The Principle of Immanence in Event-Based Distributed Systems

Pascal Dugénie, Stefano A. Cerri

The principle of immanence in event-based distributed systems

Pascal Dugenie, Stefano A. Cerri

Abstract This chapter focuses on the principle of *immanence* for autonomic event-based distributed systems such as collaborative environments on the GRID. On one hand, GRID provides a sound infrastructure for coordinating distributed computing resources and Virtual Organisations (VO). On one other hand, *immanence* is a principle that emerges from the internal behaviour of complex systems such as social organisations. Although several existing VO models specify how to manage resources, security policies and communities of users, none of them has considered mechanisms that reflect the internal activity to constantly improve the overall system organisation. The AGORA model, proposed in 2004, has been integrated in an experimental collaborative environment platform. After several years of experimentation with communities of scientists from various domains, The AGORA architecture has been enhanced with a novel design approach for VO management. The model is a dynamic system in which the result of interactions are fed back into the system structure. The basic idea is to specify a set of mechanisms to catalyse the collective intelligence of active communities in order to enable a self-organisation of the collaborative environments.

1 Introduction

1.1 The principle of immanence

Usually, *immanence* refers to philosophical and metaphysical theories, often related to religious doctrins. The general idea behind the notion of *immanence* is that the cause of the development of an object occurs inside this object. This approach

CNRS, Centre National de Recherche Scientifique
LIRMM, Laboratoire d'Informatique de Robotique et de Microelectronique de Montpellier
161 rue Ada, 34392 Montpellier Cedex 5 France
dugenie@lirmm.fr, cerri@lirrm.fr

presents much interest in complex systems theory to explain how the flow of events inside a system and its activity may engender the emergence and an organisation that is continuously in self-adapting. A living body is a typical example of such immanent system. The paradigm of complex system theory originally appeared in the domain of biology [30] and has been adapted later in the domain of social theory [23]. The term autopoiesis, coined in cognitive biology by *Maturana* and *Varela* [24], was also used to express that a complex system maintains its distinctive identity by constantly considering in its communication what is meaningful and what is not.

According to the sociologist *Niklas Luhmann*, social systems are autopoietically closed in the sense that they use and rely on resources from their environment. The principle of immanence in a social organisation emerges from this process of selection of elements in the system filtered from an over-complex environment. *Niklas Luhmann* considers mutual confidence between actors as the main factor to reduce the complexity of the system [23]. In summary, this literature suggests the existence of a strong interdependance between the *organisation* and the *activity* of a complex system: *the organisation of a social system is immanent to the activity within that system.* A circular causality[1] exists between the organisation and the activity of a system: the organisation enables the generation of activity, meanwhile the activity constantly seeks to improve the organisation. The effect of immanence is the living link between the organisation (*i.e* the static part) and the activity (*i.e:* the dynamic part) of the system. In contrast, a system whose behaviour would be completely determined from initial conditions with no feedback effect of its activity on its own structure is not an immanent sytem and has no chance to be self adaptive in case of changes of conditions of its environment.

1.2 Immanence on the Web

Talking about *immanence* in the domain of informatics tended to appear quite utopic only a few years ago. Since then, *Pierre Levy* has introduced the notion of immanence by depicting the Web as a common infrastructure (*i.e* the material part) that is immanent to a global collective intelligence (*i.e* the immaterial part) [22]. This dichotomy has been exacerbed with the recent emergence of social networks in the Web 2.0. This is one illustration of the many new forms of social interactions in collaborative environments. A social network on the Web continuously behaves to adapt its own structure throughout its internal activity. The structure is composed of a complex network of communication channels while the activity consists of the numerous interactions passing throughout these communication channels.

Nowadays, several factors encourage much interest for modelling collective behaviour: the rise of the holistic modelling approach inherited from research in

[1] A circular causality means that the effects cannot be separated from their causes.

MAS[2], or the maturity of distributed computing technologies, especially the GRID. Agents may share their knowledge and expertise, while the GRID provides computing resources and various modalities of communication. Immanence occurs as a side effect in most collaborative situations. In the case of collaborative environment operating in a GRID infrastructure, actors may act alternatively as system designers as well as users. A computing element of GRID may exchange services with the actors in the form of a given result to a given request. Similarly an actor may provide a service in the form of an answer to a question based on a particular competence. The GRID aims to organize all these kinds of heterogeneous services. The GRID provides mechanisms for instantiating services within its infrastructure, with a proactive behaviour. Thus the activity of the system, represented by the aggregation of all interaction of the GRID services, generates a logical form of organisation. One major distinction between the GRID over the Web, is the possibility for the GRID to deploy stateful resources necessary to operate autonomous services.

In this approach, interactions between services are contextualised within a generic collaborative process composed of both humans and artificial processes. The notion of agent is extended in the literature to cover both types of service producers (i.e humans and artificial processes) [2, 17]. Agents interact within a collaborative environment by providing or using services. One essential condition for a collaborative environment to become immanent is that any agent of the system may always play an active role in the elaboration of the organisation [8, 5, 6, 7]. For instance, both system designers and expert-users may feedback their experience in the cycle of development and validation of a complex application. They interact by providing services to each other via a common collaboration kernel. They may develop their point of view in the context of a collaboration process and their role may evolve if necessary. Without a flexible self-adaptiveness, such a system would not operate efficiently.

A collaborative environment on a WEB or a GRID infrastructure is a typical case of event-based distributed systems composed of social links, where events are the interactions between agents.

In order to interconnect concepts related to event-based distributed systems and the behaviour of a collaborative environment, this chapter briefly reviews, in Section 2, the state-of-the-art of virtual collaboratives environments and the deployment aspects of collaborative services on the GRID.

Through the description of the AGORA model, the Section 3 presents four interaction mechanisms which contribute to specify the principle of immanence.

[2] MAS: Multi-Agent Systems

2 Background

The scope of this chapter is in the intersection between three distinct research domains:

- the domain of CSCW[3] that explores concepts related to VCE[4],
- the domain of GRID that specifies a kind of SOA[5] and management models for VO (Virtual Organisations),
- the domain of MAS that analyses collective behaviour in distributed computing systems.

2.1 Virtual Collaborative Environments on GRID

Virtual Collaborative Environments (VCE) have emerged along with event-based distributed computing systems. A VCE aggregates resources, services and interfaces to provide a dedicated environment for collaboration. These interfaces may support several modalities of communication such as audio-video, shared visualisation, instant messagging, notification and shared file repositories. GRID is a pervasive technology allowing seamless access to distributed computing resources. A VCE on the GRID is an ubiquitous system[6] composed of stateless terminal elements where all computing resources are delivered by the infrastructure [12, 28, 20]. GRID has the capability to maintain the state of the communications independently from the type or the location of the terminal elements.

Access Grid[7] (AG) is the largest deployed GRID VCE solution world-wide. AG operates on *Globus* [11], the most popular GRID middleware. The topology of the AG infrastructure consists of two kinds of nodes: the venue clients and the venue servers [**?**]. AG venue clients can meet in a venue server to set up a meeting. AG uses the H.263 protocol [16] for audio and video encoding and a multicast method to distribute the communication flow between sites. The display from multiple H.263 cameras in every site gives a strong feeling of presence from every other site. The modular characteristic of AG allows the addition of new features such as application sharing (shared desktop, presentation, etc.) and data sharing. AG focusses on the principles of *awareness* and *ubiquity*. However, AG does not include a powerful means for VO management. VO are managed in an *ad hoc* manner at the venue server side. This requires much technical administrative work from computer experts in this domain.

---

[3] CSCW: Computer Suported Collaborative Work

[4] VCE: Virtual Collaborative Environment

[5] SOA: Service-Oriented Architecture

[6] an ubiquitous system enables the access to the VCE from anywhere at anytime.

[7] Access Grid: www.accessgrid.org

2.2 VO management models

In its original definition, a VO is a community of users and a collection of virtual resources that form a coherent entity with its own policies of management and security [15]. A rudimentary VO management system has been originally built-in *Globus* but has little potential for scalability. In order to resolve these limitations several VO management models have been proposed within the GRID community.

CAS (Community Authorization Service) has been specifically designed to facilitate the management of large VOs [26]. The functionalities for VO membership and rights management are centralised in a LDAP[8] directory. Since the structure of the VO is strongly hierarchichal, it is difficult to reorganise the initial tree once the services are deployed.

VOMS (VO Membership Service) is deployed in more recent GRID infrastructures such as EGEE [27]. It resolves some of the problems of CAS, such as the membership management by providing a more flexible relational database instead of a flat tree structure. For instance, a database has the possibility to specify complex, evolving relations between concepts such as users, groups and rights. The user management in a tree such as LDAP is not easily evolving because the concepts are embedded into the structure of the tree. However, VOMS still presents some conceptual limitations such as an inheritance link between a parent VO and its children. The subdivision of VO into groups often creates confusion in the management of rights and does not enable a complete independence between the groups and the VO. For instance, the lifetime of a group is determined by the lifetime of the parent VO.

2.3 GRID and MAS convergence

The convergence of GRID and MAS research activities has been claimed in 2004 [13] as a major research objective that sustained in the early years 2000 [3, 4]. Both domains share an approach based on SOA which enables to abstract the underlying technology behind a common concept: the *service*. Since then, a sustained research activity has attempted to formalise the mapping of GRID and MAS concepts [19, 18]. This work opens new perspectives to specify immanent systems.

As established in the OGSA (Open GRID Service Architecture) framework for resource and security management, GRID is a kind of SOA [14]. MAS focusses on complex organisation models far more advanced than the GRID ones. The well-known MAS conceptual model Agent-Group-Role (AGR) emphasises on the characteristic of flexibility required for a self-organised organisation model [10]. Further to the suggestion for convergence between GRID and MAS concepts their integration has required an extensive effort with AGIL (Agent-GRID Integration Language) [19, 18] to formalise the organisation and the interactions of distributed resources

[8] Lightweight Directory Access Protocol

and agents and propose the service as a pivot concept. AGIL has been originally proposed in 2006 as an ontology [9], then has been developed more recently as a language. The abstraction of GRID and MAS concepts allows the development of complex architectures. The AGORA model is an example of such architecture represented with AGIL concepts.

2.4 Discussion

Existing VCE solutions such as AG are usually studied in the CSCW research domain which focuses on the *collaboration processes* aspects. The aspect of *VO management* such as CAS or VOMS is studied within GRID research communities. In order to efficiently identify the impact of the principle of immanence, many issues of these two complementary aspects would be better understood if they were combined rather than studied in separate domains.

Moreover, a SOA is clearly a convenient approach for self-adaptive systems. Models based on classical client-server architecture often present a lack of flexibility resulting in restrictions in their evolution. Technological choices are adopted more or less arbitrarily by the designer of the architecture to respond to an initial need. However as soon as the need changes, the architecture must follow a different pattern. For example, identity management is clearly difficult to specify at the beginning of the life of a system. Many choices often result from technical limitations. Some architectures adopt right-centered management of groups where the members of a given group benefit of a set of rights over a service (*e.g* a group that would include all moderators of a service). Typically, this approach as used in AG does not include a powerful mean for VO management.

3 About AGORA

AGORA is a VCE architecture model which exhibits altogether the principles of *immanence*, *ubiquity* and *awareness*, in order to resolve the limitations described in the previous section. The development began in 2004 in the context of the european project ELeGI[9]. Initially, the challenge consisted of specifying a generic VCE on GRID that enable various kinds of communities of scientists to freely collaborate while minimizing the intervention of software specialists. Later the rationale of this project focused on the question of self-organisation of the communities as this aspect presented a growing interest in the domain of workflow management and concurrent access to distributed resources.

Beside *immanence*, *ubiquity* and *awareness* are the two other key principles identified that outlined the future AGORA platform. *Ubiquity* in computing science

[9] ELeGI (European Learning Grid Infrastructure)[25, 29], European Framework Program n6 on IST (Information Services and Technologies), 2004-2007

means that the location of resources is independent of the conditions of temporal and spatial use. In other words, *ubiquity* enables access to the VCE from anywhere at anytime. This implies that the resource used by the VCE has to be provided exclusively by the infrastructure and not from the terminal equipment side. The resource includes capacities of memory, processing and transmission as well as access to a large and heterogeneous instrumentation. Although this principle has been envisaged several years ago [31], the concrete deployment of operational solutions has been feasible only recently by means of pervasive technologies such as GRID. The principle of *awareness* indicates that members of a community seem to be more present to each other. The usual way to enhance the presence on the Web is to choose various modalities of communication with synchronous services.

The experimental VCE platform called AGORA UCS (Ubiquitous Collaborative Space) is the unique solution that proposes a fully integrated range of asynchronous and synchronous services. AGORA UCS includes a pool of resources accessible asynchronously and a range of services to operate on these resources concurrently and synchronously. Each member of a community may access to these resources via a viewer desktop coupled with a shared service for audio visual communication with other members. AGORA UCS is also a proof-of-concept for an architecture based on user-centric requirements that have been identified by specialists of cognitive science[1].

Extensive experiments of the AGORA UCS prototype have been performed with more than eighty users accross the world [5] and about twenty communities in various scientific domains (organic chemistry, earth studies, optical physics, microelectronics). For the last two years, experiments have been extended to social and human sciences in order to improve the mechanisms underlying the immanence principle.

The AGORA architecture is composed of:

- a conceptual model including five concepts linked by four relations,
- a set of six persistent core services to manage the collaborative environment,
- four protocol mechanisms aiming for ensuring self-organisation of the communities.

3.1 AGORA *conceptual model*

3.1.1 Overview

The AGORA conceptual model presented on Figure 1 consists of a set of five concepts and four relations. This model includes a generator of events that are originated by humans or artificial processes. The organisation of agents in groups specifies how the flow of actions and the access to the resources are scheduled.

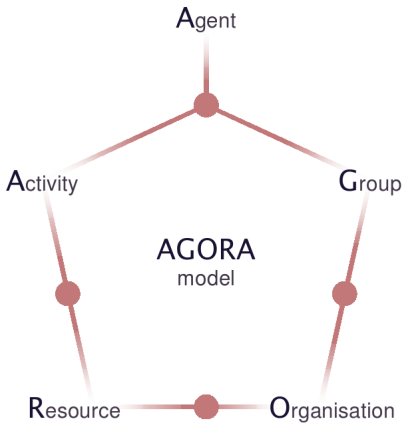

Fig. 1 The AGORA conceptual model

3.1.2 Definitions of AGORA concepts

1. **Agent** constitutes the active component of the system since every action performed by an agent has an impact on the state of the system. In a collaborative context and from the system point of view, all these actions may be performed artificially or by humans. For this reason, the meaning of agent in AGORA is clearly different than user or member. The user is one of the two kinds of agents: the human user and the artificial process. An agent holds its own identity outside the context of a community. An agent belonging to a community becomes a member of that community. A member specifies the relation (the membership relation) between an agent and a community.
2. **Group** or **Community** is a subset of agents. An agent may be a member of several communities. Once it is member, agents may undertake an activity in the context of the community.
3. **Organisation** is a composition of one given group and one unique set of resources. This concept is analogous to **VO** in GRID terminology (OGSA standardisation).
4. **Resource** is a set of means to carry out tasks, having the capacity of memory, processing, broadcasting, and access to a variety of instrumentation. This concept has been formalised as a **service container** by the GRID (OGSA).
5. **Activity** describes the flow of events and interactions within a community. Activities are always carried out by agents in the context of a community. This

involves the notions of authorisations since any activity requires resource for completing a series of operations.

3.1.3 Definition of AGORA relations

1. **Agent** - **Group** - **Activity** is a ternary relation that links an agent to a community and to a number of activities according to the context of that community.
2. **Activity** - **Resource** indicates that any activity is attached to some resource. An activity always needs various kinds of resources like processing power, storage facilities or access to peripherials (network or specific instruments).
3. **Group** - **Organization** indicates that a community is attached to a proper organization. The bijective nature of this relation expresses the fact that an organization is unique for a given community.
4. **Resource** - **Organization** indicates that a set of resources is attached to a unique organization. This relation is symetrically bijective: a resource constitutes one of the two parts of an organization, the second one being the community.

Usually, existing VO management solutions specify two binary relations for organising agents into communities. A first binary relation corresponds to the agent membership to the community (e.g the agent A is a member of the community C). A second binary relation corresponds to the level of authorisations (sometimes called the role) of this agent (e.g A is *moderator* or A is *user*, etc.). However, in many situations it becomes complex to manage relations of pairs that mix three concepts, in particular when agents play different roles in several communities. Also, a role does not have a universal significance. A *moderator* may have a level of authorisation in one community that is different than in another. The ternary relation proposed in the AGORA conceptual model allows these these conceptual limitations to be overcome by allocating unique identifiers to each set of triplets agent-activity-community. Also, the concept of activity is more appropriate than role to express a series of actions that requires a certain level of authorisation in a given community.

3.2 Persistent Core Services

AGORA includes six PCS (Persistent Core Services) instantiated in the service container of every community. They are necessary for bootstaping and maintaining a collaborative environment. Figure 2 is a representation of a service container based on the AGIL formalism.

1. **A uthorisations**: Members of a community may have a different level of permission on services. This service is in charge of assigning rights to members including the permissions over the PCS.

2. **M** **embers**: A community is composed of members. This PCS manages the description of members, adding or removing members of a community.
3. **C** **ommunity**: A number of properties (identifier, description, etc.) are necessary to describe a community at a metalevel. Also, the creation of a new community must be done in the context of another community. This PCS is in charge of two kinds of operations: intra community operations (modifying the properties of the current community) and extra community operations (instantiating a new community). Obviousely, there is a community instantiated at the initialisation of the system and only dedicated for bootstrapping the first community.
4. **H** **istory**: All data belonging to a community must be stored, maintained and indexed. This PCS is in charge of keeping track of changes, logs of events and also of recording collaboration sessions. This PCS also aggregates the events to provide an evaluation of the frequency of access to the resources, that is used for the mechanism of *implication* (see Section 3.3).
5. **E** **nvironment**: A community may personalise its environment. This environment operates in a service container. This PCS is in charge of adding or removing services (excluding the PCS).
6. **N** **otifications**: Communication between members of a community and services is performed via notifications. This service handles the flow of notifications of every kind of communication and manages the states of the exchanged messages. These notifications are particularly useful for triggering events produced during the different decision processes. For instance, all the interaction mechanisms describes in Section 3.3 require events triggered by community members in response to an initial request.

3.3 Four interaction mechanisms

AGORA aims to establish and maintain a high level of confidence inside the community by adopting protocols based on four basic interaction mechanisms called *cooptation*, *implication*, *delegation* and *habilitation*. The choice of these mechanisms have been adopted further to extensive analyses of various case studies[10]. Thus, they contribute altogether to the principle of immanence by resolving most constrains for self-organizing agents and resources within communities. On the left part of Figure 3 the VO management system determines the overall system organisation. On the middle part, mechanisms for VO members management and, on the right part, the resulting activity are directly dependent on the initial organisation. At this stage, there is no possibility to re-introduce activities back to the system organisation. The bottom part of the figure shows the mechanisms for enabling the immanence principle. For example, an agent *a* which is not yet member of a community *C* may initially be coopted by an agent *b* (already member of *C*). If *a* accepts

[10] The contributions to theses cases studies by Erion Bregu, Houda Mourad, Amel Ghezzaz, Lea Guizol and Ahmed Bendriouech, is greatly acknowledged.

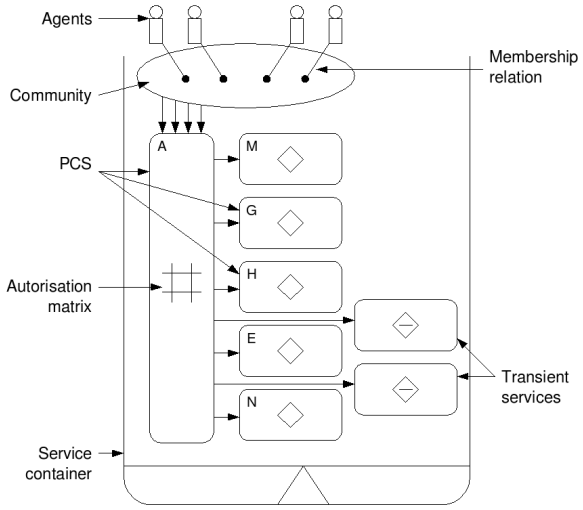

Fig. 2 The six PCS

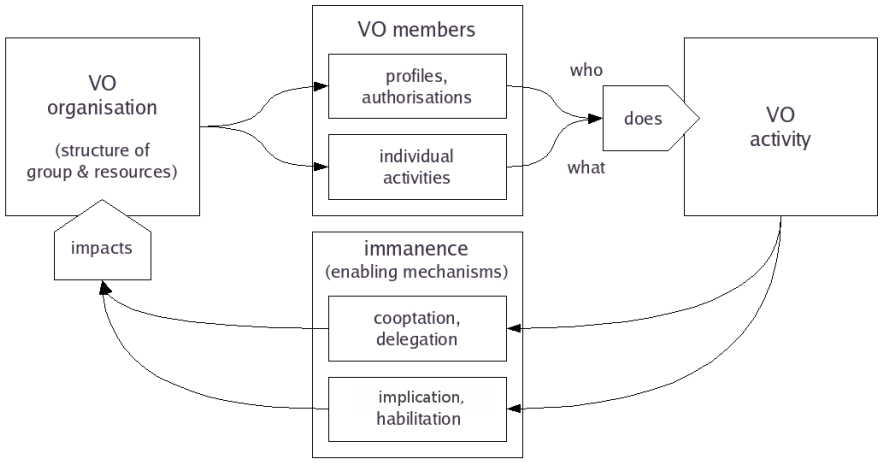

Fig. 3 AGORA's top level VO management model

the invitation from *b*, *b* may decide to delegate a set of rights on services for *a*. Once *a* becomes member of C, other members may decide to extend the habilitation of *a* over more services.

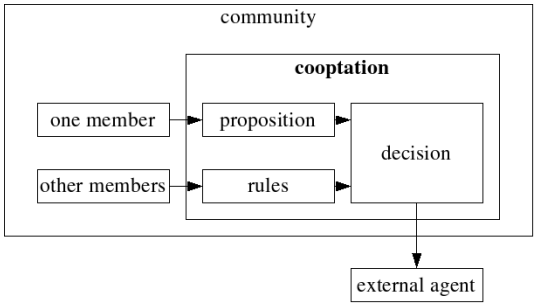

Fig. 4 Flow of events in the mechanism of cooptation

3.3.1 Cooptation

The mechanism of *cooptation* consists of appointment of new members in a community by members who already belong to it. The choice of *cooptation* as a method for members management (the **M** service) is explainable through a well-known exemple of collaborative editing environment such as a wiki. Like in many kinds of collabrative environments, there are usually two methods to introduce new members in a wiki community. The first one is to allow freedom to anyone to become member and allocate default rights to them for editing pages of the community. The second one is to declare a restricted number of moderators who have the capability to validate the requests from external users to become members. Editing rights can be further granted or refused by the moderators. However, none of these two methods is sufficiently flexible to allow a fair workflow process of contributions since the choice of the moderators is established *de facto* and seldom revised. One of the challenges for the **M** service enabling self-organization of the community is to easily add new members while keeping a suitable control of the intentions of the newly added members.

As shown on Figure 4 if coupled with a rules-oriented process, the mechanism of *cooptation* offers a flexible alternative to overcome the constrains of the two methods decribed above. When a community member suggests to invite a new member to this community, the mechanism of *cooptation* may follow rules based on thresholds for deciding if this suggestion is acceptable for the purpose of the community. For example, one condition for validating the addition of a new member in a community could be that at least two members grant this request. Every procedure and thresholds are entirely specified by the community. Obviously, some parameters might be quite different depending on whether the community is large or small and also all this depends on the specificity of the community. In any case, no external rule must be imposed by the system to the community. Once again, the principle of immanence in AGORA lays on this fundamental assumption.

Since the rules and procedures are specified by the community, there is no language that specifies the the syntax of these rules. Although a syntax may be recom-

mended, the most important is to let the entire freedom for the community to specify the language for the rules. Once a community has adopted rules, there is only the need for a translation service that converts humanly understandable rules into a set of directives and parameters that can be interpreted by an artificial process.

3.3.2 Implication

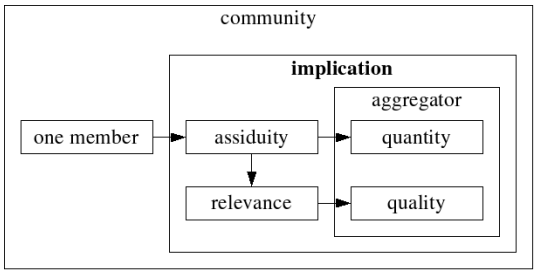

Fig. 5 Flow of events in the mechanism of implication

The mechanism of *implication* provides an estimation of the participation of each member in community activities. The *implication* aims to determine which members are the most involved in a given context and which one adds most value to the community activities. This evaluation must adopt a rating model that satifies the community to keep a high level of motivation of its members by giving them encouragements and reward from their participation. Also, measurement of this evaluation is dynamic. It takes into account quantitative and qualitative aspects, objective and subjective inputs, as well as their sustainibility in time.

Quantitative aspects can be determined by the attendance of members, the frequency and the time spent accessing the resources, or the amount of contributions to activities. These settings are easily calculable because they can be inferred from the history data provided by the **H** service. However, the most significant effect of *implication* can be best obtained from a qualitative evaluation such as assessment of the adequacy of contributions, relevance to the current activities and other subjective parameters such as the reputation of members. This is obtained from other members' feedback.

For this reason, the mechanism of *implication* incorporates a rule-oriented process that converts all inputs (qualitative and quantitative) into a global level of implication (see Figure 5). In the AGORA model, all rules and parameters of calculation of this level of implication can be specified by the community.

3.3.3 Habilitation

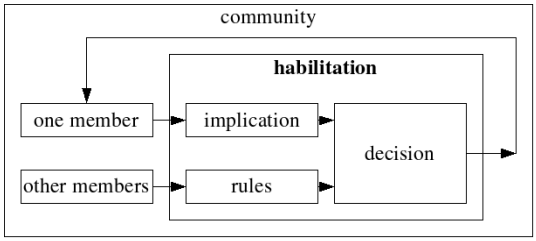

Fig. 6 Flow of events in the mechanism of habilitation

The mechanism of *habilitation* manages the decisions to gradually allocate access rights for community members to services and resources. The goal of *habilitation* is to distribute rights efficiently within communities.

As shown on Figure 6, a decision combines various parameters including the level of implication and the trust that the community has towards this member. The mechanism of habilitation analyzes these parameters to allow members to access services in order to maintain the security and development of the network.

3.3.4 Delegation

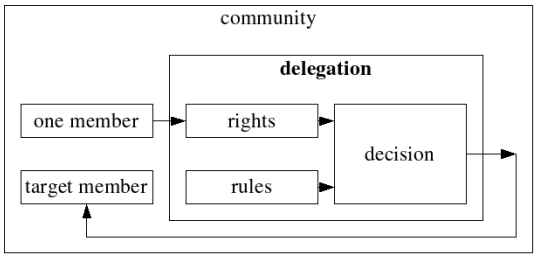

Fig. 7 Flow of events in the mechanism of delegation

The mechanism of delegation consists of allowing one member (the delegator) to transfer part of its rights to another member (the delegate). The major purpose of the delegation is continuity of the activities carried out by the community members who can not always be present to perform their tasks. Delegation may also become useful to motivate members to gain some responsibilities for contributing to the development of the community.

Since the delegate aquires the same responsibilities regarding these tasks, the delegation mechanism must ensure that appropriate rules are specified to prevent risks of failure (see Figure 7). All these rules are specified by the community. The delegation is not permanent, a validity period is specified as well as a condition of revocation at any time.

3.4 Experimentation of AGORA

Extensive series of experiments have been performed since 2005 on the AGORA UCS platform, including about one hundred users and a dozen communities in various scientific domains (chemistry, microelectronics, physics, environment as well as human science). Initially, these experiments focused on the usability issues for any user who is not familiar with concepts of computing science. Access is made possible via a simple a *stateless thin terminal* such as a web browser. The state of the VCE is maintained at the server side by means of a virtual desktop. An instance of the virtual desktop is dedicated as many times as a user is member of a community. The users have realised the importance of ubiquity since they were able to resume their session in the same state even after switching to another terminal. One advantage of this approach is security since no private information or cookies are stored on the client side. Another advantage of this approach is increase of performance since there is no bottleneck at the client side. Every operation is performed at the server side, therefore the computing resources can be more easily scaled. The aspect of awareness using both asynchronous communication via shared editing or synchronous communicatuion using instant messaging and videoconferencing, allowed an immediate bootstrap of a new VO and the acceptance of the technology was extremely high. This enhanced presence enabled a fast transfer of knowlege in particular for mastering complex computational tools.

For instance a scenario called EnCOrE[11] (see Figure 8 has provided the most relevant results [21]. AGORA UCS enabled the visual representation of chemistry models at a distance. Most attention was put by the users on the semantics of their domain rather than solving computing problems. Unskilled users were at ease in their operations. The delegation of rights was important in the absence of some members. The cooptation of new members was also necessary to build a trustful community.

[11] EnCOrE: Encyclopédie de Chimie Organique Electronique.
Demonstration available at http://agora.lirmm.fr

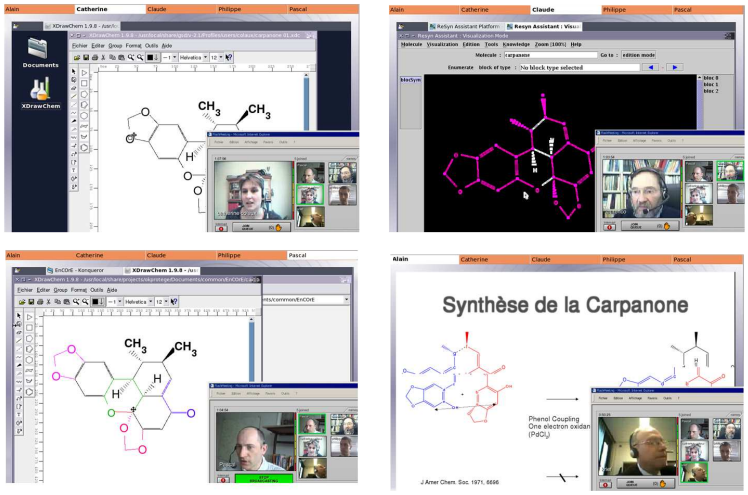

Fig. 8 Experimentation scenario EnCOrE

4 Conclusion

Through the description of the AGORA model, immanence in a event-based distributed system appears to be a sound principle to ensure efficient collaboration processes. The main reason is that the flow of events coming from various nodes of a distributed system does not follow a deterministic pattern but is always seeking for self-adaptation until reaching a suitable level of global satisfaction. The principle of immanence brings a strong framework for identifying risks of conceptual limitations in designing event-based distributed systems.

All promises of the GRID infrastructure opens many significant perspectives and gives more strenght of the AGORA architecture. Yet, an architecture allowing seamless and secure access to distributed resources remains a real technological challenge since the behavior of a community can not be forseen in advance. Flexibility of the AGORA UCS is essential to enable a community to freely organise itself. Various situations of collaboration started naturally by using asynchronous collaborative services (common knowledge base, annotation service), then were reinforced with other modalities of interaction by using a synchronous communication interface facilitate the transfer of knowledge. Discussions in real time, combined with visual representations on a shared desktop, allowed the actors to increase the effectiveness of the collaboration process.

So far, initial experiments have revealed that a user-centric approach is suitable for self-organising communities. The experiments dedicated to validate the mechanisms for immanence started more recently and do not yet cover all the scenarios we are aiming to. At the moment the objective is focusing on user self-ability to feel at

ease in using the environment provided by AGORA UCS and understand the issues related to the self management of the community and the resources provided by the infrastructure. Even at an early stage, this work has already contributed to new ways to approach complex system design where the self-organisation criteria are critical. In many situations, AGORA has achieved results beyond the current state of the art. This progress encourages the belief in a new kind of collaborative system where *organisation* and the *activity* of communities are mutually related.

Glossary of terms and acronyms

AG Access Grid

AGIL Agent-GRID Integration Language

AGR Agent-Group-Role

AGORA UCS AGORA Ubiquitous Collaborative Space is the name of the platform developed for the purpose of experimenting the AGORA principles.

CAS Community Authorization Service

CSCW Computer Suported Collaborative Work

EGEE Enabling GRID for E-Science

ELeGI European Learning Grid Infrastructure

Globus GRID middleware

LDAP Lightweight Directory Access Protocol

MAS Multi-Agent Systems

PCS Persistent Core Services

SOA Service-Oriented Architecture

VO Virtual Organization

VCE Virtual Collaborative Environment

VOMS VO Membership Service

References

1. Sven A. Brueckner and H. Van Dyke Parunak. Engineering self-organizing applications. In *First IEEE International Conference on Self-Adaptive and Self-Organizing Systems, Tutorial session, SASO'07*, Boston, 2007.
2. Stefano A. Cerri. Shifting the focus from control to communication: the STReams OBjects Environments model of communicating agents. In J.A. Padget, editor, *Collaboration between Human and Artificial Societies, Coordination and Agent-Based Distributed Computing, Lecture Notes in Artificial Intelligence*, volume 1624, pages 74–101, Berlin, Germany, 1999. Springer-Verlag.
3. Stefano A. Cerri. Human and Artificial Agent's Conversations on the Grid. In *1st LEGE-WG Workshop: Towards a European Learning Grid Infrastructure*, Lausanne, 2002. Educational Models for Grid Based Services. http://ewic.bcs.org/conferences/2002/1stlege/session3.htm.
4. Stefano A. Cerri. An integrated view of Grid services, Agents and Human Learning. In *Towards the Learning Grid: Advances in Human Learning Services*, pages 41–62, Amsterdam, The Netherlands, 2005. IOS Press. http://portal.acm.org/citation.cfm?id=1563266.1563273.
5. Pascal Dugénie. *UCS, Ubiquitous Collaborative Spaces on an infrastructure of distributed resources*. PhD thesis, University of Montpellier, 2007. In French, to be downloaded at http://www.lirmm.fr/~dugenie/these.
6. Pascal Dugénie. Ubiquitous Collaborative Spaces: a step towards collective intelligence. In *European Conference on Computing and Philosophy, ECAP'08*, Montpellier, June 2008.
7. Pascal Dugénie, Stefano A. Cerri, Philippe Lemoisson, and Abdelkader Gouaich. Agora UCS, Ubiquitous Collaborative Space. In *Intelligent Tutoring Systems, ITS'08*, volume 5091 of *Lecture Notes in Computer Science*, pages 696–698, Heidelberg, 2008. Springer Berlin.
8. Pascal Dugénie, Philippe Lemoisson, Clément Jonquet, Monica Crubézy, and Claude Laurenço. The Grid Shared Desktop: a bootstrapping environment for collaboration. In *Advanced Technology for Learning (ATL), Special issue on Collaborative Learning*, volume 3, pages 241–249, 2006.
9. Frédéric Duvert, Clément Jonquet, Pascal Dugénie, and Stefano A. Cerri. Agent-Grid Integration Ontology. In R. Meersman, Z. Tari, and P. Herrero, editors, *International Workshop on Agents, Web Services and Ontologies Merging, AWeSOMe'06*, volume 4277 of *Lecture Notes in Computer Science*, pages 136–146, Montpellier, France, November 2006. Springer-Verlag.
10. Jacques Ferber, Olivier Gutknecht, and Fabien Michel. From agents to organizations: an organizational view of multi-agent systems. In P. Giorgini, J. P. Müller, and J. Odell, editors, *4th International Workshop on Agent-Oriented Software Engineering, AOSE'03*, volume 2935 of *Lecture Notes in Computer Science*, pages 214–230, Melbourne, Australia, July 2003. Springer-Verlag.
11. Ian Foster. Globus Toolkit Version 4: Software for service-oriented systems. *Journal of Computer Science and Technology*, pages 513–520, 2006.
12. Ian Foster, Joseph Insley, Carl Kesselman, Gregor von Laszewski, and Marcus Thiebaux. Distance visualization: Data exploration on the grid. *IEEE Computer Magazine*, 32(12):36–43, 1999.
13. Ian Foster, Nicholas R. Jennings, and Carl Kesselman. Brain meets brawn: why Grid and agents need each other. In *3rd International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS'04*, 2004.
14. Ian Foster, Carl Kesselman, Jeff Nick, and Steve Tuecke. The physiology of the Grid: an Open Grid Services Architecture for distributed systems integration. In *Open Grid Service Infrastructure WG, Global Grid Forum*. The Globus Alliance, June 2002.
15. Ian Foster, Carl Kesselman, and Steve Tuecke. The anatomy of the Grid: enabling scalable virtual organizations. *Supercomputer Applications*, 15(3):200–222, 2001.
16. ITU-T. H.263, infrastructure of audiovisual services, video coding for low bit rate communication. Technical report, International Telecommunication Union, 2005.
17. Clément Jonquet and Stefano A. Cerri. The STROBE model: Dynamic Service Generation on the GRID. *Applied Artificial Intelligence Journal*, 19:967–1013, 2005.

18. Clément Jonquet, Pascal Dugénie, and Stefano A. Cerri. Agent-Grid Integration Language. *International Journal on Multi-Agent and Grid Systems*, 4(2):167–211, 2008. hal-lirmm.ccsd.cnrs.fr/lirmm-00139691.
19. Clément Jonquet, Pascal Dugénie, and Stefano A. Cerri. Service-Based Integration of Grid and Multi-Agent Systems Models. In R. Kowalczyk, M.N. Huhns, M. Klusch, Z. Maamar, and Q.B. Vo, editors, *International Workshop on Service-Oriented Computing: Agents, Semantics, and Engineering, SOCASE08*, volume 5006 of *Lecture Notes in Computer Science*, pages 56–68, Estoril, Portugal, May 2008. Springer-Verlag.
20. Julian R. Gallop Musbah Sagar Jeremy P.R.B. Walton Jason D. Wood Ken W. Brodlie, David A. Duce. Visualization in grid computing environments. In Greg Turk Holly Rushmeier and Jarke J. van Wijk, editors, *Proceedings of IEEE Visualization*, pages 155–162, 2007. ISBN:0-7803-8788-0.
21. Philippe Lemoisson and Stefano A. Cerri. Interactive construction of encore (encyclopdie de chimie organique electronique). *Applied Artificial Intelligence Journal Special issue on Learning Grid Services*, 19:933–966, 2005.
22. Pierre Lévy. *L'intelligence collective. Pour une anthropologie du cyberspace*. La Découverte, Paris, 1994.
23. Niklas Luhmann. *Social systems*. Writing science, 1995.
24. M.R Maturana and F.J Varela. *Autopoiesis and Cognition*. Reidel, Dordrecht, 1984.
25. S. A. Cerri T. Dimitrakos M. Gaeta P. Ritrovato, C. Allison and S. Salerno. Towards the learning grid: advances in human learning services. In J.N.Kok J.Liu R. Lopez de Mantaras R. Mizoguchi M. Musen N. Zhong R. D. J. Breuker, N. Guarino, editor, *Frontiers in Artificial Intelligence and Applications*, volume 127, pages X–240, Amsterdam, 2005. IOS Press.
26. Laura Pearlman, Von Welch, Ian Foster, Carl Kesselman, and Steven Tuecke. A Community Authorization Service for group collaboration. In *3rd International Workshop on Policies for Distributed Systems and Networks, POLICY'02*, pages 50–59, Monterey, CA, USA, June 2002. IEEE Computer Society.
27. V. Ciaschini A. Frohner A. Gianoli K. Lorentey F. Spataro R. Alfieri, R. Cecchini. Voms: an authorization system for virtual organizations. In *Grid Computing*, volume 2970 of *Lecture Notes in Computer Science*, pages 33–40, Heidelberg, 2004. Springer Berlin.
28. J. Shalf and E. W. Bethel. The grid and future visualization system architectures. *IEEE Computer Graphics and Applications*, 23(2):6–9, mars/avril 2004.
29. Guy Gouarderes Stefano A. Cerri and Roger Nkambou. Learning grid services. *Special issue: Applied Artificial Intelligence Journal*, 19(9-10):811–1073, 2005.
30. Ludwig von Bertalanffy. *General system theory : foundations, development, applications*. Braziller, New York, 1968.
31. Marc Weiser and John Seely Brown. Designing calm technology. *PowerGrid Journal*, July 1996.