Active Networking Support for the Grid

Laurent Lefèvre[1], Cong-duc Pham[1], Pascale Primet[1], Bernard Tourancheau[2],
Benjamin Gaidioz[1], Jean-Patrick Gelas[1], and Moufida Maimour[1]

[1] RESAM Laboratory, Université Claude Bernard Lyon1, Action INRIA RESO
Ecole Normale Supérieure de Lyon, 46 Allée d'Italie, 69364 Lyon Cedex 07, France
Laurent.Lefevre@inria.fr
[2] SUN Labs Europe, 29 Chemin du vieux chêne, 38240 Meylan, France

Abstract. Grid computing is a promising way to aggregate geographically distant machines and to allow them to work together to solve large problems.
After studying Grid network requirements, we observe that the network must take part of the Grid computing session to provide intelligent adaptative transport of Grid data streams.
By proposing new intelligent dynamic services, active network can be the perfect companion to easily and efficiently deploy and maintain Grid environments and applications.
This paper presents the Active Grid Architecture (A-Grid) which focus on active networks adaptation for supporting Grid environments and applications.
We focus the benefit of active networking for the grid on three aspects: High performance and dynamic active services, Active Reliable Multicast, and Active Quality of Service.

1 Introduction

In recent years, there has been a plethora of interest on Grid computing which is a promising way to aggregate geographically distant machines and to allow them to work together to solve large problems. Most of proposed Grid frameworks are based on Internet connections and do not make any assumption on the network. Grid designers only take into account of a reliable packet transport between Grid nodes and most of them choose TCP/IP protocol.

But one of the main complaint of Grid designers is that networks do not really support Grid applications.

Meantime, the field of active and programmable networks is rapidly expanding. These networks allow users and network designers to easily deploy new services which will be applied to data streams. While most of proposed systems deal with adaptability, flexibility and new protocols applied on multimedia streams (video, audio), no active network efficiently deal with Grid environments.

In this paper we try to merge the both fields by presenting The Active Grid Architecture (A-Grid) which focus on active network adaptation for supporting Grid environments and applications. This active Grid Architecture proposes

I.W. Marshall, S. Nettles, and N. Wakamiya (Eds.): IWAN 2001, LNCS 2207, pp. 16–33, 2001.

solutions to implement the two main kind of Grid configurations: Meta-cluster computing and global computing. In this architecture the network takes part of the Grid computing session by providing efficient and intelligent services dedicated to Grid data streams transport.

We focus on the benefit of Grid active networking for: High performance and dynamic services deployment, Reliable Multicast and Quality of Service.

This paper reports on our experience in designing an Active Network support for Grid Environments. First we classify, the Network Grid requirement depending on environments and applications needs (section 2). In section 3 we propose the Active Grid Architecture. We focus our approach by providing support for the most network requirements from Grid: High performance transport (section 4), End to end Grid QoS services (section 5) and reliable multicast (section 6). We conclude and present our future works in last section.

2 Network Requirements for the Grid

A distributed application running in a Grid environment requires various kind of data streams: Grid control streams and Grid application streams.

2.1 Grid Control Streams

First of all, we can classify the two basic kind of Grid usage:

- Meta cluster computing:
  A set of parallel machines or clusters are linked together with Internet to provide a very large parallel computing resource. Grid environments like Globus[13], MOL[24], Polder[2] or Netsolve[6] are well designed to handle meta-cluster computing session to execute long-distance parallel applications.
  We can classify various network needs for meta-clustering sessions:
  - Grid environment deployment: The Grid infrastructure must be easily deployed and managed: OS heterogeneity support, dynamic topology reconfiguration, fault tolerance.
  - Grid application deployment: Two kind of collective communications are needed: Multicast and gather. The source code of applications is multicast to a set of machines in order to be compiled on the target architectures. In case of Java based environments, the bytecode can be multicast to a set of machines. In case of an homogeneous architecture, the binaries are directly sent to distant machines. After the running phase, results of distributed tasks must be collected by the environment in a gathering communication operation.
  - Grid support: The Grid environment must collect control data: node synchronization, node workload information. The information exchanged are also needed to provide high-performance communications between nodes inside and outside the clusters.

- Global or mega-computing: These environments usually rely on thousand of connected machines. Most of them are based on computer cycles stealing like Condor[20], Entropia[1], Nimrod-G[10], or XtremWeb[3].
  We can classify various network needs for Global-computing sessions:
  - Grid environment deployment: Dynamic enrollment of unused machines must be taken into account by the environment to deploy tasks over the mega-computer architecture.
  - Grid application deployment: The Grid infrastructure must provide a way to easily deploy and manage tasks on distant nodes. To avoid the restarting of distributed tasks when a machine crashes or become unusable, Grid environments propose check-pointing protocols, to dynamically re-deploy tasks on valid machines.
  - Grid support: Various streams are needed to provide informations to Grid environment about workload informations of all subscribed machines. Machine and network sensors are usually provided to optimize the task mapping and to provide load-balancing.

Of course, most of environments work well on both kind of Grid usage like Legion[18], Globus[13], Condor[20], or Nimrod-G[10].

2.2 Grid Application Streams

A Grid computing session must deal with various kind of streams:

- Grid application input: During running phase, distributed tasks of the application must receive parameters eventually coming from various geographically distant equipments (telescopes, biological sequencing machines,...) or databases (disk arrays, tape silos,...).
- Wide-area parallel processing: Most of Grid applications consist of a sequential program repeatedly executed with slightly different parameters on a set of distributed computers. But with the emergence of high performance backbones and networks, new kind of real communicating parallel applications (with message passing libraries) will be possible on a WAN Grid support. Thus, during running phase, distributed tasks can communicate data between each others. Applications may need efficient point to point and global communications (broadcast, multicast, gather,...) depending on application patterns. These communications must correspond to the QoS needs of the Grid user.
- Coupled (Meta) Application: They are multi-component applications where the components were previously executed as stand-alone applications. Deploying such applications must guarantee heterogeneity management of systems and networks. The components need to exchange heterogeneous streams and to guarantee component dependences in pipeline communication mode. Like WAN parallel applications, QoS and global communications must be available for the components.

Such a great diversity of streams (in terms of messages size, point to point or global communications, data and control messages,...) requires an intelligence in the network to perfectly support Grid requirements.

3 Active Grid Architecture

We propose an active network architecture dedicated to Grid environments and Grid applications requirements: The A-Grid architecture.

An active grid architecture is based on a virtual topology of active network nodes spread on programmable routers of the network. Active routers, also called Active Nodes (AN), are deployed on network periphery.

Contrary to a wide active routers deployment approach and to guarantee high performance packets transport, we do not believe in the deployment of Gigabit active routers in backbones. If we consider that the future of WAN backbones could be based on all-optical networks, no dynamic services will be allow to process data packets. So, we prefer to consider backbones like high performance well-sized passive networks. We only concentrate active operations on edge routers/nodes mapped at network periphery.

Active nodes are connected between each other and each AN manage communications for a small subset of Grid nodes. Grid data streams cross various active nodes up to passive backbone and then cross another set of active nodes up to receiver node. The A-Grid architecture is based on Active Node approach: Programs, called services, are injected into active nodes independently of data stream. Active nodes apply these services to process data streams packets. Services are deployed on demand when streams arrive on an active node.

3.1 Active Grid Architecture

To support most of Grid applications, the Active Grid architecture must deal with the two main Grid configurations:

- Meta cluster computing (Fig. 1):
  In this highly coupled configuration, an active node is mapped on network head of each cluster or parallel machine. This node manage all data streams coming or leaving a cluster. All active nodes are linked with other AN mapped at backbone periphery. An Active node delivers data streams to each node of a cluster and can aggregate output streams to others clusters of the Grid.
- Global or Mega computing (Fig. 2):
  In this loosely coupled configuration, an AN can be associated with each Grid node or can manage a set of aggregated Grid nodes. Hierarchies of active nodes can be deployed at each network heterogeneity point.
  Each AN manages all operations and data streams coming to Grid Nodes: subscribing operations of voluntary machines, results gathering, nodes synchronization and check-pointing.

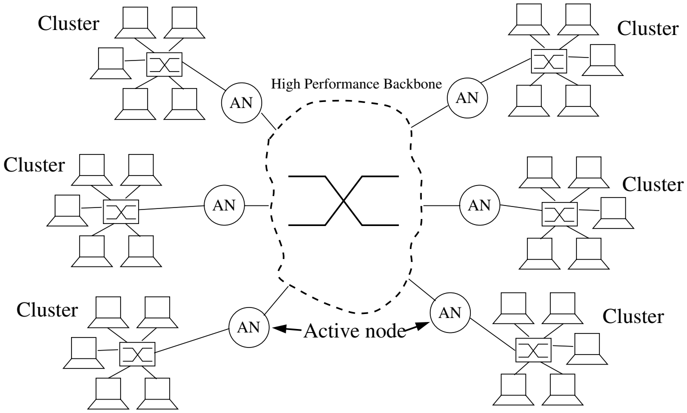

Fig. 1. Meta Cluster Computing Active Grid Architecture.

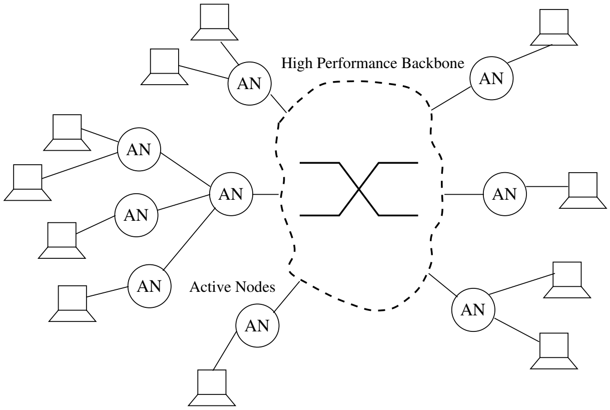

Fig. 2. Global Computing Active Grid Architecture.

For both configurations, active nodes will manage the Grid environment by deploying dedicated services adapted to Grid requirements: management of nodes mobility, dynamic topology re-configuration, fault tolerance.

3.2 Active Network Benefits for Grid Applications

Using an Active Grid architecture can improve the communications needs of Grid applications:

- Application deployment: To efficiently deploy applications, active reliable multicast protocols are needed to optimize the source code or binary de-

ployment and the task mapping on the Grid configuration accordingly to resources managers and load-balancing tools. An active multicast will reduce the transport of applications (source code, binaries, bytecode,. . .) by minimizing the number of messages in the network. Active node will deploy dedicated multicast protocols and guarantee the reliability of deployment by using storage capabilities of active nodes.
- Grid support: The Active architecture can provide informations to Grid framework about network state and task mapping. Active nodes must be open and easily coupled with all Grid environment requirements. Active nodes will implement permanent Grid support services to generate control streams between the active network layer and the Grid environment.
- Wide-area parallel processing: With the emergence of grid parallel applications, tasks will need to communicate by sending computing data streams with QoS requests. The A-Grid architecture must also guarantee an efficient data transport to minimize the software latency of communications. Active nodes deploy dynamic services to handle data streams: QoS, data compression, "on the fly" data aggregation.
- Coupled (Meta) Application: The Active architecture must provide heterogeneity of services applied on data streams (data conversion services,. . .). End to end QoS dynamic services will be deployed on active nodes to guarantee an efficient data transport (in terms of delay and bandwidth).

Most of services needed by Grid environments: High performance transport, dynamic topology adapting, QoS, on-the-fly data compression, data encryption, data multicast, data conversion, errors management must be easily and efficiently deployed on demand on an Active Grid architecture. To allow an efficient and portable service deployment, we will present in next section our approach to propose an active network framework easily mergeable with a Grid environment: The Tamanoir Framework. Then to resolve the main network Grid requirements identified in the previous section, we focus our approach on the two major services needs: QoS and reliable multicast.

4 High Performance and Dynamic Service Deployment

We explore the design of an intelligent network by proposing a new active network framework dedicated to high performance active networking. The Tamanoir[1] framework [17] is an high performance prototype active environment based on active edge routers. Active services can be easily deployed in the network and are adapted to architecture, users and service providers requirements.

A set of distributed tools is provided: Routing manager, active nodes and stream monitoring, web-based services library. Tamanoir is based on compiled JAVA/GCJ [16] with multi-threading approach to combine performance and

[1] Tamanoir (*great anteater*) is one of the strangest animal of south America only eating ants (30000 daily). We choose this animal in reference to the well-known active ANTS [27] system.

portability of services, applications can easily benefit of personalized network services through the injection of Java code.

4.1 Overview of a Tamanoir Node

An active node is a router which can receive packets of data, process them and forward them to other active nodes.

A Tamanoir Active Node (TAN) is a persistent daemon acting like a dynamic programmable router. Once deployed on a node, it is linked to its neighbors in the active architecture. A TAN receives and sends packets of data after processing them with user services. A TAN is also in charge of deploying and applying services on packets depending on application requirements. When arriving in a Tamanoir daemon, a packet is forwarded to service manager (figure 3). The packet is then processed by a service in a dedicated thread. The resulting packet is then forwarded to the next active node or to the receiver part of application according to routing tables maintained in TAN.

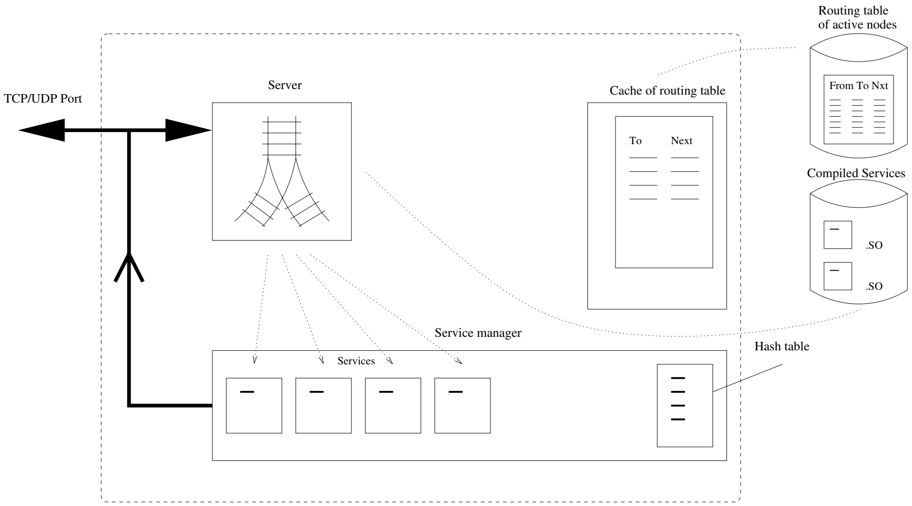

Fig. 3. TAN: Tamanoir Active Node.

4.2 Dynamic Service Deployment

In Tamanoir, a service is a JAVA class containing a minimal number of formatted methods (*recv()* and *send()* to receive a packet, apply a code on it and send the packet to another TAN, to the receiving application or even severals in the context of multicast service). Actually, each service used by an application is inherited from a generic class called simply *Class Service*. We have used this technique in order to simplify the design of future services and especially to allow the TAN to download dynamically a new class.

In each packet we find a label (or a tag) representative of the last TAN crossed by the packet. Therefore, if a TAN does not hold the appropriate service, a downloading operation must be performed.

In figure 4, we can observe three kind of service deployment. The first TAN crossed by a packet can download the useful service from either the transmitting application, or from a service broker. By using an *http address* in service name, TAN contact the web service broker, so applications can download generic Tamanoir services to deploy non-personalized generic services. After, next TANs download the service from a previous TAN crossed by packet or from the service broker.

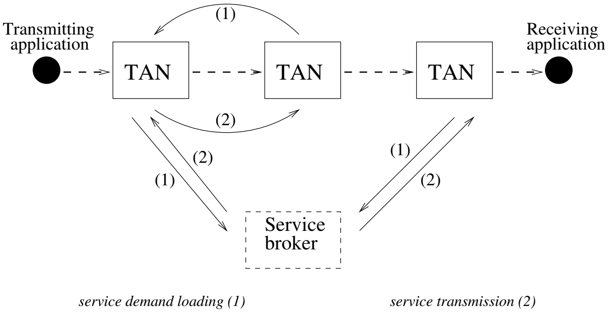

Fig. 4. Dynamic Service Deployment.

4.3 Experiments

We based our first experiments of Tamanoir on Pentium II 350 MHz linked with Fast Ethernet switches and compared Tamanoir system to the ANTS [27] most developed active network system.

Results presented in figure 5 show the delay needed to cross an active node (latency). While ANTS needs 3 ms and is dependent of capsule payload size; Tamanoir time remains constant with a latency of 750 μs. Meanwhile, ANTS process capability remains weak while Tamanoir goes 3 times faster.

These first experiments show that Tamanoir framework can support a Grid environment without adding to much latency to all data streams. So Tamanoir can efficiently deploy services on active nodes depending on Grid requirements: QoS, data conversion, multicast,“on the fly” data compression. Next sections will focus on two main kind of services: QoS and multicast.

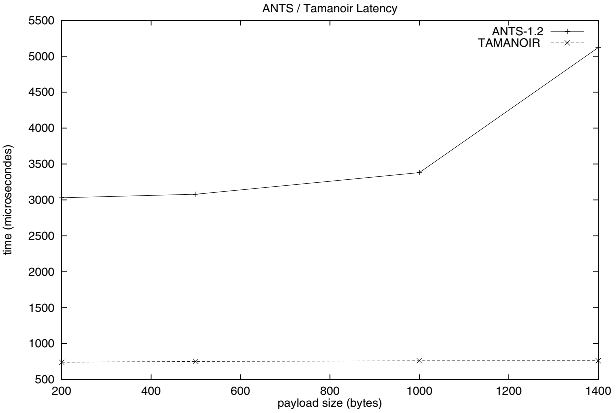

Fig. 5. Latency: Cost to Cross an Active Node.

5 Active Grid Quality of Service

In this part, we focus on the QoS problematic of the Grid flows and try to present the opportunities the active network technology offers to the QoS management and control of this type of streams. We study in particular how the solutions proposed for the multimedia applications can be derived to meet the specific requirements of the Grid applications.

5.1 What Is Quality of Service?

Quality of Service (QoS) represents the set of those quantitative and qualitative characteristics necessary to achieve the required functionalities of an application. In the Network community, QoS is a set of tools and standards that gives network managers the ability to control the mix of bandwidth, delay, variance in delay (jitter) and packet loss. Controlling this mix allows to provide better and more predictable network service .

The problem of QoS appears in the Internet since it has become the common infrastructure for a variety of new applications with various requirements for QoS guarantees. The traditional best-effort model has not been designed to support time-sensitive and heterogeneous traffic. The emergence of multimedia applications requires new QoS solution.

The first step is to introduce the capabilities required to support QoS in the Internet infrastructure and developing mechanisms and algorithms that scale while enabling a wide range of QoS guarantees. The second step is to enable users and applications to access these new capabilities. This last task is quite difficult and can be considered as one of the main reasons for the relatively slow deployment of QoS in the IP networks.

Three types of QoS guarantees (or end-to-end QoS level) are proposed by an IP network: Best-effort, statistical guarantees or strict guarantees (absolutes). To obtain the required end-to-end QoS different approaches are possible. They can be divided into two groups: The pure in-network mechanisms and the end-to-end mechanisms. The pure in-network mechanisms are based on resource reservation or on class-based services (like IntServ [8] or DiffServ [7]). The main problem of the IntServ architecture is the scalability. This solution requires that control and forwarding state for all flows are maintained in routers. The DiffServ architecture appear to be a more scalable and manageable architecture. It is focusing not on individual flows but on traffic aggregates, large sets of flows with similar service requirements. The service differentiation goals are to accommodate heterogeneous applications requirements and users expectations and to permit differentiated pricing of Internet service. The DiffServ model requires that complex classification and conditioning functions are implemented only at network boundary nodes, and that per-hop behaviors are applied to aggregates of traffic which have been appropriately marked using the DS field [22] in the IP header.

Until now, neither IntServ nor DiffServ seems to offer the unique solution for all the various requirements. The aim of an end-to-end QoS mechanisms is to mask the deficiencies of the network QoS. For classical data transmission on a best-effort IP network, the role of the end-to-end TCP protocol is error control and error recovery by retransmission of the lost packets. For multimedia applications, QoS mechanisms for adaptability (such as the forward error correction (FEC)) are incorporated in the adaptive application itself and not in the transport protocol (RTP [4]). The advantages of this adaptive approach is that the application monitors the experimented QoS, and can detect variation and react appropriately.

5.2 Grid QoS

The QoS performances requirements of Grid streams are more disparate and flexible than for multimedia application. In traditional QoS approach, the streams specification includes quantitative parameters that can be classified in performance parameter (bit rate), temporal characteristics (delay, jitter (delay variation)), integrity parameters (loss rate and error rate).

If we suppose that some kind of QoS will be available in the near future in the Internet, one can ask if the Grid applications will benefit from the currently proposed QoS services. For example, the Grid community is interested by a guarantee on the delivery of a complete bulk data file, but not by the priority of each individual packet. This service differs from traditional QoS offerings in that the user specifies the ultimate delivery time when the data transfer must complete. To ensure that the transfer completes on time it is necessary to determine when the transfer should start and to control the transfer of the individual packets.

An other Grid QoS service should for example provides information about the achievable throughput and about the stability-level for data-delivery between two points in the network. Network and throughput measurements are central to the Grid QoS problematic.

An other key component for the Grid is the dynamic generation of performance forecasts. For example, the Network Weather Service (NWS) [28] periodically takes measurements of the load of resources it has to monitor and uses them to generate performance forecasts. One of the NWS sensors is the network sensor [15] whose aim is to take measurements that represent the network quality in term of latency and bandwidth. The different components of the NWS are distributed on monitored hosts.

The GARA (Globus Architecture for Reservation and Allocation) [14] provides advance reservations and end-to-end management for quality of service on different types of resources (network, storage and computing). This architecture remains a traditional solution, in the sense that the processing task associated with transport purpose are performed on the end systems (reservation and adaptation).

5.3 The Active Grid QoS Approach

In our active Grid QoS approach, we study how to implement new services in the active edge nodes. We propose an active QoS model cumulating the advantages of IntServ, DiffServ and the end-to-end adaptation mechanisms. Our active QoS approach allows to:

- enlarge the QoS tools spectrum by processing on the individual flows,
- maintain a scalable QoS approach like DiffServ in the core network,
- realize a dynamic and efficient adaptation at the edge according to the real state of the network.

Since end-to-end advanced network resource reservation is impossible on the Internet, we argue that for Grid flows, dynamic and specific adaptation is required. For this, an active Grid QoS service should provide the user the ability to characterize a flow in term of end-to-end delay or end-to-end loss rate.

It is also necessary to know the relative importance of a packet in order to know what to do with it in different network condition: dropping, slowing, storing, duplicating. In the congested nodes the time constrained flows must be treated in priority. The active nodes can have a finer vision of the individual data streams, and can react immediately to congestion and implement appropriate packet discard for each stream.

In the Tamanoir architecture, capsules are transported. Data capsules can carry different types of information, that can be used for processing during the travel:

- semantics from the application (type of payload, end-to-end target QoS performances) This information can be interpreted as an Active DiffServ code point (ADSCP). This code point is analogous to the DSCP of the traditional DiffServ model but can be application specific. This information characterizes the flow with high level and end-to-end information which is application specific and easier to handle than token bucket specifications in a resource reservation protocol like RSVP [9],

- self transfer monitoring information (eg. cumulated time of transfer),
- state of the already crossed routers (heavy loaded, congested, etc.). This information similar to an ECN (Explicit Congestion Notification) can be processed by the active nodes on the way.

QoS monitoring active services are associated to this QoS model and indicate the state of the nodes and network performances between two active nodes.

On Tamanoir we have developed several prototypes of active QoS services:

- an active QoS adaptation service,
- an active DiffServ service,
- an active monitoring service.

These prototypes have been realized to demonstrate the ability of the Tamanoir approach for providing Intelligent QoS mechanisms on a classical best-effort IP network or on a DiffServ IP network. They have been validated on our local platform.

The active QoS services we propose are able to modify the carried data during their travel in the network. A QoS adaptation can be made "on the fly". This adaptation is function of the performance experienced by the packets of each particular flow. QoS adaptation means dropping , filtering operation (dynamic rate shaping, QoS filters) but also data staging. QoS signaling like informing the user/end application of degradation is an other task performed by the active agents. This adaptive approach is more efficient than the traditional adaptive application philosophy which regulates the flow according to report from the receiver. The latency of the reaction to congestion can be important and it can be dramatic especially if the application throughput is very high. In an active approach, the overload situation can be anticipated by active QoS monitoring and the QoS adaptation, located at the active edge router, made closer to overflowed router. The reaction to a congestion is then faster and the global QoS improved.

At the deployment of the Grid architecture, specific services are downloaded and activated in edge active nodes. Ones are QoS monitoring agents responsible of the QoS parameters measurement. Other services, QoS adapters, intercept and process the flows when necessary. The agents are able to exchange reports and to communicate with hosts.

Grid QoS services based on bandwidth requirements concern more applications deployment and large parameters transfer between nodes. Delay based QoS services will concern Grid control streams (workload, fault tolerance, etc.) and data streams of pipelined coupled applications.

6 Active Reliable Multicast for the Grids

6.1 Reliable Multicast for the Grid

Multicast is the process of sending every single packet to multiple destinations. Motivations behind multicast facilities are to handle one-to-many communications in a wide-area network with the lowest network and end-system overheads.

In contrast to best-effort multicast, that typically tolerates some data losses and is more suited for real-time audio or video for instance, reliable multicast requires that all packets are safely delivered to the destinations. Desirable features of reliable multicast include, in addition to reliability, low end-to-end delays, high throughput and scalability.

These characteristics fit perfectly the grid computing community as communications in a grid make an intensive usage of data distribution and collective operations. In a very simple grid session, an initiator sends data and control programs to a pool of computing resources; waits for some results, iterates this process several time and eventually ends the session. The finer the computational grain is, the minimum the transmission end-to-end delay will need to be kept. It is also desirable to minimize the overhead at the source since it may need to gather results and build data for the next computing step. More complex sessions put higher demands on the network resources and on the multicast/broadcast communication facilities (cooperation among the receivers, receivers acting as sources for the other receivers, . . .).

Meeting the objectives of reliable multicast is not an easy task. In the past, there have been a number of propositions for reliable multicast protocols that rely on complex exchanges of feedback messages (ACK or NACK) [12, 11, 23, 29]. These multicast protocols usually take the end-to-end solution to perform loss recoveries. Most of them fall into one of the following classes: sender-initiated, receiver-initiated and receiver-initiated with local recovery protocols. In sender-initiated protocols, the sender is responsible for both the loss detection and the recovery [12]. These protocols do not scale well to a large number of receivers due to the ACK implosion problem in the source. Receiver-initiated protocols move the loss detection responsibility to the receivers. They use NACKs instead of ACKs. However they still suffer from the NACK implosion problem when a large number of receivers have subscribed to the multicast session. In receiver-initiated protocols with local recovery, the retransmission of a lost packet can be performed by any receiver [11] in the neighborhood or by a designated receiver in a hierarchical structure [23]. All of the above schemes do not provide exact solutions to all the loss recovery problems. This is mainly due to the lack of topology information at the end hosts.

In this section on multicast protocols, we show the benefits a computing grid can draw from an underlying active reliable multicast (ARM) service by comparing the performances (mainly the achievable throughput) of several active mechanisms with the non-active case.

6.2 Active Reliable Multicast Explained

In active networking, routers themselves play an active role by executing application dependent functions on incoming packets. Recently, the use of active network concepts [25] where routers themselves could contribute to enhance the network services by customized functionalities have been proposed in the multicast research community [26, 19]. Active services for ARM contribute mainly on feedback implosion problems, retransmission scoping and cache of data. New

ARM protocols open new perspectives for achieving high throughput and low latency on wide-area networks. For instance, the cache of data packets allows for local recoveries of loss packets and reduces the recovery latency. Global or local suppression of NACKs reduces the NACK implosion problem and the subcast of repair packets only to a set of receivers limits both the retransmission scope and the bandwidth usage, thus improving scalability.

Designing an efficient ARM protocol is not an easy task and difficult design choices must be made. In order to demonstrate the benefit of ARM on a computing grid, we will compare 3 generic protocols noted S_1, S_2 and S_3. S_1 uses the global suppression of NACK packets within active routers whereas S_2 uses the NACK local suppression strategy (the receivers wait for a random amount of time prior to sending a NACK to the source). Finally, we have S_3, which is similar to S_1 in performing a global NACK suppression strategy, that also implements the subcast service within active routers in addition to the NACK suppression service. The next subsection presents some performance results using the previously described notations. At this point, we must mention that a full version of the results can be found in [21].

6.3 Performance of Active Reliable Multicast

In the following scenario, we will assume that the computing resources are distributed across an Internet-based network with a high-speed backbone network in the core (typically the one provided by the telecommunication companies) and several lower-speed (up to 1Gbits/s) access networks at the edge, with respect to the throughput range found in the backbone. Our test scenario involves an initiator (source) and a pool of computing resources (receivers) where communication from the source to the receivers are multicast communications. We will call *source link* the set of point-to-point links and traditional routers that connects the source to the core network. Similarly, a *tail link* is composed of point-to-point links and routers connecting a receiver to the core network. Active routers are associated to the tail links (the low- to medium-performance Internet links). However, it is possible that not all routers implement active services. Each active router A_i is responsible of B receivers $R_{i1}, \cdots, R_{iB}$ forming a local group. A receiver associated with an active router is said *linked.* The other receivers are said *free.* Figure 6 depicts the test scenario.

Figure 7 plots the ratio of linked receivers and active routers throughput as a function of the loss probability for S_2 and S_3. This figure illustrates the benefit of global NACK suppression when several local group sizes are defined. For reasonable loss probabilities, S_3 performs better than S_2 at the linked receivers end. This is because the linked receivers under S_3 benefits from the subcast service. In S_3, a linked receiver receives only once a data packet in contrast with S_2 where a linked receiver could receive more than one copy of the same data packet. Moreover, in S_2, a linked receiver can continue to receive NACKs from its active router every time a receiver in its local group has experienced a loss.

The subcast facility has the advantage of unloading the receivers and/or the active routers depending on whether we benefit from this facility from the

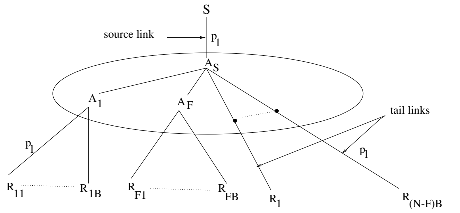

Fig. 6. A Simple Grid Session Model.

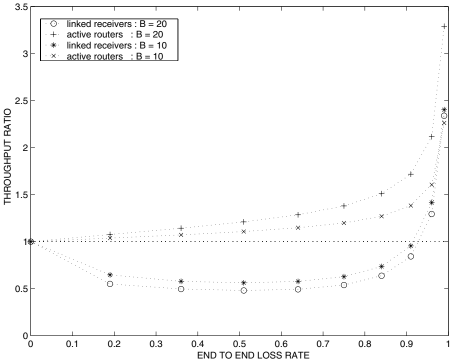

Fig. 7. Benefit of Glocal Suppression.

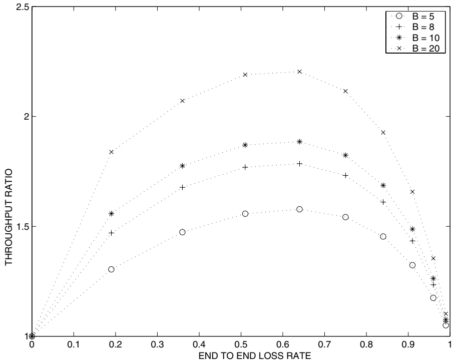

Fig. 8. Benefit of Router Subcasting.

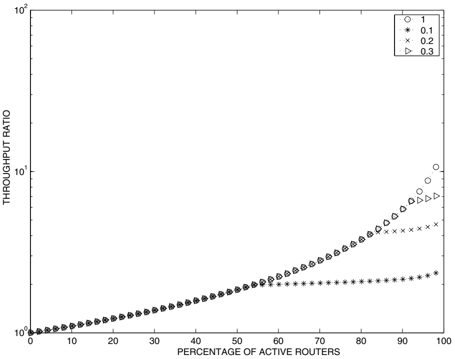

Fig. 9. Impact of Active Router Density.

source or not. To see the benefit of performing the subcast from the active routers associated to the linked receivers, figure 8 plots the throughput ratio at a linked receiver in S_3 and S_1. We can see that the subcast permits a higher throughput at the linked receivers in S_3. The gain obtained with the subcast depends on the local group size and the loss rate. These two parameters gives an idea on the number of receivers that have experienced a loss. Therefore, it is very beneficial to perform the subcast when the local group size is large (large scale distributed computing).

Figure 9 shows the impact of the active routers density on a protocol's performances in term of the overall throughput. The figure plots the overall throughput gain as the number of active routers is increased compared to the no active routers case. We have $N = 100$ and have 1000 end-receivers. The number of active routers is varied on the x-axis and the y-axis shows the throughput ratio when compared to a non-active solution. Several multiplicating factors to the active routers' processing power are applied (for instance 0.1 means 10 times slower). We can see that with the same processing time at the active routers and the receivers, the overall throughput can be an order of magnitude higher if all the receivers are linked. Most interestingly, if the active router's processing power is divided by 10 in S_3, we can still double the overall throughput provided that 55 % of routers are active. Most predictions assume that the active router processing power will certainly be 5 or 10 times greater in a near future. However, in case an active router is overloaded and exhibits less processing power than simple receivers, active services still provide more performances than the non-active case if the density of active routers is increased.

7 Conclusion and Future Works

We have studied the Grid computation models in order to determine the main Grid network requirements in terms of efficiency, portability and ease of deployment.

We then studied a solution to answer these problems: The active networking approach, where all network protocols can be deported in the network in a transparent way for the Grid designer and the Grid user. All communications protocols required by the Grid (multicast, dynamic topology management, QoS, data conversion, "on the fly" data compression,. . .) can be implemented as active services deployed on demand in active nodes.

We specially explored how active networking provides an elegant solution that can handle efficiently the QoS and multicast services required by Grid environment and Grid applications.

We proposed such active network support: The Tamanoir Framework and studied active QoS and reliable multicast services on top of it. The first results are promising and should lead major improvements in the behavior of the Grid when the A-Grid support will be deployed.

By proposing new intelligent services, active network can be the perfect companion to easily and efficiently deploy and maintain Grid environments and applications.

Next step will consist of merging the Tamanoir framework with a Globus Grid environment and we are currently adding active storage protocols by including in Tamanoir framework the distributed storing facilities provided by the Internet Backplane Protocol software (IBP [5]). This distributed storage facility in the network will help us to implement active reliable multicast service on top of Tamanoir environment. We have seen that with the active network technology, it is possible to efficiently transfer the QoS management and control functions inside the network. New active Grid QoS services will be proposed to allow active nodes to adjust Grid streams depending on QoS requirements.

References

[1] Entropia: High performance internet computing. http://www.entropia.com.

[2] The polder metacomputing initiative. http://www.science.uva.nl/projects/polder.

[3] Xtremweb: A global computing experimental platform. http://www.xtermweb.net.

[4] Audio-Video Transport Working Group, Henning Schulzrinne, Steve Casner, Ron Frederick, and Van Jacobson. RTP: A transport protocol for real-time applications. Internet Request For Comments RFC 1889, Internet Engineering Task Force, January 1996.

[5] M. Becj, T. Moore, J. Plank, and M. Swany. Logistical networking: Sharing more than the wires. In C. S. Raghavendra S. Hariri, C. A. Lee, editor, *Active Middleware Services, Ninth IEEE International Symposium on High Performance Distributed Computing*, pages 140–154, Pittsburgh, Pennsylvania, USA, August 2000. Kluwer Academic Publishers. ISBN 0-7923-7973-X.

[6] M. Beck, H. Casanova, J. Dongarra, T. Moore, J. Planck, F. Berman, and R. Wolski. Logistical quality of service in netsolve. *Computer Communication*, 22(11):1034–1044, July 1999.

[7] Steven Blake, David Black, Mark Carlson, Elwyn Davies, Zheng Wang, and Walter Weiss. An architecture for differentiated services. Internet Request For Comments RFC 2475, Internet Engineering Task Force, December 1998.

[8] Robert Braden, David Clark, and Scott Shenker. Integrated services in the internet architecture: an overview. Internet Request For Comments RFC 1633, Internet Engineering Task Force, June 1994.

[9] Robert Braden, Lixia Zhang, Steve Berson, Shai Herzog, and Sugih Jamin. Resource reservation protocol (RSVP) – version 1 functional specification. Internet Request For Comments RFC 2205, Internet Engineering Task Force, September 1997.

[10] Rajkumar Buyya, Jonathan Giddy, and David Abramson. An evaluation of economy-based reseource trading and scheduling on computational power grids for parameter sweep applications. In C.S. Raghavendra S. Hariri, C.A. Lee, editor, *Active Middleware Services, Ninth IEEE International Symposium on High Performance Distributed Computing*, Pittsburgh, Pennsylvania, USA, August 2000. Kluwer Academic Publishers. ISBN 0-7923-7973-X.

[11] S. Floyd, V. Jacobson, and Liu C. G. A reliable multicast framework for light weight session and application level framing. In *ACM SIGCOMM'95*, pages 342–356, 1995.

[12] XTP Forum. *Xpress Transport Protocol Specification*, March 1995.

[13] I. Foster and C. Kesselman. Globus: A metacomputing infrastructure toolkit. *Intl J. Supercomputing Applications*, 11(2):115–128, 1997.

[14] I. Foster, A. Roy, V. Sander, and L. Winkler. End-to-end quality of service for high-end applications. *IEEE Journal on Selected Areas in Communications - Special Issue on QoS in the Internet*, 1999.

[15] B. Gaidioz, R. Wolski, and B. Tourancheau. Synchronizing network probes to avoid measurement intrusiveness with the network weather service. In *9th IEEE High-performance Distributed Computing Conference*, pages 147–154, August 2000.

[16] GCJ. The gnu compiler for the java programming language. http://sourceware.cygnus.com/java/.

[17] Jean-Patrick Gelas and Laurent Lefèvre. Tamanoir: A high performance active network framework. In C. S. Raghavendra S. Hariri, C. A. Lee, editor, *Active Middleware Services, Ninth IEEE International Symposium on High Performance Distributed Computing*, pages 105–114, Pittsburgh, Pennsylvania, USA, August 2000. Kluwer Academic Publishers. ISBN 0-7923-7973-X.

[18] Andrew Grimshaw, Adam Ferrari, Fritz Knabe, and Marty Humphrey. Legion: An operating system for wide-area computing. *IEEE Computer*, 32(5):29–37, May 1999.

[19] S. K. Kasera, S. Bhatacharyya, M. Keaton, D. Kiwior, J. Kurose, D. Towsley, and S. Zabele. Scalable fair reliable multicast using active services. *IEEE Network Magazine's Special Issue on Multicast 2000*, 2000.

[20] Miron Livny. Managing your workforce on a computational grid. In Springer Lecture Notes in Computer Science, editor, *Euro PVM MPI 2000*, volume 1908, September 2000.

[21] M. Maimour and C. Pham. A throughput analysis of reliable multicast protocols in an active networking environment. In *Sixth IEEE Symposium on Computers and Communications (ISCC 2001)*, July 3-5th 2001.

[22] Kathleen Nichols, Steven Blake, Fred Baker, and David Black. Definition of the differentiated services field (DS field) in the IPv4 and IPv6 headers. Internet Request For Comments RFC 2474, Internet Engineering Task Force, December 1998.

[23] S. Paul and K.K. Sabnani. Reliable multicast transport protocol (rmtp). *IEEE Journal of Selected Areas in Communications, Special Issue on Network Support for Multipoint Communication*, 15(3):407–421, April 1997.

[24] A. Reinefeld, R. Baraglia, T. Decker, J. Gehring, D. Laforenza, J. Simon, T. Romke, and F. Ramme. The mol project: An open extensible metacomputer. In *Heterogenous computing workshop HCW'97, IPPS'97*, Geneva, April 1997.

[25] D.L. Tennehouse, J.M. Smith, W.D. Sincoskie, D.J. Wetherall, and G.J. Winden. A survey of active network research. *IEEE Communications Magazine*, pages 80–86, January 1997.

[26] L. Wei, H. Lehman, S.J. Garland, and D.L. Tennenhouse. Active reliable multicast. In *IEEE INFOCOM'98*, March 1998.

[27] David Wetherall, John Guttag, and David Tennenhouse. Ants : a toolkit for building and dynamically deploying network protocols. In *IEEE OPENARCH'98*, April 1998.

[28] R. Wolski. Forecasting network performance to support dynamic scheduling using the network weather service. In IEEE Press, editor, *6th IEEE Symp. on High Performance Distributed Computing, Portland, Oregon*, 1997.

[29] R. Yavatkar, J. Griffioen, and M. Sudan. A reliable dissemination protocol for interactive collaborative applications. In *ACM Multimedia'95*, November 1995.