A Generic Architecture for Supervision of Distributed Applications

A. SADOVYKH

Laboratoire d'Informatique, Université Paris 6 (LIP6), 8 rue du Capitaine Scott, 75015 Paris, France
Andrey.Sadovkykh@lip6.fr

S. Wesner

High Performance Computing Centre Stuttgart (HLRS), Allmandring 30, 70550 Stuttgart, Germany
wesner@rus.uni-stuttgart.de

J.-E. BOHDANOWICZ

European Aeronautic Defence and Space Company - Launch Vehicles (EADS-LV),
66, route de Verneuil, BP 3002, 78133 Les Mureaux, France
JeanEric.Bohdanowicz@launchers.eads.net

Abstract

Application Development is moving from monolithic systems towards distributed architectures based on middleware technologies such as Web Services. Such a kind of application requires a sophisticated monitoring framework that provides information on all levels from hardware over network up to the application level. Existing monitoring solutions do not provide such an integrated view. The GeneSyS IST project (IST-2001-34162) aims at the specification and development of such a framework. The basic concepts developed so far are presented within this position paper.

1. MOTIVATION

The goal of the GeneSyS (Generic System Supervision) IST project (IST-2001-34162) [1] started on March 2002 is to specify and develop a new supervision middleware for distributed systems and applications. The need for such a framework arises from the fact that distributed applications are getting more and more accepted and used. Especially the introduction of Web Services [2] driven by the global players pushed the interest in distributed applications further. In order to reflect the requirements of different areas within the architecture the consortium consists out of academic and industrial partners from different domains. The partners are EADS Launch Vehicles (France), D3 Group GmbH (Germany), HLRS (Germany), MTA SZTAKI (Hungary) and NAVUS GmbH (Germany).

2. DRAWBACKS OF EXISTING SUPERVISION SYSTEMS

The major drawback of most of supervision tools [8] is their closed and proprietary nature and their limited view on a specific domain of monitoring such as the network. The major problems are:

- The interfaces are not open, i.e. they are neither documented nor publicly available.
- The inter-agent protocols are often not standardised and based on operating system dependent protocol encoding.
- The most of the existing supervision solutions are aimed to a specific domain of monitoring such as network monitoring or designed to monitor a specific commercial application. This lack of generality does not allow monitoring of the whole system from the network level up to application level, groupware, and workflow.
- The inflexible architecture when transportation core, agents, consoles are not separated makes

difficult integration with 3d party supervision tools and to instrument applications in order to integrate them into the monitoring system.

GeneSyS should give to the researches and developers the appropriate framework to solve these problems. Following sections describe in more details how GeneSyS is supposed to ensure generality, interoperability and reliability of supervision in heterogeneous environments.

3. OBJECTIVES OF GENESYS

The top-level objectives of the GeneSyS project are:

1. To specify and develop an open, generic, modular and comprehensive supervision concept,
2. To integrate and validate this supervision structure within various industrial contexts,
3. To achieve the adoption of the GeneSyS concepts by the stakeholders, and to ensure that the vision of the proposed generic structure will become a new emerging standard.

The basic concept as shown in figure 1 is to have two layers within the architecture. The first layer is responsible for the connection of the GeneSyS middleware with external entities that provide data that can be monitored. This layer uses the means that allow the collection of data. This can range from analysing log files up to using bi-directional protocols such as the Simple Network Management Protocol (SNMP) in order to collect the data. The second layer called the core or GeneSyS middleware provides basic services such as registering and locating entities that can be monitored.

FIG 1: Building blocks of GeneSyS

4. VALIDATION SCENARIOS

GeneSyS will be validated using several scenarios. The first two scenarios will be held for the prototype version 1 (planned for March 2003). A Distributed Training scenario will be held for the prototype version 2 (planned for September 2004).

The overall goal is to validate the architecture against different applications in different domains in order to prove the concept to be generic and applicable.

The following sections describe the scenarios planned for the validation.

4.1 Distributed Engineering Scenario (Preliminary Design Review)

The Preliminary Design Review (PDR) application is property of EADS-LV. It is used for collaborative work during essential points of the Automated Transfer Vehicle (ATV) design : the design reviews. These reviews involve up to 200 reviewers. During this process, the reviewers review the design, create Review Item Discrepancies (RIDs), meet on-line using groupware tools to discuss/accept/decline RIDs. GeneSyS provides outstanding capability to enhance the PDR application with supervision. Supervision agents will be used at application, groupware, network, system levels to help administrators to maintain this complex distributed system.

4.2 Automotive Scenarios

The two proposed Automotive Scenarios will show two kinds of use of GeneSyS :

- the supervision of a workflow process at BMW facilities,
- and the supervision of collaborative working process during crash tests analysis in collaboration with GECI.

The first scenario aims at supervising a workflow system that is guiding the change management for the engine control units software of the BMW motors.

The second one aims at supervising a collaborative working process between Radioss (simulation software) experts and the automotive manufacturer in a remote way using the GTI6 collaborative working tool [7].

4.3 Distributed Training Scenario

The distributed training pilot application will simulate real use-case in the space domain. The scenario will interconnect several complex real-time simulators in the frame of a space mission rehearsal. It is essential to simulate the contingency situations and prepare in advance the astronauts and the ground controllers to these cases. The trainees will be plugged directly to the simulators to complete "native" simulations. GeneSyS will allow the instructors to supervise the simulations and training process in the real-time. The technical operators will benefit; from standard agents for system, network and groupware. It will help them to support this complex heterogeneous system.

5. PRELIMINARY ARCHITECTURE

The architecture for the GeneSyS system consists out of components with a different level of generality. There will be components that provide basic services that can be used for all kind of applications or can be seen as integral part of the GeneSyS middleware. As each of the above described validation scenarios have not only common requirements but also domain and application specific requirements there will be also components that are dedicated to support a specific application but can utilize the functionality of the common components.

5.1 Collecting Agents

A collecting agent is an agent that implements an interface to a monitored entity. These agents are able to provide the data that is available by the monitored entity to the Connector in order to get published via

GeneSyS to other agents.

Several types of Collecting Agents have already been identified for the first validation scenarios. GeneSyS does not specify how an agent must collect the data from the monitored entity. So it is possible that different implementations of a System Monitoring Agent Interface uses completely different mechanisms for retrieving the data. On a Microsoft Windows operating system the data retrieval would be performed using special Windows API functions as under Linux other operating system functions would be used.

The following table lists a set of identified agents together with a short description of their functionality.

Agent Name:	Description:
System Monitoring Agent	The system monitoring agent collects the hardware status information of a system. This can be information on the current load situation expressed in CPU load and memory utilisation also the performance values from external hardware such as disks.
Database Monitoring Agent	This agent monitors the data available from a DBMS (Database Management System). It provides information on the server, the type of the DBMS system and the current number of users and processes of a database.
Network Component Monitoring Agent	This component almost completely maps to the data that is provided via SNMP either from passive components such as Ethernet cards and also active components such as IP routers.
Connection Quality Agent	The connection quality agent measure parameters of IP connections between hosts. As all scenarios within GeneSyS are distributed applications the quality of the connection between these components is important for the operation of these applications.
Server Application Agent	A distributed application often involves several servers. Each of these servers must be supervised. Examples for values collected from this kind of component is the number of clients connected to the server, the average response time and the availability of services.
Client Application Agent	The client application agent is monitoring the client part of the distributed application. The information collected is linked to the client supervised by the agent and, unlike the server agent, to the user that is using the client.
Groupware Application Agent	The groupware application agent's duty is to monitor H.323/T.120 based multimedia conferencing application. A H.323/T.120 conference consists of a host and a number of callers. The host participant (initiator of the conference) provides the node controller that manages participants and their applications, therefore the host terminal is a good candidate of monitoring.

5.2 Complex Agents

Beside the basic components that retrieve the data from hardware, network, middleware and applications elements with more comprehensive functionality is needed. These components use the basic agents for collecting the raw data and provide themselves aggregated or derived information to other agents or clients. As an example a Simulation Agent can use a configuration for a distributed simulation that contains the information on the basic collecting agents for the involved systems, network connections and applications. The Simulation Agent derives from the data gained from the basic agents the information if the simulation is performing well or not. Of course this kind of scenario allow multiple levels where complex agents relies on other complex agents.

Another task of this kind of agent can be to act as an intermediate component between the collecting agent and the clients requesting the data. As a general goal the monitoring should not affect the overall performance of the system under monitoring. A Proxy component that collects the data on a regular basis from the collecting component caches the data and provides the data on behalf of the collecting agent reduce the load for the collecting agent. Many other scenarios are possible such as adding access control and encryption. So a Complex Agent is a "Consumer" as it collects the data and also a "Data Provider" for other consuming components.

5.3 Web Services as implementation technology

The elements that must be supervised within the scenarios are implemented in different programming languages and run on different kind of operating systems. Therefore a platform and programming language neutral infrastructure is needed. The Common Object Request Broker Architecture (CORBA) offers this functionality but requires the installation of an Object Request Broker (ORB) on the client and on the server side. Web Services [2] promise to fulfil the requirements above and are easy to deploy and use. GeneSyS will use Web Service Technology for implementing the different agents and the middleware components.

6. APPLICABILITY TO GRID COMPUTING

Also within the GRID [1] community the need for a monitoring architecture for distributed systems has been identified. The Grid Monitoring Architecture Working Group [5] is currently working on the standardisation of a Grid Monitoring Architecture within the Global Grid Forum (GGF) [6]. Although the architecture discussed here is not only aiming at GRID environments we believe this architecture can also be useful within this context. In fact GRID environments can be seen as an additional validation scenario. As GRID computing is still oriented towards High Performance Computing there will be of course different monitored entities such as a "Queuing System Monitor" and corresponding agents. But we think the requirements for supervision do not differ conceptually. In fact the architecture currently proposed within the GGF consists out of Consumer, Producer and ConsumerProducer components that can be mapped to the proposed Collecting Agent, Complex Agents and Client Agents. On the technology level the integration will be seamless as the GRID infrastructure is about to change to a Web Service oriented basis as discussed in [3] and [4].

7. CONCLUSIONS

We have shown in several scenarios the need for a distributed monitoring framework. Even if the applications analysed are from very different domains a common architecture can be derived from the requirements. The architecture is itself a distributed application and consists out of a dynamic set of Monitoring Agents that are responsible for collecting a specific set of data. The architecture is open to be extended with Monitoring Agents for all kind of Monitored Entities. It is planned to publish the GeneSyS middleware and also Monitoring Agents and Clients under an open source policy as it is available.

REFERENCES

- [1] GeneSyS project web-site <http://genesys.sztaki.hu/>
- [2] WebService Activity of W3C, <http://www.w3.org/2002/ws/>
- [3] Foster et al., Grid Services for Distributed System Integration, <http://www.globus.org/research/papers/ieee-cs-2.pdf>
- [4] I. Foster et al., The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, tech. report, Globus Project; <http://www.globus.org/research/papers/ogsa.pdf>
- [5] Grid Monitoring Architecture Working Group, <http://www.didc.lbl.gov/GGF-PERF/GMA-WG/>
- [6] Global Grid Forum, <http://www.gridforum.org>
- [7] GTI6 web site, <http://www.gti6.com>
- [8] Gunter Rackl, Monitoring and Managing Heterogeneous Middleware, Ph.D. thesis, Technische Universitat München, 2001

© EADS-LV, HLRS, LIP6 2002