The framework for Cloud-based virtual organization engineering (CVOE) is intended to promote a cohesive approach which considers a process view of information processing within the context of the entire virtual organizational operational environment. This conceptual framework innovatively combines ISO/IEC 42010:2007 recommendations with a number of software architectures, development principles and design patterns in order to provide the highest possible flexibility for the dynamic reconfiguration of resources used by a certain instance of a Cloud-VO. The main focus is on: 1) views reuse for multiple VOs; 2) simulate the process and analyze the costs with respect to business objectives.

In order to achieve these goals, the framework takes the REST (Representational State Transfer – a well-known architectural style) principles and applies them to the Cloud-VO engineering by declaring that every view used to describe an activity is a *Resource*. As a consequence, we may say that the main components of a Cloud-VO are: activities, resources and business rules. Each activity uses resources of various types. For example, each activity is supposed to execute a service and is supposed to be managed by a certain organizational role. Following the vision of our framework, both the service and the role are resources that may be located anywhere in the Clouds. Resources are organized in Resource Catalogues. Each entry in a Resource Catalogue has a globally unique identifier given by its own URI and needs to be further described in detail based on the declared architectural viewpoints of the Cloud-VO. The viewpoints will actually become the columns in these catalogues (see Table 2). By creating a repository of catalogues (Table 2), the VO designer you will quickly discover that the complexity can be easily managed by filling each cell of this repository with URIs. Also, the designer may add any new viewpoint as a new column (cost, for example).

Table 2. Resource Catalogues used to describe the resources involved in Cloud VO activities

		Viewpoints		
Name	**Description**	**Business (human readable)**	**Information (machine readable)**	**Application (execution environment)**
Business Services Catalog (BSC)	Services the VO can use	URI to the service description (e.g. Wiki-page)	URI to IN/OUT parameters' structure (WSDL, XML)	URI to the service
Business Documents Catalog (BDC)	Document templates	URI to the BD description	URI to the template	URI to the persistence service
Business Events Catalog (BEC)	List of the events that the VO responds to	URI to the Event description	URI to the event Structure	URI to the ESB
Roles Catalog (RC)	The VO structure	URI to the description of the role	URI to the formal description	URI to the authentication service
Business Processes catalog (BPC)	Business processes	URI to the BP description (e.g. BPMN)	URI to the BP formal description (BPEL, JPDL)	URI to the service that will actually execute the business process
Objectives Catalog (OC)	Describe the VO's objectives	URI to the description	URI to the formal properties	URI to the service that will monitor the activities

Once everything is treated as a resource, the business rules expressions associated to different views of each activity may utilize the URIs and thus opening a new range of opportunities for business rules to be declared by business people using DSL (Domain Specific Language) statements based on resource names found in various catalogs. The Cloud-VO can thus be designed using the Resource Allocation Matrix (VO-RAM) found in Table 3. For each activity we will have two rows: the resource URIs and the business rules statements using those URIs.

Table 3. Resource Allocation Matrix for the Cloud VO

Viewpoints	**When (Event)**	**How (Service)**	**What (Structure)**	**Who (Role)**	**Objective**
Business	URI (from BEC)	URI (from BSC)	URI (from BDC)	URI(from RC)	URI (from OC)
Rules	Rule expressions using resource names from various catalogs. The business process engine running the VO will translate these rules based on the associated URIs.				

The main sequence that will be executed at runtime by the VO's internal business process engine would take the form of:

```
When Event then
PUT(Where, EXECUTE (How (What) ))
APPROVE(Where, Who)
```

Starting from the above mentioned principles, the CVOE framework proposes a VO development life-cycle made of the following activities:
1) create resource catalogs;
2) create the Cloud-VO business process;
3) create the resource allocation matrix;
4) simulate the Cloud-VO;
5) generate the Cloud-VO physical definition;
6) run the Cloud-VO instances.
Table 4 shows an example of the VO-RAM for the first activity of the VO exemplified earlier in the paper (see also figure 1). In this example we use the resources' URI directly together with specific expressions used to communicate with the process engine in order to read/write context variables. The whole set of expressions that can be used in VO-RAM will form the VO's expression language. Indeed an expression language will be needed if we want to avoid the overhead of calling VO's internal business process execution services by their externally accessible URIs.
By replacing resources' URIs with their corresponding catalogue URI, we can settle the basis for a technique that will provide the reusability of resource descriptions. This way, the VO designer (broker) will be able to add as many viewpoints as needed to the catalog description without the need to alter the VO-RAM.

Table 4. Resource Allocation Matrix example

	When	How	What	Where	Who
Marketing	http://abc.com.events#StartMktCampaign	http://salesforce.com/services/234RFDS5454	http://abc.com/res/offer	${sentOffers}	http://abc.com/users#John
Receive order	http://abc.com.events#OrderEventEmail	http://abc.com/services/orderReceived	${orderByEmail}, ${offer}	http://google.com/spreadsheets?key=2131FGD	http://abc.com/users#Doe
Rules	When ${orderByE-mail}.customer.name ="Client A" Then Notify http://eolcloud.com/users#Mark				

6 Related work

To our knowledge, there is no similar work regarding virtual organization engineering framework based on Cloud resources taking into account the concept of reusing the multiple views and viewpoints needed to satisfy all the stakeholders' perspectives.
As we have seen earlier in this paper, there are a number of frameworks addressing creation and management of a VO. By summarizing the literature, we can identify three general models: a) frameworks based on agents and rules [21], [22]; b) frameworks based on services and SOA [27], [30]; c) frameworks that focus on Semantic Web technologies to create the required collaboration environment [28], [29]. All these frameworks address the problem of identifying the optimal model of coordination for the autonomous services/agents and the management of semantic agreements within the VO contextual environment. Quite the opposite, our framework takes into consideration the whole complexity of a VO system based on Cloud services.
There are also well known frameworks focusing on enterprise engineering in general, like Zachman, ArchiMate, TOGAF, eTOm. However, all of them do impose rigid struc-

tures on the architecture and are not suitable for MDD (Model Driven Development) automation process From this point of view our framework has taken the abstraction process to the next level: the designer may define his own views and viewpoints and still be able to apply MDD techniques to obtain its running VO system based on resource URIs and descriptions found in Resource Catalogues.

7 Conclusions and future work

The approach introduced by this paper makes very easy for designers to reuse resources and different descriptions associated with different viewpoints. It also allows simulating and dynamically modifying different Cloud-VO configurations in order to identify the optimal configuration of resources used for Cloud-VO activities. As future work we intend to formally define the VO-RAM expression language and to create the architecture needed for the VO-RAM to be embeddable as a plug-in tool into various workflow/business process engines

References

[1] J. Patel, et al, "Agent-based virtual organizations for the Grid", in *Proc. The fourth international joint conference on autonomous agents and multiagent systems*, The Netherlands, July 25-29, 2005, DOI=10.1145/1082473.1082668

[2] G. Frackowiak, et al, "Adaptability in an Agent-Based Virtual Organization – Towards Implementation", *Web Information Systems And Technologies, Lecture Notes in Business Information Processing*, vol. 18, Part 1, 2009, pp. 27-39

[3] R. Nagel and R. Dove, *21st century manufacturing enterprise strategy*, Bethlehem- PA: Iacocca Institute, 1991

[4] B. Ghilic-Micu, and M. Stoica, "Virtual Organization – Cybernetic Economic System. Modeling Partner Selection Process", *Economic Computation and Economic Cybernetics Studies and Research*, vol. 043, no 02, 2009

[5] P. Grefen, et al, "Internet-Based Support for Process-Oriented Instant Virtual Enterprises", *IEEE Internet Computing*, vol. 13, no. 6, 2009, pp. 65-73

[6] D. Radoiu, "Virtual organizations conceptual modeling". In: *Studia univ. Babes Bolyai, Informatica*, vol. LIII, no. 1, 2008

[7] L. Creţu, "InformationTechnology for Organization (re)Design (3)", *Revista de Informatică Economică*, Bucuresti, vol. 33, no. 1, 2005

[8] L. Camarinha-Matos and H. Afsarmanesh, "Creation of virtual organizations in a breeding environment", in *Proc. INCOM'06*, St. Etienne, France, 17-19 May, 2006

[9] L. Zhang and Q. Zhou, "CCOA: Cloud Computing Open Architecture", in *Proc. IEEE 7th International Conference on Web Services (ICWS 2009)*, Los Angeles, CA, USA, July 6-10, 2009, pp.607-616

[10] R. Miles and C. Snow, "Organizations: New Concepts for New Forms", *California Management Review* XXVIII(3), 1986, pp. 62-73

[11] W. Davidow and S.M. Malone, *The Virtual Corporation. Structuring and Revitalizing the Corporation for 21st Century*, HarperBusiness Publishing, 1993

[12] M. Hammer and J. Champy, *Reengineering the corporation. A manifesto for business revolution*. Nicholas Breadley Publishing, 1993

[13] R. Adam, "Întreprinderile virtuale versus întreprinderile tradiţionale", Revista Informatica Economică, vol. 29, no. 1, 2004

[14] M. E. Porter, *Competitive advantage - Creating and sustaining superior performance*. New York: The Free Press, 1985

[15] J. Galbraith, *Organization Design*, Addison-Wesley, 1977, pp.3

[16] S. L. Goldman and R. N. Nagel, *Agile Competitors and Virtual Organizations - Strategies for Enriching the Customer*.

New York: Van Nostrand Reinhold, 1995

[17] A. Mowshowitz, "The Switching Principle in Virtual Organization", in *Organizational Virtualness and Electronic Commerce Proceedings of the 2nd International VoNet – Workshop,* Zurich, September 23-24, 1999

[18] O. E Williamson, *Markets and Hierarchies: Analysis and Antitrust Implication*. New York: Free Press, 1975

[19] B. Katzy, C. Zhang and H. Löh, "Reference Models for Virtual Organizations", in L. M. Camarinha-Matos, H. Afsarmanesh and M. Ollus (eds) *Virtual Organizations – Systems and Practices*, Springer Science +Business Media, Inc., 2005

[20] C. Castelfranchi, "Engineering social order", in Omicini A, Tolksdorf R, Zambonelli F (eds.) *Engineering societies in the agents world*, Springer, 2000, pp 1–18

[21] J. McGinnis, K. Stathis, F. Toni, "A Formal Framework of Virtual Organisations as Agent Societies", in Formal Aspects of Virtual Organisations 2009 (FAVO2009), Eindhoven, The Netherlands, 3 November 2009, pp. 1–14, doi:10.4204/EPTCS.16.1\

[22] J. Pitt, L. Kamara, J. Sergot and A. Artikis, "Formalization of a voting protocol for virtual organizations", in F. Dignum, V. Dignum, S. Koenig, S. Kraus, M. P. Singh and M Wooldridge, (eds) *The 4th International Joint Conference on Autonomous Agents and Multiagent Systems* (AAMAS - 2005). ACM, 2005, pp. 373–380

[23] Y. Udupi and M. P. Singh, "Contract Enactment in Virtual Organizations: A Commitment -Based Approach", in: AAAI'06: Proceedings of the 21st National Conference on Artificial intelligence. AAAI Press, Boston, Massachusetts, 2006, pp. 722–727

[24] C. Sierra, J. Thangarajah, L. Padgham and M. Winikoff, "Designing Institutional Multi-Agent Systems", *Agent-Oriented Software Engineering VII,* LNCS 4405, 2007, pp. 84–103

[25] H. Lopes Cardoso and E. Oliveira, "Electronic institutions for B2B: dynamic normative environments", *Artificial Intelligence and Law* vol 16, no. 1, 2008, pp. 107–128.

[26] J. Va´zquez-Salceda, H. Aldewereld and F. Dignum, "Implementing norms in multiagent systems", in Lindemann G, Enzinger J, Timm IJ, Unland R (eds) *Multiagent system technologies*. Springer, 2004, pp 313–327

[27] M. Dolenc, "Semantic Grid Platform in Support of Engineering Virtual Organisations", *Informatica,* 32, 2008, pp 39-49

[28] E. Del Val, "THOMAS: A Service-Oriented Framework For Virtual Organizations", in *Proceedings of 9th International Conference on Autonomous Agents and Multiagent Systems* (AAMAS 2010), Toronto, Canada, May, 10–14, 2010,

[29] L. Zhou, H. Chen, Y. Mao, "A Semantic-based Framework for Virtual Organization Management", in: *The Third ChinaGrid Annual Conference (chinagrid 2008)*, 2008, pp. 294-299

[30] K. Boukadi, L. Vincent and C. Ghedira, "Multi-layer Framework for Virtual Organizations Creation in Breeding Environment", in *PRO-VE 2010, International Federation for Information Processing*, Springer, Boston, 2010, pp. 287 – 296.

[31] F. Chong and G. Carraro (2006), *Building Distributed Applications - Architecture Strategies for Catching the Long Tail*. Available on-line: http://msdn.microsoft.com/en-us/library/aa479069.aspx

[32] B. Orriens, J. Yang and M. Papazoglou, "A Rule Driven Approach for Developing Adaptive Service Oriented Business Collaboration", in *Proc of the IEEE International Conference on Services Computing*, Chicago, USA, Sept.

18-22, 2006, DOI= 10.1109/SCC.2006.14

[33] S. Reid (2008), "Forrester's SaaS Maturity Model. Transforming vendor strategy while managing customer expectations", *Forester research*, available: http://www.forrester.com/rb/Research/forresters_saas_maturity_model/q/id/46817/t/2

[34] S. Robbins (2008), "SaaS Architecture Maturity Model", *InfoQ*, available: http://www.infoq.com/news/2008/02/saas-architecture-maturity-model

[35] D. Amrhein and S. Quint (2009), "Cloud computing for the enterprise. Part 1: Capturing the Cloud", *IBM DeveloperWorks*, April 2009, available: http://www.ibm.com/developerbsphere/techjournal/0904_amrhein/0904_amrhein.html

[36] IEEE, *ISO/IEC 42010:2007 - Recommended Practice for Architectural Description of Software-intensive Systems*, available: http://www.iso-architecture.org/ieee-1471/

Liviu Gabriel CRETU, PhD, holds a permanent position of lecturer at Faculty of Economics and Business Administration. He has more than fifteen years of practice, research and teaching in the area of information management and information systems. He has written three books and has co-authored other three, he published over thirty articles in various journals and conference proceedings, he was a member of three national-level consortium-research projects and he was editor in chief of an international on-line journal for two years. He was also actively involved in various projects with private software companies spread over several countries in Europe, as well as with the European Commission. His research interests are mainly related to: enterprise engineering, enterprise systems architecture and design for SaaS, business process management, business rules engines, messaging systems, Cloud Computing and Semantic Web.