IL NUOVO CIMENTO Vol. 28 C, N. 2 Marzo-Aprile 2005
DOI 10.1393/ncc/i2005-10185-2

MOSÈ: A grid-enabled software platform to solve geoprocessing problems(*)

G. Folino, A. Forestiero, G. Papuzzo and G. Spezzano

Institute for High Performance Computing and Networking (ICAR)-CNR
87036 Rende (CS), Italy

(ricevuto l'8 Febbraio 2005; approvato il 23 Maggio 2005; pubblicato online il 23 Settembre 2005)

Summary. — Grid computing has emerged as an important new field in the distributed computing arena. It focus on intensive resource sharing, innovative applications, and in some cases, high-performance orientation. This paper describes how grids technologies can be used to develop an infrastructure for developing geoprocessing applications. We present the MOSÈ system, a grid-enabled problem solving environment (PSE) able to support the activities that concern the modelling and simulation of spatio-temporal phenomena for analyzing and managing the identification and the mitigation of natural disasters like floods, wildfires, landslides, etc. MOSÈ takes advantages of the standardized resource access and workflow support for loosely coupled software components provided by the web/grid services technologies.

PACS 07.05.Tp – Computer modeling and simulation.
PACS 01.30.Cc – Conference proceedings.

1. – Introduction

The MOSÈ (Spatio-Temporal *MO*delling of Environmental Evolutionary Processes by means of Geo*SE*rvices) system is a grid-based problem solving environment (PSE) for the developing of geoprocessing applications. MOSÈ is a PSE able to support the activities that concern the modelling and simulation of spatio-temporal phenomena for analyzing and managing the identification and the mitigation of natural disasters like floods, wildfires, landslides, etc. The activities managed by MOSÈ are characterized by the necessity to handle large amount of spatio-temporal data and to support the interoperability among simulation models, distributed GIS, visualization systems, parameter

(*) Paper presented at CAPI 2004, 8° Workshop sul calcolo ad alte prestazioni in Italia, Milan, November 24-25, 2004.

estimation services, discovery of spatio-temporal patterns in pre-existing data, etc. In this domain, the data conversion and the access, search, discovery and organization processes are complex problems because data are geo-referenced, stored in distributed GIS and can be used along three dimensions: temporal, spatial and referred to the physical properties. MOSÈ provides web-based access to the spatial data by a browser and allows to observe and manipulate data in a 2D/3D space by selecting regions in thematic maps. Users can examine features and patterns in a map in order to identify the region from which data must be extracted. Different physical properties can be extracted by thematic maps by data mining algorithms (clustering and classification) and used to select the most appropriated simulation model for the region analyzed. Similarly, from a temporal point of view, users can extract data that concern different parts of a temporal graph defining a period of time where the temporal data must be searched. Selected data can be used in input to a cellular automata simulation model of the phenomenon that interacts with components for the parameter estimation for automated calibration of the model, 3D data visualization, and spatial data mining tools for analyzing the results.

MOSÈ uses a web-service–based computing portal architecture to coordinate the access to the resources. Workflow technology is used to compose the services. The main components of MOSÈ are simulation services, geographic information (GI) services, geographic data and catalogues providing ontologies and metadata on the data and services. Each component has a wrapper and a XML interface for a simple composition.

This first version of the MOSÈ system regards the analysis of landslide hazard areas in the Region Campania near the Sarno area. The main actor in this scenario is a manager who wants to get an overview of the Sarno area with the indication of the regions which are currently slid down and those which are susceptible to slide down (landslide hazard areas) within a fixed time. For each scenario, the manager generates a workflow that orchestrates the web services necessary to obtain the outcome, and submitted it to the MOSÈ workflow enactment engine, which takes care of its execution. Some of the components that constitute the MOSÈ system use results of previous research developed in the past years and guarantee high performance and accuracy of the results [1].

The paper presents the software architecture of the MOSÈ system (sect. **2**), an example of grid service that enables geological simulations to be executed on the grid (sect. **3**), the geographical metadata and ontology model adopted (sect. **4**). Moreover, an example of workflow design shows how a scientist, using MOSÈ, can execute an analysis of landslide hazard areas of Sarno (sect. **5**). Finally, conclusions are presented (sect. **6**).

2. – Software architecture of MOSÈ

The MOSÈ middleware is built on existing web/grid services technologies and standards [2]. This section provides brief information on the MOSÈ architecture. A web-based interface, shown in fig. 1, is used to access the services offered by MOSÈ. The web-based portal supplies access to the spatial data by the client browser and allows observing, selecting, and manipulating data in a 2D/3D space selecting regions in thematic maps. Users can examine features and patterns in a map in order to identify the region from which data must be extracted and/or analysed. The MOSÈ architecture uses a client/server topology employing a service-oriented architecture. The architecture, shown in fig. 2, includes some components exported as web/grid services, each with an associated repository preserving historical (or previously created) information, a workflow executor and web-based access to a Geographical Information System (GIS).

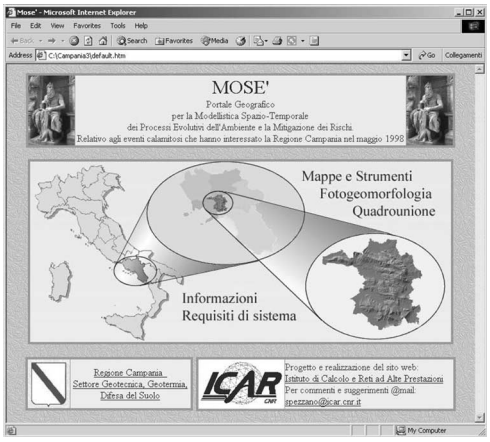

Fig. 1. – MOSÈ web-based GUI.

The main components exported as web/grid services are:

– Data extractor component, to extract raster maps from the GIS by Geomedia Web Map tool.

– Visualization component, based on AVS-Express, to implement 2D/3D visualizations and virtual reality representations of one or more layers of the data extracted from the GIS.

– CamelotGrid, a cellular automata (CA) simulation tool running on the computational grid (this component is described in detail in the next paragraph). In a Cellular Automaton time is discrete like in a map, space is discrete (the cells of a cellular automaton) and also state space is discrete (each cell can have only a finite number of states). So it is the simplest dynamical system that describes systems evolving in space. A spatial process is normally described by a set of PDE (Partial

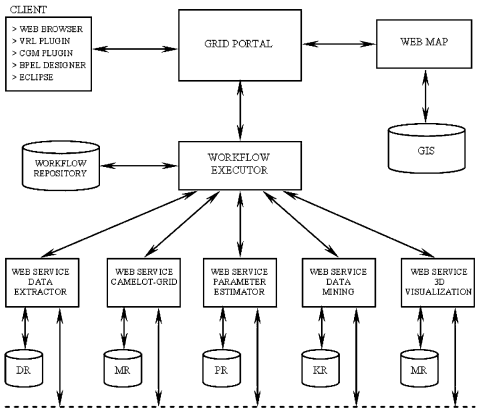

Fig. 2. – The MOSÈ software architecture.

Differential Equations) with time, space and state space continuous. Cellular automata may thus be considered as discrete idealizations of the partial differential equations often used to describe natural systems.

– Estimation of model parameters component, based on a parallel genetic algorithm running on a parallel machine available on the grid. The CA models simulated with CamelotGrid are calibrated with the parameters that are estimated by this component.

– Data mining component, performing operations of spatial clustering, classification, etc.

The core of the system is the Workflow Executor (WE) that receives a workflow built by BPEL Designer and executes it on the grid. A repository is associated to each component to reuse results or models previously obtained. CamelotGrid maintains a repository of the models of simulations, the parameter estimation service retains the parameters estimated for different regions for future reuse, the data extractor keeps data in the data repository, the 3D visualization component maintains a repository of 3D models of its simulations and finally the data mining component uses a knowledge repository to save acquired knowledge. Note that models can be obtained from previously executed simulations or can be inserted ex-novo using apposite tools of the PSE. MOSÈ can be used to execute simulations of different complex natural phenomena. Users must specify the new CA model of the phenomenon and transfer it to CamelotGrid. New metadata and ontologies describing the data must be introduced and new GIS containing the geographical data for the application can be linked by the Web Map tool.

3. – An example of grid service: CamelotGrid

In this section, CamelotGrid is presented as a significant example of grid service implemented in MOSÈ. CamelotGrid exploits the resources of the grid in order to reduce its execution time and improve simulation performances. CamelotGrid is a simulation environment that provides a complete integrated computing environment for CA programming, allows to specify the global criteria defining the autonomic requirements of the application and to support the execution of cellular applications over the grid. It extends the original Camelot [3] architecture for incorporating the features of self-configuring, self-optimizing, self-healing, etc., of an autonomic system [4], in order to realize a grid-enabled middleware architecture where the runtime autonomic management of the application is done without any user intervention. In CamelotGrid a spatio-temporal problem can be modelled by a 2D or 3D array of cells where each cell represents a portion of a landscape. CamelotGrid extends the basic cellular automata model and allows to simulate CA having the following features:

– temporal and spatial heterogeneity both in the transition function and in the neighborhood;

– asynchrony in the transition function so that every cell can, at each step, nondeterministically choose between changing its state according to σ or keeping it;

– complex time-dependent neighborhoods (*i.e.* block rules), and

– probabilistic and hierarchical transition functions.

```
.....
  steering {
    .....
  }

  autonomic {
    if ( step % 10 == 0 ){
      if ((Cardinality (VM)== NumberOfNodes && Efficiency < 0.7 )
          redistribute();
      if ((OnFailure (VM)) || responseTime > 2))
          reconfigure();
    }
  }
```

Fig. 3. – An example of use of autonomic section in CARPET.

By means of CARPET [3] (a language used in Camelot to define cellular algorithms), a user can describe, using the declaration, body and steering section the transition function that represents the cellular program of the complex phenomenon that he/she intends to model. In the new version, CARPET has been extended with a new section, called autonomic. In the autonomic part, a user can define a set of rules to specify high level policies that capture different aspects of autonomic behaviour. For example, suppose that the current resources (*i.e.* processors available) represent the virtual machine (VM) on which the cellular program will be executed. We can specify a threshold for the desired efficiency under which a *dynamic reallocation* of the automata must be performed in order to maintain the efficiency. In alternative the application must be *reconfigured* if the function *OnFailure* return a value true or a fixed response time is passed. In fig. 3 it is shown how this can be specified in the autonomic section.

The efficiency (computed using the performance model described in [5]) and the response time are evaluated at run-time and automatically updated each ten steps.

In Camelot, the run-time support was implemented as a SPMD (Single Program, Multiple Data) program. The latest implementation is based on the C language plus the standard MPI library and can be executed on different parallel machines and clusters of workstations. The concurrent program that implements the architecture of the system is composed by a set of macrocell processes, an execution engine and a GUI process. Each macrocell process operating on a strip of cells of the CA runs on a single processing element of the parallel machine and updates the state of cells belonging to its partition. The synchronization of the automaton and the execution of the commands, provided by a user through the GUI interface or described in the steering section, are carried out by the execution engine. The execution engine partitions the 2D or 3D cellular space according to the indications of the user and assigns the portion of the cells that must be processed to each macrocell process. We extended the CAMELot architecture to support the development of autonomic cellular applications on the grid. CamelotGrid is the new middleware architecture implemented on existing grid middleware and runtime services. An overview of the CamelotGrid middleware architecture is shown in fig. 4.

The grid middleware layer takes advantage of the services offered by the Globus Toolkit [6], by the MPICH-G2 library, a grid-enabled implementation of the MPI v1.1 standard, and by the Network Weather Service (NWS), a monitoring system to forecast short-term network performance. In particular, NWS provides accurate forecasts of dynamically changing performance characteristics from a distributed set of metacomputing resources. MPICH-G2 uses services (*e.g.*, job startup, security) provided by Globus to

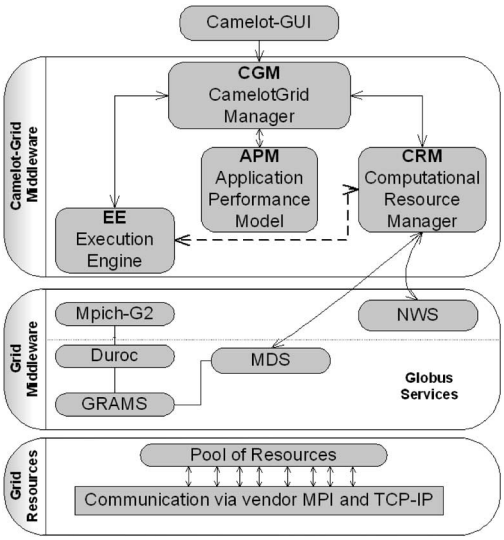

Fig. 4. – The CAMELotGrid middleware architecture.

coordinate and manage work on multiple computer systems, potentially of different architectures. The main components of the CamelotGrid infrastructure architecture include the CamelotGrid Manager (CGM), the Computational Resource Manager (CRM), the Application Performance Model (APM), and the Execution Engine (EE).

In the following part of this section, it will be described how we use the CamelotGrid infrastructure services to exploit the heterogeneous Grid resources and to achieve autonomic runtime management. The main administrative component in the CamelotGrid architecture is CGM, which is the autonomic application manager that sets up and configures the application execution environment, manages and controls all the autonomic requirements (*e.g.*, self-optimizing, self-configuring, etc.) through the rules defined in the autonomic section of CARPET. When the WE invokes CamelotGrid a CGM is started for each application. CGM starts APM that allows predicting application execution times on a given set of resources with a given topology. CGM uses APM to choice the most appropriate tool of resources in order to schedule the macrocell processes [5]. Afterwards, CGM starts the CRM module that provides CGM with resource information, including the number of current available computation resources and their usages. CRM analyzes the information coming from the grid and notify any changes on the network resources to CGM. For example, CRM can determine if the network is congested or if new nodes are available in the grid environment. CRM uses the two most widely used grid resource information systems, the Metacomputing Directory Service (MDS) and NWS. MDS collects and publishes system configuration, capability, and status information such as operating system, processor type and speed, number of available CPUs, and the software installed. The information typically retrievable from a NWS server includes the fraction of CPU available to a newly stated process, the amount of memory currently unused, and the bandwidth at which data can be sent/received to/from a remote host. Using the information coming from CRM about the number and the main characteristics of nodes available on the grid and the network performance, CGM, assisted by APM, identifies a subset of the available resources to constitute the VM on which the

application will be executed. Then the same CGM calculates a mapping of data and/or task for these resources and decides the best schedule for the macrocell processes. Note that CGM uses an adaptive scheduling algorithm based on the scheduler of GRADS (GRid Application Development environmentS) described in [7]. Finally CGM divides the data space into subspaces, defines the initial allocation of the macrocell processes to physically available resources and starts the execution engine. The EE generates the processes required to simulate the model on each subspace in parallel. EE can observe and control the simulation by the commands defined in the steering section. On request of CGM, EE can save or load a configuration of the automaton, abort and restart the computation with a new mapping and schedule. Furthermore, EE can communicate to CGM and CRM information about the termination and the status of the computation of the automaton. GGM receives information on the dynamic changes of the VM from CRM and cooperates with APM to determinate whether the application is delivering an acceptable level of performance (*e.g.*, a level of efficiency). The evaluation of acceptable levels of performance is the shared responsibility of the APM and the CGM. To this aim, CGM uses the autonomic section of CARPET. During the execution, if CGM determines that the application is not making reasonable progress with respect to the defined policies (or alternately, if the system becomes aware of more suitable execution resources), a rescheduling action can be invoked. Examples of rescheduling actions are replacing particular resources, redistributing the application workload/task on the current resources, and adding or removing resources, or doing nothing (continuing execution with the current VM).

4. – Metadata and ontology

Ontologies and metadata are the basic elements through which we can build a semantic infrastructure that underpin resource sharing and interoperability. Through ontologies the concepts of a particular domain can be shared between different applications. Ontologies can be used to capture domain expertises deploying, enabling understanding and sharing of knowledge. Inference engines can draw conclusions based on the rules or axioms to create new knowledge and eventually to solve problems. Independently of the modeled domain, ontologies can be used to: share common knowledge; allow reuse of knowledge about a domain, analyze a knowledge domain, categorize items (*e.g.*, sale products or web sites), etc.

The effective use of a grid-based PSE requires the definition of an approach to manage the heterogeneity of the involved resources that can include computers, data, network facilities, sensors, and software tools provided by different organizations [1]. Heterogeneity arises mainly from the large variety of resources within each category. The management of such resources requires the use of metadata that, through an accurate categorization of resources, provides useful information about the features of resources and their effective use. In MOSÈ metadata information associated to a resource is divided into the following sections:

- Ontological metadata, used to classify the resource.
- Semantic metadata, to characterize the resource within the specific geoprocessing domain or other related.
- Resource metadata individuates the characteristics specific to a type of resource (*e.g.*, characteristics of software, data sources, etc.) and specifies how to access and use a resource (parameters, URI (Uniform Resource Identifier), etc.).

Specific metadata domain and schemas will be inserted in the repository of the information system as described in sect. **2** and can be queried or browsed using the high-level characteristics of the PSE. We are working to describe ontological metadata to solve the semantic heterogeneity and interpretation problems for communicating a shared and common understanding of some domain across people and computers.

5. – Example of workflows design

MOSÈ is a grid-portal that includes geoprocessing services necessary to the Earth Science community to study natural disasters. It includes services to access geographical data from remote GIS, for modeling and simulation of complex phenomena, for parameter estimation, for spatial data mining and for data visualization.

The current version of the MOSÈ system concerns the analysis of landslide hazard areas. Landslides are complex natural phenomena that are hard to model and simulate. Predicting hazardous events like landslides is particularly difficult because no laboratory exists that can preliminarily measure the necessary variables, refine the techniques, and apply the results. Moreover, landslide simulations need to be accurate and often require massive amounts of computation. The simulation of landslide hazards is particularly relevant for the prevention of natural disasters, since it enables to compute risk map and helps to design protection works. MOSÈ provides tools to improve the overall ability to predict landslides and mud-flows and to develop countermeasures to limit their disastrous consequences. We are applying MOSÈ within a research project that handles the landslide events that have interested the Campania Region in may 1998 in the Sarno area. MOSÈ uses grid and web services technologies to build a service-oriented architecture with which to make interoperate, by standardized interfaces, the different geoprocessing services. The access to services can be obtained by a grid-portal, a web-based user interface, that allows scientists to orchestrate the different services using geo-workflows. MOSÈ is a system with considerable resource demands which arise not only from the inner complexity of its components, but also from complex geo-workflows and usage scenarios in which a substantial number of components instances need to executed - for instance in parameter studies. The orchestration of web services can be conducted in MOSÈ by means of BPEL Designer, an Eclipse Plug-in, and the resultant workflow is sent to the WEB for the execution over the grid.

To better understand the process of geo-workflow creation and orchestration, we illustrated it with a concrete example applied to the problem of the landslide simulation. We are interested to estimate the parameters of the model of the landslide simulation in a subarea of Sarno and to verify if these parameters can be generalized for other neighbouring regions. Our simulation is based on the CA model of debris/mud-flows defined by Di Gregorio and coworkers [8] in the cited project.

Figure 5 shows the workflow designed by the BPEL Designer. The workflow describes the composition of the web services to answer to the user query. The blocks represent the assignation or the composition of input parameters and/or variables and the invocation of web services. Note that, on the bottom right corner of the BPEL Designer, the user can choose the various components extracted from a local repository.

In the workflow, using the parallel flow construct of BPEL, data raster are extracted from two neighbor regions called region 1 and region 2. Then, the parameters on region 1 are estimated. Subsequently, the parameter estimated on the region 1 are used to generate the simulation on the region 2. At this end, the CamelotGrid service is invoked. To verify if the parameters extracted on the region 1 are suitable for the region 2 we

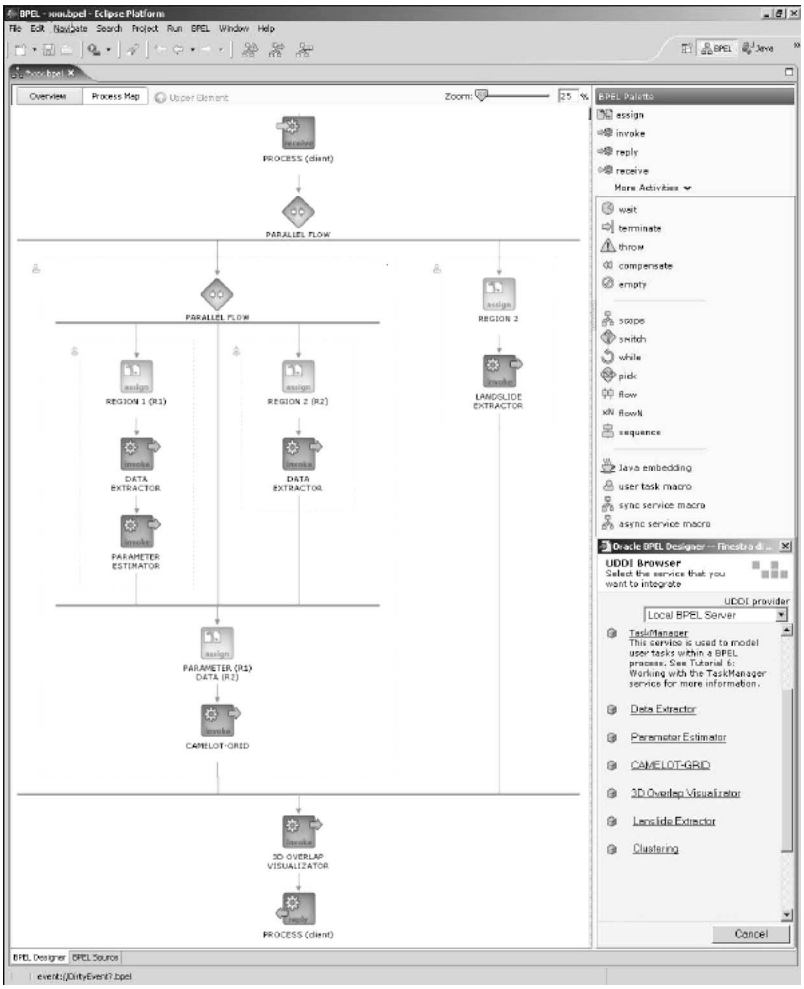

Fig. 5. – Workflow building example using BPEL Designer.

extract the real shape of the landslide from the region 2. At this point, the overlap visualizer component is used to overlap real and simulated landslides. So, graphically, we can confirm (or not confirm) that the parameters of the region 1 are valid for the region 2. Figure 6 shows the final result of the workflow (*i.e.* the output of the overlap visualizer), with real landslide delimited from the white line and the predicted one represented by the filled part.

6. – Conclusions

MOSÈ is a work in progress, but already now demonstrates the vast potential grid and web-based e-science methods will have when routinely available to the broader scientific computing community.

MOSÈ takes advantage of the standardized resource access and workflow support for loosely coupled software components provided by web services. The combination of the

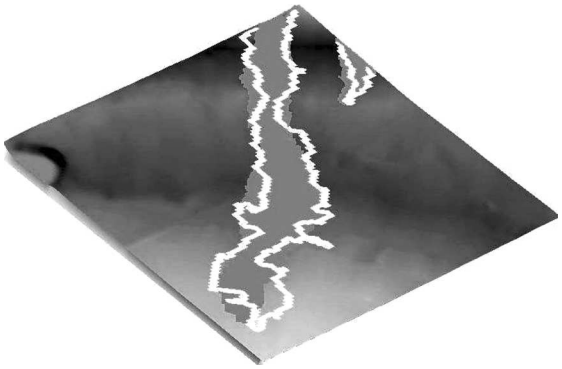

Fig. 6. – Visualisation by means of the Overlap Visualizer of the workflow result.

advanced grid technology with the web interface enables even the unskilled user to run complex, preconfigured environmental simulations without taking care about technical details. The primary advantages of using MOSÈ are the performance gain from being able to use additional resources and the support for the interoperability of data and resources.

For the near future it is planned to migrate MOSÈ to Globus Toolkit v4. As our component model does not require the internals of components to be modified, we expect this to proceed rather smoothly. In the mid term, we will further develop MOSÈ with the goal of building a grid and web-based platform for broad e-science. The application domain of geoprocessing and risk management will remain a main focus of our prototyping and case study work.

* * *

This work was supported by the Environmental Policies Councillorship of the Campania Regional Government and by Project "FIRB Grid.it" (RBNE01KNFP).

REFERENCES

[1] Cannataro M., Comito C., Congiusta G., Folino G., Mastroianni C., Pugliese A., Spezzano G., Talia D. and Veltri P., *A general architecture for grid-based pse toolkits*, in *Workshop on State-of-art in Scientific Computing (PARA'04)* (LNCS, Springer) 2004.

[2] Foster I. and Kesselman C., *The Grid2: Blueprint for a New Computing Infrastructure*, edited by Foster I. and Kesselman C. (Morgan Kaufmann) 2003.

[3] Spezzano G. and Talia D., *Programming cellular automata for computational science and parallel computers*, *Future Generation Computer Systems*, **16** (1999) 206-212.

[4] Kephart J. and Chess D., *The vision of autonomic computing*, *IEEE Computer*, **36** (2003) 41-50.

[5] Folino G. and Spezzano G., *CAMELotGRID: A Grid-based PSE for Autonomic Cellular Applications*, in *13th Euromicro Conference on Parallel, Distributed and Network-based Processing* (IEEE CS) 2005, pp. 206-212.

[6] Foster I. and Kesselman C., *Globus: A Toolkit-Based Grid Architecture*, in *The Grid: Blueprint for a New Computing Infrastructure*, edited by Foster I. and Kesselman C. (Morgan Kaufmann) 1999.

[7] Dail H., Berman F. and Casanova H., *A decoupled scheduling approach for Grid application development environments*, *J. Parallel and Distributed Computing*, **63** (2003) 505-524.

[8] D'ambrosio D., Di Gregorio S., Iovine G., Lupiano V., Rongo R. and Spataro W., *First simulations of the Sarno debris flows through cellular automata*, *Geomorphology*, **54** (2004) 91-117.