

RELIABLE MULTICAST WITH ACTIVE FILTERING FOR DISTRIBUTED SIMULATIONS

Gowri Rajappan

Michel Dalal

Nevelex Corporation

Sunnyvale, CA

e-mail: {gowri, dalal}@nevelex.com

and

Thomas Kostas

Kostas Consulting

Reading, MA

e-mail: tom@kostas.org

ABSTRACT

Large distributed simulations where multicast is used for data distribution have an inherent scalability problem—the number of multicast groups grows rapidly with the number of participants. We describe a publish/subscribe based active networking system where receiver interest information is stored as packet filters at the active nodes on the multicast distribution tree and multicast data is pruned at the earliest node possible. Modifications to the system provide reliable data delivery to support loss-sensitive data streams. Selective caching at some active nodes allows the recipient of a lossy data stream to partially trace back to the first cached instance of the missing packet. We argue that the performance penalty for applications that do not use the active filtering service is minimal, enabling the system to coexist in an open network with other applications.

INTRODUCTION

Active networks have been studied as a solution to problems with traditional network infrastructure—difficulty of introducing new technologies and standards, inflexible infrastructure that makes incorporation of new services difficult, and redundancy of operations at the various layers of the network stack [1]. In active networks end users or applications have the ability to influence and modify the network infrastructure dynamically. This is achieved by enabling the intermediate network nodes to process special code included in data packets. An end node or application has the ability to insert code into the packets they source.

A survey of active networking research is provided in [1]. An active networking toolkit known as ANTS is described in [2]. An active network node based on spe-

cialized hardware is presented in [3]. Many emerging services such as video on demand and grid computing involve groups of network entities communicating and collaborating. Hence we are interested in group communications [4] over active networks. The group communication applications place strenuous QoS demands on the network infrastructure. Hence there is need for an efficient and reliable multicast solution for active networks. In [5], the authors present a reliable multicast framework based on the ANTS toolkit. This work involves a general and flexible framework for reliable multicast support, and was done with the intent of studying reliable multicast service provision for active networks. We are interested in extending a similar service to a high performance active network infrastructure designed for distributed simulations and other such large-scale group communication applications.

Grid computing [6] and large-scale distributed simulations require a flexible infrastructure that is *scalable*, *timely*, and *context sensitive*. We present such an infrastructure based on interest based active filtering at the network nodes. We call this network protocol as Dynamic Active Filtering with Reliable Multicast (DAFRem). This work is a follow-up of prior work by the authors on research and implementation of Specialized Active Networking for Distributed Simulation (SANDS) [7, 8]. SANDS is a high performance active networking infrastructure designed to enable large scale simulations.

DISTRIBUTED ACTIVE FILTERING INFRASTRUCTURE

SANDS is an active networking infrastructure in which the nodes can be selectively activated by installation of appropriate code, and performance gains scale as the number of nodes thus activated [7, 8]. SANDS aims to

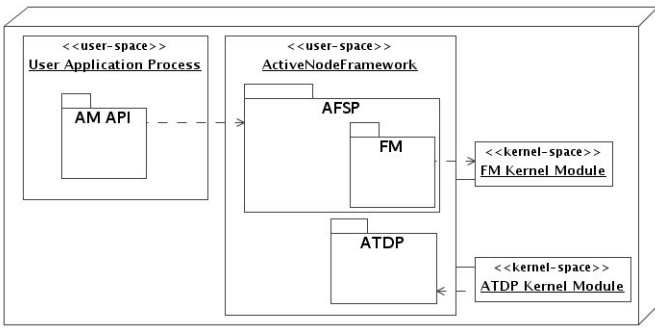


Figure 1: Dynamic Filtering Infrastructure Active Node.

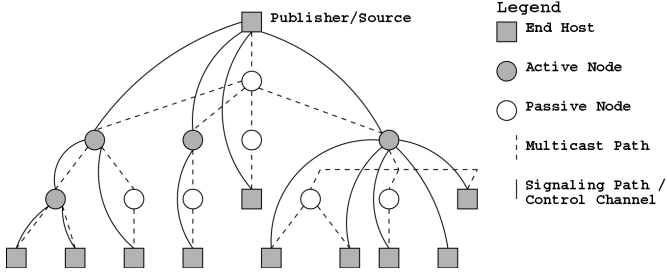


Figure 2: A multicast tree showing the signaling overlay network between the active nodes.

enable massively distributed (possibly real-time) simulations with millions of multicast groups. Such applications are beyond the capability of conventional networking protocols. The SANDS design is based on a publish/subscribe strategy. Data sources periodically publish a description of the available information, and data receivers have the capability to subscribe to known data groups. A combination of these two mechanisms is used to dynamically affect node participation. The data traffic is shaped by pruning unwanted data packets at the earliest node possible along the multicast distribution trees by use of a dynamic filtering mechanism.

Figure 1 is a logical block diagram of an active node in SANDS, as implemented by the authors. The core of SANDS is the Active Topology Discovery Protocol (ATDP) which dynamically establishes reliable signaling paths between active nodes called control channels. The channels are established between adjacent active nodes in a multicast tree creating an active overlay network, as shown in Figure 2. The control paths established between active nodes are discovered using *Source Path Messages* (SPMs). SPM is an application-specific multicast data packet with a simulated router-alert option set. SPMs are periodically sent by the multicast publisher node of a multicast session to initially establish and then main-

tain the overlay network. Each SPM packet contains information about the last active node that processed it. An active node in the multicast tree processes SPMs by (1) obtaining connectivity information for the upstream active node (parent node) as those packets arrive on the network interface and (2) modifying the packet to contain the current node's connectivity information as it leaves on a network interface.

The overlay network provides the reliable communication path for sending control messages between the active nodes. The active network core allows for the dynamic installation of additional protocols using a predefined API. In addition, these protocol implementations may communicate across active nodes by encapsulating their specific signaling messages within ATDP protocol messages. With multicast-based Active Interest Filtering (AIF) enabled, via an implementation of the Active Filter Signaling Protocol (AFSP), a node becomes a member of the active filtering system. The AFSP in our implementation is based on the AFSP version 2 (AFSPv2) in the original SANDS specification. AFSP requires a Filter Manager (FM) subsystem to manage filter life-cycles and handle packet filtering. Multicast data packets published by a source include a content header that the FM uses for filtering. Each active node includes an Application Manager (AM) API that allows an end-user application to:

1. Register filters into the AFSP subsystem.
2. Publish data with content headers.
3. Subscribe to published data.

Essentially, the AM API standardizes the process of content header insertion into data packets and content header removal from data packets. Data packets published via the AM API do not include any executable code. They are marked with a special header that describes the data content as a point in an n-dimensional coordinate space. An FM Loadable Kernel Module (LKM) is used to process these content descriptors. The LKM drops packets before they leave a network interface if the coordinate point describing the data does not fall within any of the filters applicable to that interface.

RELIABLE MULTICAST SERVICE

The baseline specification for a reliable multicast service is derived from [5]. Three types of services are identified: *Unreliable*, *Time-constrained Reliable*, and *Reliable*. Unreliable data service does not profess any delivery guaran-

tees. Time-constrained Reliable service allows the sender to specify a time-to-expire for the data packet, and reliable data transport procedures are invoked within this time frame. This service is suitable for real-time data delivery applications with a brick-wall utility curve (i.e., the data is useful only when it is delivered within a specified time frame, after which its utility plummets to zero.) In a Reliable service, the protocol ensures delivery to all receivers, and notifies the sender of any irrecoverable failures. This service is suitable for error-intolerant, but somewhat latency-tolerant applications. The achievable error and latency values depend on the physical network characteristics.

Additional features specified at the active nodes are—

1. Data caching. This allows data packet retrieval by partial trace back.
2. Peer inquiry for packet retrieval. An active node can inquire peer active nodes to recover a missing packet, instead of tracing back to the source.
3. ACK fusion. Acknowledgements from the child receivers are combined at the active nodes.
4. NACK processing. When a missing packet is detected at a receiver, a NACK is sent to the parent active node, which initiates and manages a packet retrieval procedure.

An unreliable service allows any receiver to join or leave a session, and the data source need not be notified of identities of the receivers that join or leave a session. But for Time-constrained Reliable and Reliable services the data source is informed of the identity of the receivers. Optionally, the data source can authenticate the receivers that join the session.

Flow control helps the source adapt to the end-to-end bandwidth of the participating receivers. It provides the capability to identify and appropriately deal with (e.g., terminate the connection) slower receivers that impair the application. Explicit flow control is implemented at the source. In addition, the distributed filtering infrastructure provides implicit flow control and complexity-bandwidth tradeoff, as follows.

Filter economy is traded off for pipeline width. By aggregating filters at an active node more packets than necessary may be passed (wide pipeline.) This is because filter aggregation, which is done to increase filter economy, usually has a larger coverage than the combined effect of

the filters applied in sequence. On the other hand, maximum packet pruning is achieved by applying the original filters in sequence, and outgoing pipeline width is minimum (but at the cost of more filtering.) A tradeoff achieves a compromise between the two extremes. For example, if we assume that data packet contents are described as real numbers, the filters are then ranges of real numbers. If Filter A is $[0, 2]$ and Filter B $[4, 5.5]$. Then these filters can either be applied in sequence, or they can be aggregated to yield Filter C as $[0, 5.5]$. This reduces the number of filters to be applied, but it allows extra packets through (leakage.)

By appropriate coordination some active nodes can prune more and others less depending on the traffic/congestion conditions. This is a multi-node, distributed version of the previous single-node tradeoff. For instance a parent node that is connected through high speed links to its child active nodes can negotiate to take over caching responsibilities of the child active nodes, and in turn be allowed to aggregate its filters. It alleviates caching requirements of the child nodes. On the other hand, if child nodes are participating in many of the same multicast sessions, the parent node can perform sequential filtering and the child nodes can employ simple all-pass filters. Specific optimizations are performed by exchange of *processing optimization capsules*. Processing optimization capsules are used to achieve fine-grained tradeoff between filtering complexity and bandwidth efficiency.

The setup procedure for DAFReM is as follows. Once topology discovery by ATDP is completed, the data sources publish their content description information by sending *publication capsules* to the active nodes it is connected to, and this information propagates over the network to be held at the active nodes.

The next phase is subscription. The receivers that are interested in receiving specific data send *subscription capsules* to its parent active node. The active nodes use an intersection of the publish and subscribe information to set up filters. These filters are then forwarded upstream and they eventually reach the head of the multicast distribution tree. When the filter is set up at the node following a source, data from that source is allowed through. This is known as *Diffusion Routing* (the default filter is a null filter that does not allow any data through.)

The default service as described above constitutes the Unreliable service. In order to participate in either the Time-Constrained Reliable or Reliable service the receivers are required to publish their identity and capability information. When a receiver wishes to partici-

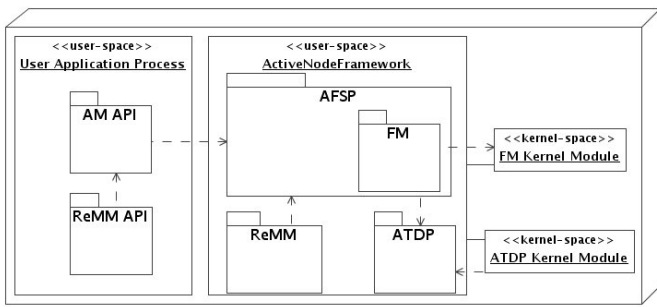


Figure 3: DAFReM active node showing the portable user-space components and the nonportable kernel-space components.

pate in a reliable service, it sends a *reliable subscription capsule* upstream. The reliable subscription capsule has identification (possibly authentication), capability, and subscription information. Its parent active node then appends this information to a *Cache Table*. The cache table consists of receiver IDs whose data needs to be cached until an ACK is received.

The parent active node also makes an entry in a *Data Retrieval Table*, so that in case of NACKs the active node can activate the data retrieval procedure. The data retrieval table has information about which of the peer nodes, if any, cache data of interest to the receiver. Peer information is obtained by a peer enquiry mechanism provided by AFSP. Peer retrieval allows lateral data recovery instead of having to partial trace back to the source, when the requested retransmission data is not found locally.

The active node that is the nearest upstream parent of a receiver node is responsible for aggregating ACKs from that and other child receiver nodes and sending it back towards the source. Whenever a missing data packet is detected, the parent active node manages Negative Acknowledgments (NACKs) and retransmissions.

For mission critical data streams, the source can authorize the receivers that participate in the session. When the receiver specifies the subscribe information and it trickles upstream to the source, the source can either accept or deny the request based on the receiver capabilities, for instance. If the source denies the request then a downstream pass strips the filters that were waiting to be setup at the intermediate active nodes. This deny notification works its way back to the receiver.

As shown in Figure 3, the reliable service is added as a module to the active node. The Reliable Multicast Mod-

ule (ReMM) uses the signaling facilities provided by the ATDP and AFSP to set up, manage, and tear down reliable multicast sessions. ReMM provides UDP-based reliable multipath communication and has high messaging efficiency. ReMM API provides the interfaces required for joining, setting up, and tearing down of reliable multicast groups.

PERFORMANCE CONSIDERATIONS

First we argue that the original unreliable service provided by the SANDS dynamic active filtering infrastructure is not adversely affected by addition of a reliable service. Secondly we argue that the dynamic filtering infrastructure itself does not disrupt co-existing services in an open network. Finally we postulate that the presented active filtering infrastructure does not suffer from the security concerns associated with most active networks.

From the perspective of a receiver wanting to receive an unreliable service, addition of the reliable service is transparent—the subscribe mechanism for an unreliable service is still the same as before. But there is an impact on the serviceable bandwidth available to the receiver due to messaging and retransmissions attributable to the reliable service. The reliable service comes with an admission control policy since the receivers have to reveal their information and the source has an opportunity to decline service. Hence a certain measure of fairness can be established by inclusion of consideration towards unreliable services in the admission policy.

The packets that are not tagged as active filterable are not processed by the active nodes, and hence other services in an open network do not incur a direct processing penalty. But they have an indirect penalty since processing of active filterable packets takes up cycles. However, it can be argued that by appropriate tweaking using processing optimization capsules, taxing of resources in the open portion of a heterogeneous network can be avoided. A large portion of the applications DAFReM is designed for are assumed to be deployed in specialized networks. Open network is the non-specialized parts of the infrastructure (which cannot be avoided in group communication scenarios over the Internet, for instance) where other applications co-exist.

The active filtering mechanism does not require the active nodes to execute an embedded code locally, which is the primary security concern with most active networks. The embedded header information in data packets is used only for co-ordinate marking in order to achieve an accept or reject decision. This narrow implication of the

embedded information essentially makes this infrastructure an extension of conventional address-based routing scheme, in a manner of speaking, and hence eliminates additional security concerns.

DAFReM is efficient in a network of persistent flows, i.e., when the filters are constant for relatively long periods of time. When the flows are shorter, additional network traffic and filter processing overhead are incurred due to dynamic filter alteration. This is the case, for instance, when the consumer node is a data collection application or a human user at an interactive terminal. The consumption profile in this case may consist of sampling various subsets of the produced data for relatively short time durations. In this case, one may decide that network bandwidth cost is less expensive than filter alteration cost. If so, the nodes farther away from the fickle consumer node abstain from filtering traffic in the consumer's direction, and all pruning is done only close to the consumer. Conversely, in the case of persistent flows, bulk of pruning can be done close to the producer node thus maximizing throughput. Such considerations of Congestion and Flow control in DAFReM are important issues and they will be elaborated in a follow-up paper.

To summarize, DAFReM is especially suitable for applications consisting of distributed producers and consumers that need to be highly connected using fine-grained data pipes. Applications such as grid computing and large-scale distributed simulations fit this profile. For instance, a distributed simulation of a battlefield contains many geographically and contextually co-dependent entities. An actor (e.g., a soldier) in this theater is interested in all influencing activities that lie within its range of interest. Such continuous sifting of content- and context-tagged events is efficiently accomplished using distributed filtering.

CONCLUSIONS

We described a Dynamic Active Filtering infrastructure with Reliable Multicast (DAFReM.) This system was designed for large-scale group communication applications such as distributed simulations and grid computing. We presented the architecture, discussed the implementation, and treated performance issues. Quantitative performance measurement is yet to be done, but we argue that the DAFReM infrastructure does not disrupt co-existing network services. The efficient, flexible and distributed nature of the infrastructure implies an ability to achieve the three initial goals of scalability,

timeliness, and context-sensitivity. We intend to present performance results of the implementation in a future paper. Future work includes studying performance of specific applications such as distributed simulations and grid computing in the DAFReM infrastructure.

REFERENCES

- (1) David L. Tennenhouse, Jonathan M. Smith, W. David Sincoskie, David J. Wetherall, and Gary J. Minden. A survey of active network research. *IEEE Communications Magazine*, pages 80–86, January 1997.
- (2) David J. Wetherall, John V. Guttag, and David L. Tennenhouse. Ants: A toolkit for building and dynamically deploying network protocols. In *Proc IEEE OPENARCH*, pages 117–129, April 1998.
- (3) Dan S. Decasper, Bernhard Plattner, Guru Parulkar, Sumi Choi, John D. DeHart, and Tilman Wolf. A scalable high-performance active network node. *IEEE Network*, Jan/Feb 1999.
- (4) Christophe Diot, Walid Dabbous, and Jon Crowcroft. Multipoint communication: A survey of protocols, functions, and mechanisms. *IEEE Journal on Selected Areas in Communications*, 15(3):277–290, April 1997.
- (5) María Calderón, Marifeli Sedano, Arturo Azcorra, and Cristian Alonso. Active network support for multicast applications. *IEEE Network*, pages 46–52, May/June 1998.
- (6) Ian Foster, Carl Kesselman, and Steven Tuecke. The anatomy of the grid: Enabling scalable virtual organizations. *Intl Journal of High Performance Computing Applications*, 15(3):200–222, 2001.
- (7) Matthew D. Dorsch, Thomas Kostas, and Victor Skowronski. Reducing bandwidth requirements of distributed simulations. In *Proc Interservice/Industry Training, Simulation, and Education Conference*, 2002.
- (8) S. Zabele, M. Dorsch, Z. Ge, P. Ji, M. Keaton, J. Kurose, J. Shapiro, and D. Towsley. Sands: Specialized active networking for distributed simulation. In *Proc IEEE DARPA Active Networks Conference and Exposition*, May 2002.