AutoGrid: Towards an Autonomic Grid Middleware

Márcio Sekeff, Stanley Sousa, Francisco Silva

Laboratório de Sistemas Distribuídos
Departamento de Informática
Universidade Federal do Maranhão, Brasil

June, 2007
1. Introduction

2. Autogrid Architecture

3. AutoGrid Autonomic Features

4. Conclusions
Motivation

- Computer grids have been used to solve problems in varied areas, empowering the conception of applications that allow combining computations, experiments, observations, and data got in real time;

- The complexity of modern grid environments have turned impracticable its configuration, maintenance and recovery in case of failures solely by human beings, due to:
  - Its dynamism, that can be observed in terms of high variation in resource availability, node instability, and workload variations in nodes and network links;
  - Its heterogeneity, by aggregating a great amount of computation and communication resources, databases and, sometimes, sensors and specific peripherals;
  - Its high scalability, by integrating computational resources spread through several administrative domains.
Providing a greater autonomy to grid systems is one of the greatest challenges for the new generation of grid middlewares;

The term autonomic computing has been used to denote a system that exhibits functional properties, such as:

- Context-awareness: the system must be aware of its execution environment and be able to react to environmental changes;
- Self-Optimization: the system should be able to detect performance degradation and intelligently perform self-optimization actions;
- Self-Healing: the system must be aware of potential problems, and should be able to reconfigure itself in order to continue to function smoothly and to recover from failures using diverse and adaptive failure-handling techniques;
- Self-Configuration: the system must have the ability to dynamically adjust its resources based on its state and the state of its execution environment.
The AutoGrid Project

- Main goal: the development of a robust and self-managing autonomic grid system;
- It is based on the Integrade grid middleware, incorporating autonomic mechanisms to its infrastructure in order to make its configuration and administration independent from human intervention;
- Our research primarily focuses on adding to Integrade four autonomic properties: context awareness, self-healing, self-optimization, and self-configuration.
The Adapta Framework

- Integrate is being augmented with a framework for developing self-adaptive distributed applications;

- The framework decouples the code that governs the business rules from the code responsible for adaptation, based on computational reflection;

- The meta-level receives event notifications describing environmental changes and reacts by applying reconfiguration actions to the base level objects;

- Adapta also has a runtime system that monitors the executing environment and notifies events to registered components whenever a resource availability condition is detected.
The Adapta Framework

- Each adaptive application component also a reconfiguration file that contains the representation of its object model described in a XML based language;

- Application developers or administrators can alter an application component object model by simply modifying its corresponding reconfiguration file;

- Modifications to the object model are interpreted and built on runtime, using the Adaptive-Object Model (AOM) pattern.
The basic unit: cluster, a collection of machines usually connected by a local network;

Clusters can be organized in a hierarchy, encompassing a large number of machines;

Each cluster contains:
  - A Cluster Manager node, responsible for managing the cluster computing resources and for inter-cluster communication;
  - Workstations, which export part of its resources to grid users. They can be shared or dedicated machines.
AutoGrid Main Components

- Application Submission and Control Tool (ASCT): allows users to submit applications and control their execution;

- Application Repository (AR): stores the code of applications that can be executed on the grid;

- Local Resource Manager (LRM): runs in each cluster node. Responsible for instantiating and executing applications scheduled to the node.
AutoGrid Main Components

- Monitoring Service (MS): responsible for monitoring resource properties, such as: available memory, CPU load, network bandwidth and latency, amount of application threads.

- Local Event Service (LES): receives notifications from a collocated MS, notifying relevant resource availability variations;

- Event Processing System (EPS): detects composite events from different event sources (grid nodes);

- Dynamic Reconfiguration System (DyReS): adaptation engine that applies reconfiguration actions to the application base level in response to environmental changes. It also coordinates dependent components during the reconfiguration process.
AutoGrid Main Components

- **Global Resource Manager (GRM):** manages the cluster resources and runs the application scheduler. It is augmented with autonomic capabilities using DyReS adaptation engine;

- **Execution Manager (EM):** maintains information about each application execution, such as its state, executing node, input and output parameters, submission, and conclusion timestamps. Coordinates the application recovery process in case of failures;

- **Stable Storage:** a distributed data repository of applications execution checkpoints.
Context-awareness Mechanism

- The Monitoring Service (MS) regularly inspects the underlying hardware and the executing environment on every grid node using monitor objects (1), which are individually assigned to a single property;
- Monitors can be dynamically instantiated to introduce new monitoring requirements not known at design-time or replaced on the fly;
- Each property has a set of operating ranges. For example, one could use the following operation ranges for monitoring the CPU load usage: [0%, 40%), [40%, 75%), and [75%, 100%]. The MS notifies a collocated Local Event Service (LES) whenever there is a change on the operating range of a property (2).
The reconfiguration process
The LES evaluates if a determined resource availability condition occurred based on a boolean expression provided by the application developer as part of the event definition (3):

UCP: Load \{grt\ 80 \ AND \ MainMemory:Available \ les \ 50; \}

Once an event is detected, LES notifies the Event Processing System (EPS) (4) and all subscribed components (5);

EPS is a distributed event service that detects composite events from different event sources (distributed nodes) (6) and notifies subscribed components embodied by a DyReS instance (7).
AutoGrid Self-configuration Mechanism

- Each adaptive component must instantiate a meta-level called DyReS;
- It is responsible for receiving event notifications (from LES or EPS) and for applying reconfiguration actions on behalf of its component;

**Implemented reconfiguration actions:**

1. Dynamic change of application parameters;
2. Dynamic replacement of application algorithms with a well-defined state transfer protocol.

- DyReS can be extended by expert users to introduce new actions, such as component addition, removal, or replacement.
AutoGrid Self-configuration Mechanism

- DyReS uses the ComponentConfiguration architecture, maintaining references to the components it depends on (called hooks), as well as references to dependent components (called clients);
- The reference chain is used for synchronizing reconfiguration actions between application components (8 to 13);
- Example: applying a video compression algorithm to a video stream.
The reconfiguration process
AutoGrid Self-healing Mechanism

- The dynamic nature of the grid infrastructure, its high scalability, and great heterogeneity exacerbates the likelihood of errors occurrence;
- Diverse failure handling techniques are applied:
  - Retrying: restarts an application from scratch;
  - Replication: submits the same application for execution a number of times, generating various application replicas;
  - Checkpointing: periodically saves the state of the computation in a stable storage during the failure free execution time. Upon a failure, the application restarts from the last saved point.
- Considering the objective of minimizing the application response time, experiments demonstrated that the best failure handling technique varied as we altered environment parameters such as the MTBF and the downtime.
AutoGrid Replication Mechanism

- The Global Resource Manager (GRM) decides the amount of replicas to be generated for a given application submission based on the execution environment MTBF and the application mean execution time;
- It also runs the scheduler, assigning different nodes for application replicas;
- The Monitoring System periodically computes the cluster MTBF based on the data stored by the Execution Manager (EM);
- An event is generated whenever a significant change is detected, updating GRM MTBF parameter;
- We measured the benefits of varying the amount of replicas with a set of simulations.
AutoGrid Self-optimization Mechanism

- The computational grid comprise a highly dynamic environment (amplified on opportunistic grids, since nodes are not dedicated);
- In this context, load balancing techniques, adaptive scheduling, and dynamic re-scheduling are important topics to be investigated;
- Dong et all refers to an adaptive solution to the scheduling problem as the one in which the algorithms and parameters used to make scheduling decisions change dynamically;
- Maheswaran et all quantifies the relative performance of scheduling heuristics. Simulations indicated that the most appropriate algorithm varied as the environment changes;
- The Adapta framework provides the means for dynamic switching the GRM scheduling algorithm, including a state transfer mechanism;
- Investigations are being made to measure the exact moment of replacement, according to the application arrival rate, resource usage, network status and other environmental properties.
Conclusions

- This paper presented AutoGrid, an initiative towards an autonomic grid system that uses Integride grid middleware as its foundation;
- The autonomic features includes context-awareness, self-configuration, self-healing, and self-optimization;
- AutoGrid design emphasizes openness and flexibility, allowing the modification of a component object model on the fly. The adaptation engine is extensible, allowing advanced users to design new adaptive mechanisms;
- AutoGrid autonomic features are being evaluated through experiments and simulations, measuring the benefits to the grid infrastructure.