# Applying Enterprise Models to Design Cooperative Scientific Environments

Andrea Bosin[1], Nicoletta Dessì[1], Maria Grazia Fugini[2], Diego Liberati[3], and Barbara Pes[1]

[1] Università degli Studi di Cagliari, Dipartimento di Matematica e Informatica, Via Ospedale 72, 09124 Cagliari
andrea.bosin@dsf.unica.it, {dessi, pes}@unica.it
[2] Politecnico di Milano, Dipartimento di Elettronica e Informazione, Piazza da Vinci 32, I-20133 Milano
fugini@elet.polimi.it
[3] IEIIT CNR c/o Politecnico di Milano, Piazza da Vinci 32, I-20133 Milano
liberati@elet.polimi.it

**Abstract.** Scientific experiments are supported by activities that create, use, communicate and distribute information whose organizational dynamics is similar to processes performed by distributed cooperative enterprise units. On this premise, the aim of this paper is to apply existing enterprise models and processes for designing cooperative scientific experiments. The presented approach assumes the Service Oriented Architecture as the enacting paradigm to formalize experiments as cooperative services on various computational nodes of a network. Specifically, a framework is proposed that defines the responsibility of e-nodes in offering services, and the set of rules under which each service can be accessed by e-nodes through service invocation. By discussing a representative case study, the paper details how specific classes of experiments can be mapped into a service-oriented model whose implementation is carried out in a prototypical scientific environment.

## 1 Introduction

A key success factor to promote research intensive products is the vision of a large scale scientific exploration carried out in a networked environment, with a high performance computing infrastructure, e.g., of grid type, that supports flexible collaborations and computation on a global scale. The availability of such a virtual cooperative environment should lower barriers among researchers taking advantage of individual innovation and allowing the development of collaborative scientific experiments.

Up to now, however, rarely are technologies developed specifically for the research community, and ICT developments are harnessed to support scientific applications varying in scale and purpose and encompassing a full range of engagement points, from single-purpose-built experiments to complex support environments.

The range of accessible technologies and services useful to scientific experiments can be classified broadly into three categories:

C. Bussler et al. (Eds.): BPM 2005 Workshops, LNCS 3812, pp. 281 – 292, 2006.

- Toolkits specifically aimed at supporting experiments.
- Software tools that are not specifically on-purpose for support to experiments, but that are still essential in enabling them (e.g., Matlab, data mining tools, data warehousing).
- More widely deployed infrastructures that may be useful in scientific experiments, such as Web services, or Grid computing.

Hence, a problem is the *integrated* use of heterogeneous applications and software tools that were not designed specifically to promote interaction and cooperation, but still are inherently suitable for cooperation support. This scenario is similar to that of enterprise environments, whose progress requires large-scale collaboration and efficient access to very large data collections and computing resources [1]. Although sustainable interoperability models are emerging for market players (such as service providers, stakeholders, policy makers, and market regulators), they are currently deployed mostly in areas where high computing power and storage capabilities, usually needed by scientific environments, are not mission-critical.

Recently, emerging technologies such as Web Services and the Grid [2] [3] [4] have enabled new types of scientific applications consisting in a set of services to be invoked by researchers. Assuming the Service Oriented Architecture (SOA) as enacting paradigm [5], the purpose of this paper is to explore the application of existing enterprise models and processes for supporting distributed scientific services. To this aim, we address how experiments can be formalized as workflows that express a sequence of cooperative tasks, each performed by a set of e-services. The "enterprise" environment supporting distributed scientific "processes" is a network of cooperative e-nodes (e.g., the research laboratories, the hospitals, the analysis centers) having a local configuration and a set of shared resources. Since an experiment involves multiple e-nodes interacting with one another in order to offer or to ask for services, the paper proposes a framework that defines the responsibilities of e-nodes in offering services, and the set of rules under which each service can be accessed by e-nodes through service invocation.

To illustrate how a methodology for executing scientific experiments can be decomposed into simpler tasks, the paper considers four main classes of methodological approaches directly pertaining the experimental environment, whose application is mainly thought for data (and possibly also computationally) intensive processing. To identify how an experiment can be mapped onto a service oriented model, we discuss a representative case study related to one of the considered classes. In more detail, we present a model of the experiment in a 4-level UML Use-Case diagram. Finally, a prototype environment based upon emerging Web service technology is described and applied.

The paper is organized as follows. Section 2 reviews some related works on cooperative systems and e-services. The cooperative framework is described in Section 3. Section 4 outlines four main classes of experiments and Section 5 shows how these experiments can be modelled using UML diagrams. An implementation prototype is presented in Section 6. Finally, conclusions as well as future works are outlined in Section 7.

## 2 Related Works

In general, the term cooperative systems is used to denote distributed information systems that are employed by users of different organizations under a common goal [6][7]. An extension of the cooperative paradigm, referred to as *e-applications*, is becoming more and more frequent: e-applications allow the dynamic composition of services, referred to as *e-services* in the following, provided by different organizations on the net. In addition to geographical distribution and inter-organization cooperation, in e-applications (i) cooperating organizations may not know each other in advance and (ii) e-services can be composed both at design and run-time. Moreover, the availability of complex platforms for e-services [8] allows open cooperation among different organizations.

A classical approach to e-applications is based on UDDI registry, allowing publication and discovery of services on the Web. In UDDI, offered e-services are described, based on free text, and consumers of the e-services interact with the offering organization on the basis of interfaces. Other proposals for architectures for e-services and workflow-based environments have been recently presented in the literature [8][9]. The starting point is the concept of *cooperative process* [10][11], defined as a complex process involving different organizations.

Activities that involve multiple different organizations using e-service technology to exchange information need to be coordinated. WSDL is one of the currently most popular linguistic means providing a mechanism by which the format and structure of the exchanged messages is defined. Methods are not yet completely defined in the literature to specify the sequence and conditions, or *choreography*, of message exchange. A proposal is under study in the W3C [12]. To solve this problem, a shared common or global definition of the sequence and conditions where messages are exchanged is produced that describes the behavior of all the involved participants.

In general, high-performance computing and communication technologies are enabling computational scientists, or e-scientists, to study and better understand complex systems, allowing for new forms of collaboration with the ability to process, disseminate, and share information [13]. Global-scale experimental networking initiatives have been developed in the last years: the aim is to advance cyberinfrastructure for e-scientists through the collaborative development of networking tools and advanced Grid services [14] [15].

## 3 The Cooperative Framework

The concept of "what an experiment is" is rapidly changing in an ICT oriented environment, moving from the idea of a local laboratory activity towards a computer and network supported application including the integration of:

- a variety of information and data sources;
- the interaction with physical devices;
- the use of heterogeneous software systems.

In our approach, scientific experiments are modeled analogously to cooperative enterprise processes, as *e-processes* that operate on, and manipulate, data sources and

physical devices. Each e-process is attributed to and associated with a set of specific experimental tasks and workflows are employed to control these tasks.

This modeling approach is assumed as an enactment paradigm to conceive an experiment, at least for certain application domains; an experiment is regarded as an application whose tasks can be decomposed and made executable as (granular) services individually. The decomposition is based on appropriate modeling of the experiment as a set of components that need to be mapped to distinct services. The decomposition offers good opportunities for achieving an open, scalable, and cooperative environment for scientific experiments.

Let us now introduce the framework modeling the experiments on a network of cooperative e-nodes having a local configuration and a set of shared resources. Services correspond to different functionalities across several research domains, and encapsulate problem solving and simulation capabilities. All services have an e-node that is responsible for offering the service and which sets the rules under which the service can be accessed by other e-nodes through service invocation. An experiment involves multiple e-nodes interacting with one another in order to offer or to ask for services.

A user is a researcher who has the following possibilities:

1) selection of the experiment of interest and of the information sources he/she wants the experiment be carried on;
2) acquisition and collection of local data;
3) surveillance/monitoring of local experiments, which are part of a cooperative experiment;
4) definition of new experiments; this implies that the global workflow of the experiment must be designed;
5) inspection of remote data sources and experiment results, e.g., by mining in a data warehouse;
6) cooperation with users of other e-nodes, for example to co-design experiments and jointly evaluate results.

Fig.1 shows a conceptual framework for e-nodes cooperation whose basic components are Pool of Services (PS) and Smart Scientific Spaces ($S^3$). *A **Pool of Services (PS)*** is an e-node described by:

1) A set of local information sources;
2) A set of local services.

The *local information sources* collect the experimental data related to the activities assigned to the e-node. Their granularity can vary from the file, to the database, or data warehouse level. Parts of local information sources can be declared as public by the e-node and hence become part of the network (i.e., remotely accessible).

The *local services* map granular experimental tasks performed by the e-node or other local control activities. The services organize their activity on the basis of both local and network information sources, and are related to a particular experimental context by a workflow describing the tasks to be executed and the context knowledge applied to solve a problem, to enact a decision or to achieve a goal.

The *design of an experiment* is accomplished by a *Chief Scientist* user who is in charge of breaking down the experiment into a sequence of tasks to be executed in a distributed way. The granularity of these tasks is decided by the Scientist, and their

assignment to the e-nodes is negotiated in terms of resources, time, costs and other project management parameters by an *Experiment Manager* user. This user ensures that each task is realizable by some e-node according to fixed time and cost parameters, in order to avoid unrealizable and ambiguous tasks. The specified tasks express also the life cycle of the experiment and defines the set of PSs participating in the experiment.

In the case of complex experiments requiring the collaboration of various services spread over different e-nodes, the selected PSs interact in a particular environmental context called ***Smart Scientific Space*** **($S^3$)** and established by a selected e-node that act as *Experiment Master Pool* (EMP). The EMP is responsible for setting up, controlling, and splitting the experiment. A representation schema enables the EMP to identify the PS allowed to interact with one another in the context of the ($S^3$), and a workflow describes the experiment plan, i.e., the sequence of activities to be carried on to execute the cooperative experiment.

Each interacting PS locally has the knowledge of the portion of workflow activities, called *component services*, that are assigned to the node. Making a component service available in the appropriate smart space requires registration facilities to be available at the EMP. Besides local knowledge, the local workflow includes: points of access to remote information sources, and points of cooperation with other e-nodes' workflows, for example requests of parallel processing services available on a specialized e-node, as well as possible destinations of processed information.

The EMP is also in charge of designing the cooperative experiment and of configuring a monitoring activity (project management) to ensure that the experiment plan is adhered to. A cooperative experiment is represented by a single $S^3$ or by a series of distinct smart spaces, each responsible for different facets of a single cooperative experiment. This mechanism enables different PSs to retain the ownership of their own experiments, but to enable the cooperation of PSs under appropriate conditions.

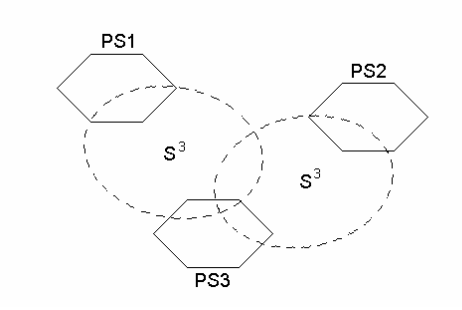


**Fig. 1.** Conceptual framework for e-nodes cooperation: Pool of Services (PS) and Smart Scientific Spaces ($S^3$)

The global experimental environment is viewed as a number of smart scientific spaces where the physical distribution of resources and access problems are masked via the service interface and the workflow ensures that appropriate PSs cooperate. As new information sources and processing techniques become available, they are simply represented as new services thus ensuring the environment scalability.

## 4 Classes of Experiments

In our approach, an experiment is defined as the application of a subset of the available methodological approaches to the selected data set on an execution platform composed of: 1) a workflow; 2) the distribution thereof; 3) the involved nodes and their relative roles in the experiment; 4) the set of involved resources, such as data areas, data repositories, and e-services.

Four main classes of methodological approaches to the experiments can be identified:

1. *Process Simulation and Visualization* on the already available information sources.
2. *Supervised or Unsupervised Classification* of observed events without inferring any correlation nor causality, such as in clustering, and neural networks.
3. *Machine Learning: Rule Generation and Bayesian Networks* able to select and to link salient involved variables in order to understand relationships and to extract knowledge on the reliability and possibly causal relationships among related co-factors via tools like logical networks and Cart-models.
4. *Identification of the Process Dynamics*.

Such classes, listed in increasing order of logical complexity, might have an impact on the design of the experiment and of its execution modality in terms of execution resources. For example the first class does not seem to require more than mediating possibly different repositories, while the other classes may require recursive approaches and hence intense computation either on a single specialized node or in a grid structure.

In more detail, in the following we are going to identify how an experiment can be mapped onto a service-oriented model. As a case study, we consider an experiment related to the identification of process dynamics: the piecewise affine identification problem [16].

Piecewise affine identification associates temporal data points in the multivariable space in such a way to determine both a sequence of linear sub-models and their respective regions of operation. It does not even impose any continuity at each change in the derivative. It exploits a clustering algorithm. The three following steps are executed:

*Step 1 – Local linear identification*: small sets of data points close to each other are like to belong to the same sub-model. Thus, for each data point, a local set is tentatively built, collecting the selected point together with a given number of its neighbors. For each local data set, a linear model is identified through least squares procedure.

*Step 2 – Clustering*. The algorithm clusters the parameter vectors of the identified models and thus also the corresponding data points.

*Step 3 – Piecewise affine identification.* Both the linear sub-models and their regions are estimated from the data in each subset. The coefficients are estimated via weighted least squares, taking into account the confidence measures. The shape of the polyhedral region characterizing the domain of each model may be obtained via Linear Support Vector Machines [17].

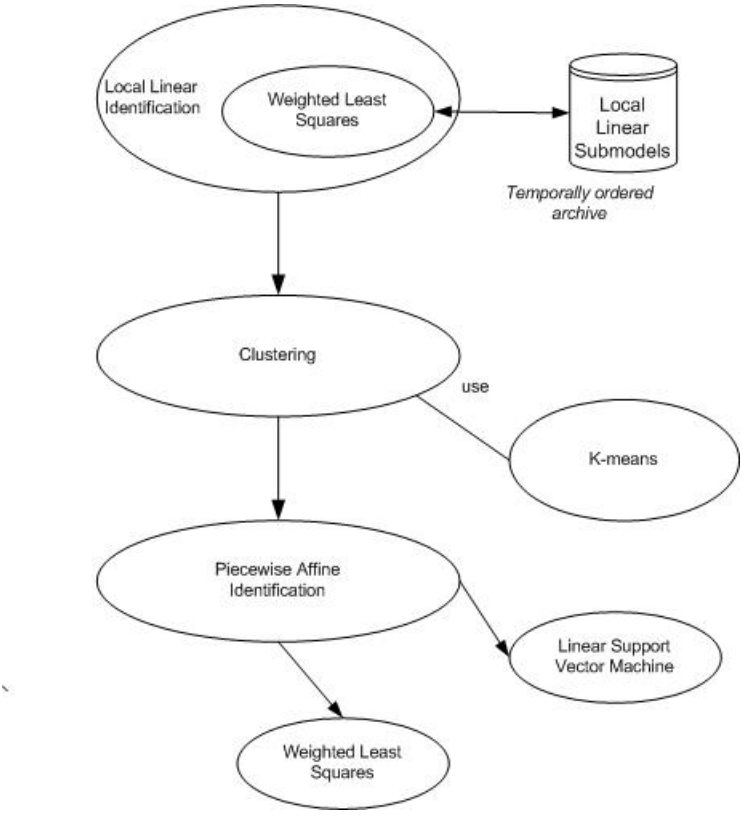


**Fig. 2.** Steps of the identification of the process dynamics experiment

The steps are depicted in Fig.2, where the circles represent activities and the arrows represent dependencies. The K-means algorithm is employed (*use* line) and can be (re)used several times. As an independent step, the Clustering activity can be assigned to a dedicated node. Also the algorithm deputed to define the regions in which every linear model is valid may be one of the many general purpose available for such task (here a simple Support Vector Machine is chosen). As such, it is also indicated as a separate re-usable module, useful in many other problems. Moreover, the Weighted Least Square module is another possible re-usable e-service, being used twice within this kind of experiment.

The illustrated experiment shows that some portions, both of processes and of data or knowledge, can be shared in a collaborative environment. One of the reasons for executing an experiment in a distributed way might be that one organization needs to process data under a specific costly product available on a node; rather than acquiring the product (e.g. SAS, Oracle, or Matlab), the organization invokes a remote service on the remote node endowed with the license. Another reason is that some cooperating organizations might want to inspect data dispersed on their databases, with no changes to their local computational environment.

## 5 Modeling Scientific Experiments

In the current laboratory practice, a researcher begins with the assertion of a high level goal needed to test a scientific hypothesis or to obtain some additional knowledge on a previous experiment.

According to the presented approach, this goal has to be decomposed into a set of tasks (the experiment life cycle) each accomplished by an appropriate class of services. From a methodological point of view, we observe that heterogeneous services can provide similar capabilities, but the Chief Scientist is in charge of choosing the most suitable methods to accomplish each task, that is, he is in charge of designing the workflow of the scientific experiment. In particular, if the researcher wants to rerun an experiment, the workflow must take into account the changes in the choice of methods as well as in the service availability.

In the presented approach, the Chief Scientist interacts and chooses services, workflows, and data within an experimental environment whose cooperative framework has been defined to extend the integration of scientific experiments to a level of scenario-based interaction. This scenario is profitable for many reasons, like exchanging scientific data and processing tools which results in a reduced number of software acquisitions, or load balancing work between specialized researchers.

The modeling process is organized in three steps.

1) The experiment is decomposed into a set of basic tasks.
2) A model is provided that structures basic tasks according to the template of the experiment.
3) The experiment flow is stated by the definition of a public workflow for all pools involved in the experiment.

In some more detail, we again focus on *Identification of the Process Dynamics* experiments since they are representative for the considered classes of experiment. The experiment decomposition has been described in the 4$^{th}$ class of experiments, and Fig. 2 shows the flow of basic tasks. This flow may be considered identical for all the first-glance experiments on Identification of the Process Dynamics, except for the choice of the clustering methodology, that can be different from the k-means algorithm.

We then consider the experiment to be performed in the following scenario. Two researchers at two different PSs (namely the pools PA and PB) carry on a cooperative experiment belonging to the class *Identification of the Process Dynamics*. The pool PB acts as EMP and the pool PA as a cooperative pool. Since we suppose that a Support Vector Machine knowledge is not available at either pools, they need the cooperation of a third pool PC, where the Support Vector Machine service is available. Also, it may be substituted by a more complex but also more refined technique, should the specifications of the experiment require such effort, which would probably need in turn a still different domain knowledge.

Fig. 3 shows the model of the experiment in a 4-level UML Use-Case diagram. Pools are modeled by participating actors invoking services offered at different levels. The core idea is that the scientist designs an abstract workflow out of the available services by hiding the low-levels details.

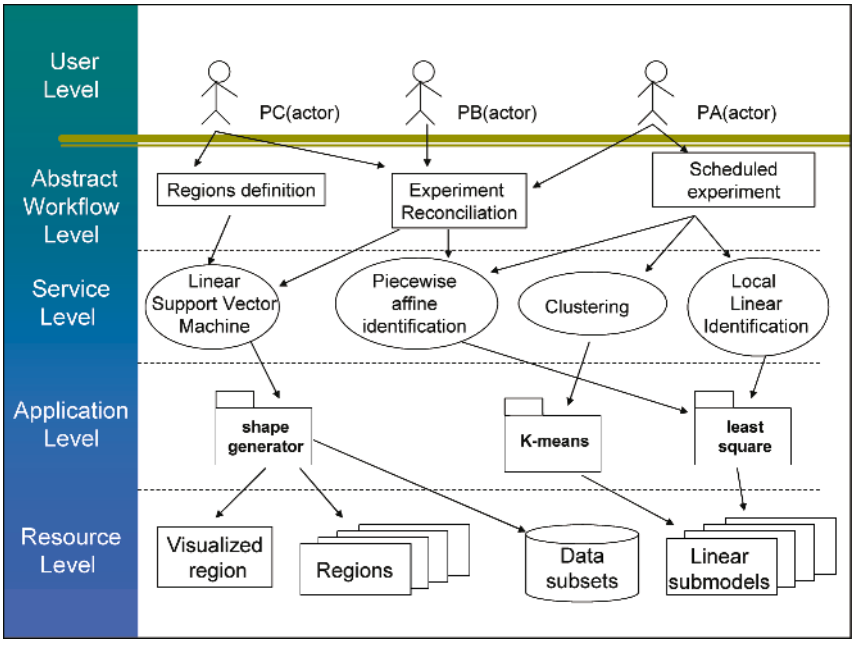


**Fig. 3.** The model of the experiment in a 4-level UML Use-Case diagram

It is important to outline that our approach takes into account the structure of the whole experiment rather than only the workflow layers as conventional workflow systems do.

## 6 Addressing Implementation Challenges

The proposed approach views a scientific experiment as a process and a flow of relationships rather then a compartmentalised event. It assumes that users are similar in that ready access to various resources will empower collaborative experimentation for novice users as well as for experts. What remains open, however, are many questions about the evaluation of such a approach. Since the realisation of a distributed general purpose scientific environment is not immediate, the evaluation effort described here involves a prototypical environment based upon emerging web service technology and applied to the above mentioned four classes of experiments.

The prototype is intended to help scientists to extract increasingly complex knowledge from data, promoting information reuse in well defined experimental patterns. In this section, we detail the general architecture of the prototype that implements a set of basic data mining tasks as widely accessible Web services. What we intend to demonstrate is how to realize in practice data mining e-services that can be used in a distributed cooperative experiment, according to the layered model shown in Fig. 3. In particular, the abstract workflow layer, dealing with planning and composition of all the tasks that make up an experiment, has not been implemented yet. We plan to

do it making use of a workflow engine based on WfMC [18] and OMG [19] specifications, which uses the WfMC XPDL definition format, such as Enhydra Shark [20].

The implementation of lower levels is based on the J2EE [21] and Oracle [22] platforms; however, the use of standard technologies (HTTP, XML, SOAP, WSDL) and languages (Java, SQL) makes it flexible and easily expandable.

The prototype maps the layered model into the multi-tier architecture shown in Fig. 4. While the tier separation can be purely logical, our prototype allows the physical separation of tiers, where each one is located on a separated and networked hardware resource.

The user tier represents the consumers of data mining services. Clients located across the Web (i.e. on the e-nodes of the experiment which requires the data mining service) invoke the data mining services provided by the service layer. Clients are implemented by standalone Java applications that make use of existing libraries (J2EE application client container) in charge of the low-level data preparation and communication (HTTP, SOAP, WSDL).

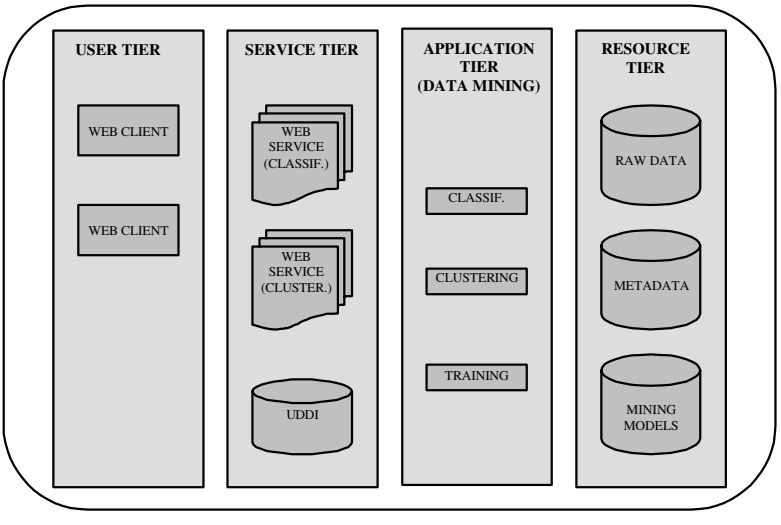


**Fig. 4.** The multi-tier architecture

The service tier exposes the data mining services as web services to its clients. Web services are implemented as Java web components along with their WSDL description, and are deployed in a web container managed by a J2EE Application Server. The application server and the web services it exposes can run anywhere on the web. The clients through the use of an UDDI registry can query web service endpoint locations. At this level, the implementation-independent web service invocation performed by clients, is mapped to the specific data mining implementations (i.e. the Java classes, specific to Oracle or to other data mining platforms, are activated).

The core of the data mining activities is implemented by the application tier, that consists in proprietary Java libraries and PL/SQL procedures (Oracle data mining

server). This level can be distributed across the Web (different mining activities can take place on different e-nodes), and can be physically separated from the previous level, as it is in our implementation.

The resource tier is composed by the data repositories (raw data, metadata, mining models, etc.) accessed by the single data mining activities. In our implementation all the data repositories are managed by a single DBMS [23] which also acts as a data mining server (at the data mining level), although in general this is not required since data repositories can be distributed. Actually, the prototype supports access to flat files and structured data including relational tables stored by a DBMS, that can be combined to build highly personalised data-sets. Remote raw data can be pre-processed for data mining tasks and added to the database so that they can serve for future experiments. The actions performed by the prototype include the access to user procedure as well as to a DBMS data-mining component [23].

Although we have implemented a prototypical version, aimed at carrying out data mining experiments, our approach is extendible to different cooperative experiments.

## 7 Conclusions and Future Work

We have illustrated how enterprise models and processes can be applied for modeling scientific cooperative services. The proposed cooperative framework for distributed experiments is quite general and flexible, being adaptable to different contexts.

Given the challenge of evaluating the effects of applying emerging web service technology to the scientific community, the evaluation performed up to now takes a flexible and multi-faceted approach: it aims at assessing task-user-system functionality and can be extended incrementally according to the continuous evolution of scientific cooperative environment.

In future works, we are going to extend the classes of experiments of interest, to implement the proposed structure in specific application fields where the need looks of primary relevance, and of course to detail the sequences of interaction among actors in the specific use cases.

## References

1. Pollock J.T., Hodgson R., Adaptive Information: Improving Business Through Semantic Interoperability, Grid Computing, and Enterprise Integration, Wiley Series in Systems Engineering and Management, Wiley-Interscience, 2004.
2. De Roure D., Baker M.A., Jennings N. R., Shadbolt N., The Evolution of the Grid, in Berman et al. (eds), 2003.
3. Berman F., Fox G., Hey T. (eds), Grid Computing: Making the Global Infrastructure a Reality, John Wiley and Sons, Inc., New York, 2003.
4. Foster I., Kesselman C., Nick J.M., Tuecke S., The Physiology of the Grid: An Open Grid Service Architecture for Distributed System Integration, The Globus Project, 2003.
5. Pilioura T., Tsalgatidou A., e-Services: Current Technologies and Open Issues, Proc. of VLDB-TES, 2001.
6. Brodie M.L., The Cooperative Computing Initiative. A Contribution to the Middleware and Software Technologies, GTE Laboratories Technical Publication, 1998.

7. Proceedings of the 6th International Conference on Cooperative Information Systems (CoopIS), 25 - 29 October 2004, Larnaca, Cyprus, Springer Verlag, 2004.
8. Mecella M., Parisi Presicce F., Pernici B., Modeling e-Services Orchestration through Petri Nets, Proc. VLDB-TES Workshop, 2002.
9. Baresi L., Bianchini D., De Antonellis V., Fugini M.G., Pernici B., Plebani P., Context-aware Composition of E-Services, Proc. VLDB-TES Workshop, Berlin, September 2003.
10. Schuster H., Georgakopoulos D., Cichocki A., Baker D., Modeling and Composing Service-based and Reference Process-based Multi-enterprise Processes, Proc. 12th International Conference on Advanced Information Systems Engineering (CAISE 2000), Stockholm, Sweden, 2000.
11. Hsieh Y., Shi M., Yang G., CovaModeler: A multi-user tool for modelling cooperative processes, International Journal of Computer Applications in Technology 2003 - Vol. 16, No.2/3 pp. 67-72.
12. W3C Working Draft 24 March 2004, http://www.w3.org/TR/2004/WD-ws-chor-model-20040324/ .
13. Brown, M. (2003). "Blueprint for the Future of High-performance Networking", Communications of the ACM, vol. 46, no. 11.
14. De Fanti, T. et. al. (2003). "Translight: A Global-scale LambdaGrid for e-science", Communications of the ACM, vol. 46, no. 11.
15. Newman, H. et al. (2003). "Data-intensive for e-Science, Communications of the ACM, vol. 46, no. 11.
16. Ferrari Trecate G., Muselli M., Liberati D., Morari M., A clustering technique for the identification of piecewise affine systems, Automatica 39: 205-217, 2003.
17. Vapnik V., Statistical Learning Theory, New York: Wiley, 1998.
18. http://www.wfmc.org
19. http://www.omg.org
20. http://shark.objectweb.org
21. Armstrong E., Ball J., Bodoff S., Bode Carson D., et al., The J2EE 1.4 Tutorial, December 16, 2004.
22. http://www.oracle.com
23. http://www.oracle.com/database/Enterprise_Edition.html