The Intermon Simulation Framework

Maurizio Bartoli[1], Florian Baumgartner[2], Christof Brandauer[3], Torsten Braun[2],
Sándor Kardos[4], Fabrizio Orlandi[1], Matthias Scheidegger[2], Jörn Seger[5]

[1]*Telecom Italia, Via G. Reiss Romoli 274, 101048 Turin, Italy*
[2]*IAM, University of Bern, Neubrückstrasse 10, 3012 Bern, Switzerland*
[3]*Salzburg Research, Jakob Haringer Str. 5/III – A-5020 Salzburg, Austria*
[4]*TMIT, Budapest University of Technology and Economics, Hungary*
[5]*Elekt. Systeme und Vermittlungstechnik, University of Dortmund, Germany*

Abstract

The Intermon project aims to enhance inter-domain traffic analysis in large scale Internet infrastructures. An important part of the system is a set of simulators especially designed for inter-domain scenarios based on measurements from the real network. Simulation results are analyzed using the built-in visual data mining system. This paper presents the architecture enabling the interaction user, measurement tools, simulators and visualization components in a widely distributed environment. It also focuses on the issues that arise when building a system that is distributed over most of Europe.

1 Introduction

The aim of the Intermon project is to enhance inter-domain QoS and traffic analysis in large-scale, multi-domain Internet infrastructures. To this purpose the Intermon project defines, designs and implements a scalable architecture to provide both telecom operators and corporate users with a set of services that—in an automated fashion—collect, process and present information about network status (i.e. network topology, traffic and Quality of Service) and SLA fulfillment in both intra-domain and inter-domain scenarios. By using visual data mining techniques Intermon users can easily and efficiently access the information they need. The core of the Intermon architecture is a large, distributed information base which collects data provided by both measurement tools and analytical models.

A crucial part of Intermon is a set of network simulators that can be used to simulate the impact of certain changes to the network monitored by Intermon. Simulation scenarios are based on measurement data provided by other tools in the Intermon architecture and automatically converted to simulator-specific scenarios. The reason for choosing a set of simulators instead of implementing only a single one is that each simulation approach has its strengths in different aspects of network simulation. Integrating multiple simulators enables the user to choose the simulator best suited for his specific needs.

The common basis for all these modeling and simulation approaches is the inter-domain point of view. Traditional simulation approaches try to model the network as exactly as possible, making simulation scenarios of inter-domain networks virtually impossible due to scalability problems. Choosing a suitable abstraction reduces these scalability problems and enables the simulation of large scale inter-domain scenarios, at the cost of reduced accuracy. To provide high usability, all simulators use a common graphical user interface. Topology and measurement data collected by other tools and components of the Intermon system can be extracted from the database and presented using 3D visualization techniques. Depending on the choice of simulator the user is then able to interactively change a set of parameters of the simulation scenario.

As Intermon is a distributed system, components like simulators may be located on several independent computers, which requires well-defined protocols and an appropriate infrastructure. The infrastructure is provided by a Virtual Private Network, which connects almost all Intermon partner locations. The VPN simplifies communication issues within the distributed system, and can also provide a high degree of security by using encrypted tunnels between the Intermon sites.

In a typical use case the user sends simulation requests through the graphical user interface and over a central entity (the global controller) to the simulator. After the simulation task has finished the results are returned and can then be examined using the highly configurable visual data mining (VDM) toolkit. Since the messages are persistent the results can always be referred to later.

The remainder of this paper is organized as follows: Section 2 gives an overview of the Intermon architecture and especially the simulator integration, while Section 3 concentrates on infrastructural issues. Sections 4 and 5 go into more detail about the simulator infrastructure and the visual data mining component, respectively. Finally, Section 6 summarizes the paper.

2 Architecture

2.1 Overview

The Intermon toolkit architecture integrates a set of tools for structure discovery of inter-domain topology, as well as measurement, modeling, simulation and visual data mining of inter-domain traffic (Fig. 1). It has been developed using the Java programming language, and implemented as a distributed framework composed of the following main architectural components:

A Global Controller (GC), which is unique within a given Internet Autonomous System (AS); it accepts user requests and implements the Intermon services in collaboration with the other Intermon architecture components. The user-GC interaction aspect of the global controller is implemented using message driven beans, a J2EE enterprise bean type that acts on the receipt of messages, filtered by message properties.

Tool Managers (TMs), responsible for translating a task description coming from the GC into a tool specific input, and for adapting the tool output to the Intermon database structure.

Tool Wrappers (not shown in Fig. 1) to add support for remote control and data collection for an existing tool according to a common tool usage protocol.

The distributed nature of the Intermon architecture, coupled with the long-term nature of many inter-domain monitoring requests, calls for a flexible, scalable, and asynchronous communications structure: this is achieved through the use of the Java Messaging Service (JMS) API, which is a specification published by Sun Microsystems defining a set of programming interfaces by which Java programs can access message-oriented middleware. JBossMQ was chosen as the JMS system for the Intermon implementation. The messaging model used is of the publish/subscribe type, where the applications or components address messages to a topic object, which is responsible for distributing messages received from a publisher to all active subscribers.

Because of the great number of tools, modules and interfaces in the project an important focus was the definition
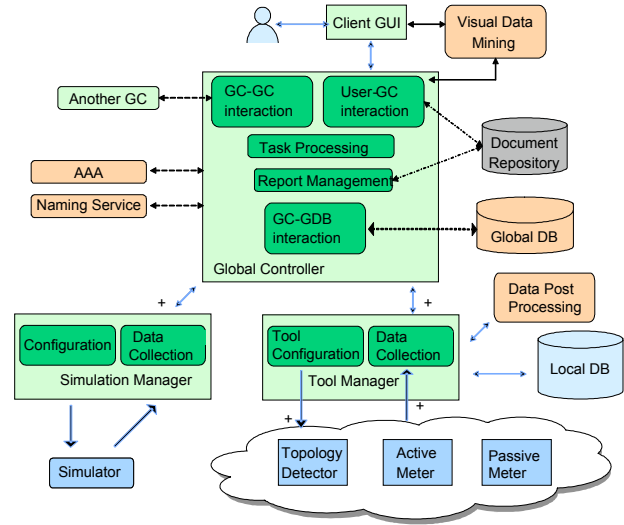
Figure 1. A global view of the Intermon architecture

of data formats. In the Intermon project the use of XML as a basis for data formats is mandated.

The architectural components described above rely on a large, distributed information base, which collects data provided by both measurement tools and analytical models. The database access uses Java Data Base Connectivity (JDBC) methods to store results and query the MySQL database.

From the user's perspective, the available tasks, commands and configurations for all the tools in Intermon are accessible through a menu-based Graphical User Interface (GUI).

2.2 Modeling and Simulation Toolkit

As a part of the Intermon framework the modeling and simulation toolkit makes use of the main architecture components GUI and Global Controller (GC). To integrate the simulation software into the the system, simulation-specific tool managers have been designed. Fig. 2 shows the modules of the Intermon architecture that are involved in the task of modeling and simulation.

A customized GUI interface helps the user to build simulation scenarios and to select, configure, start and control the appropriate simulator to evaluate them. Once the simulation request has been submitted the communication process invokes the GC functions that handle the forwarding of simulation requests to the targeted simulation manager. These functions may check and/or modify the request messages and also access the database to fetch additional information for the specific simulator. The message
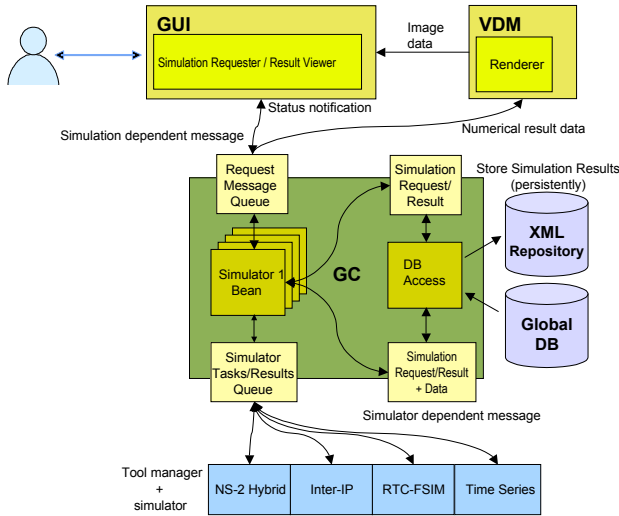
Figure 2. Intermon modules involved in modeling and simulation

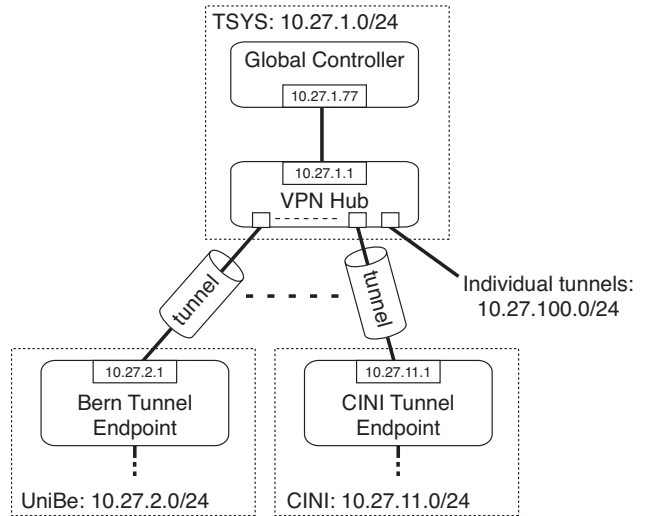**Figure 3. Setup of the virtual network**

is then passed to the manager and wrapper modules that are in charge of triggering the simulation.

When a simulation has finished the manager sends a message via the GC to the GUI with information about the location of the results (either transferred to the DB or stored in a remote repository) and possibly a summary of the simulation outputs.

The user can then request further analysis (e.g. statistical functions) to be applied to the results. This often involves the preparation of images from the processed data sets, which is the task of the Visual Data Mining (VDM) toolkit. In that case a visualization request is issued to the GC, referring to the results of the specific simulation task and passed to the VDM component.

3 Infrastructure

The Intermon architecture has been designed to allow for a highly distributed system. Measurements, data storage, simulation and visualization can all be hosted on individual systems and then communicate by means of a global controller (GC). On the inter-organizational level even GCs may communicate and exchange data. This ability to distribute tasks over the network to servers at several different organizations also implies that a significant part of communication between the systems must go through the Internet, which may cause security problems. Measurement data of internal networks are considered sensitive data by ISPs and should not be sent in clear text over public networks. Furthermore, most organizations have firewall routers installed to protect their intranets, making communication using the

Java Messaging System that is required by the global controller difficult. This is due to the very dynamic behavior of JMS, which makes configuring the firewall a complicated endeavor.

We have designed a connectivity solution based on a virtual network to finally solve this problem. This solution uses IPIP tunnels (IP protocol 4, also know as IPencap) to encapsule packets exchanged between the organizations' local networks. The advantage of IPIP is the available support for it. It is supported on Windows 2000 Server, Linux, the various BSDs, MacOS X, and others. By performing the tunneling on the network layer firewall configurations are simplified to a minimum.

Without such a setup we would have to insert a lot of rules into the firewall configurations in order to allow the GC messages to pass. Furthermore, any additional services would require further changes to the configuration. With IPIP tunnels these rules can be reduced to a single one that allows protocol 4 packets to pass. Thus, protocol types and ports are not an issue anymore. Although there is no encryption the step to a full VPN is small one since the IPIP tunnels are easily replaceable by IPSEC tunnels or application layer VPN software like OpenVPN, for example. Note that the communication between GCs will require a AAA architecture, which is outside the scope of this paper.

The virtual network has a star topology to concentrate complexity at a single node and thus to make firewall and end system configuration in the single organizations simple. Figure 3 shows this network setup with two example remote sites (University of Bern and CINI). We used the (B-sized) network 10.27.0.0/16 as address space, assigned each site a C-sized subnetwork. Each subnet is connected to T-Systems Berlin (where the GC is located) via a tunnel

with the endpoints $10.27.x.1$ in the subnet and $10.27.1.x$ on jt-ext (for Bern $x = 2$).

To enable communication with the virtual network, each partner organization needs to configure a computer in their LAN to be a tunnel endpoint towards Berlin and to be a secondary gateway towards the other computers in the LAN. On UNIX alike systems this is usually done by invoking `iptunnel` to configure an IPIP tunnel. Alternatively, OpenVPN can be used to set up an encrypted tunnel over UDP using a single ports on both endpoints.[1] Then, any other computers in the LAN need secondary addresses— preferably $10.27.x.10$, $10.27.x.11$ and so forth—and two additional routing table entries: One to the secondary local network $10.27.x.10/24$ (usually automatically configured by the operating system), and one to the tunnel endpoint for any other addresses in the Intermon VPN (i.e. in $10.27.0.0/16$). The configuration can also be done without using secondary addresses and gateways, but doing so breaks any other network connectivity of these computers. Experience shows that problems with secondary addresses only arise on Windows systems where DHCP is used to configure the primary network interface. In such cases, adding an additional address and gateway appears to be impossible.

A small fraction of the Intermon servers and clients could not be connected to the VPN, for technical or political reasons. If the services in question can be restricted to a small number of ports, network address translation (NAT) on the central hub can be used to solve the problem.

4 The Simulation Framework

The Intermon architecture includes four modeling and simulation approaches, each specialized on estimating and predicting certain aspects of network behavior. This enables the user to perform network planning tasks and what-if analysis, in addition to examining the status quo of the monitored network. This section gives an overview of the available modeling and simulation approaches and presents their integration into the architecture.

4.1 Available Simulators

4.1.1 Hybrid NS-2

The hybrid simulation module [1] combines packet-based simulation of ns-2 with analytical models by using a hot-plug mechanism, which extends ns-2 nodes using modules that behave like whole networks (e.g. autonomous systems). This abstraction allows the user to simulate large scale topologies in a fraction of the time a full scale packet-based simulation would take.

The analytical multi-domain model is based on the assumption that, at a given time, the Internet can be divided into areas where congestion is negligible, interconnected by bottleneck links. Congestion free areas are treated as black boxes and called domain models. Packet loss and delay caused by excessive queues, occurring on the bottleneck links, are modeled by inter-domain link models. Creating models for congestion free areas has the advantage that the simulation of packet losses and excessive queuing can be restricted to a small part of a simulation scenario (the inter-domain link models), thus greatly reducing the complexity of the domain models. In fact it is sufficient to model domains using quasi-static delay distributions. Apart from its scalability advantage this approach may be useful to model network areas of which we do not know the exact topology.

Suggested application areas for the hybrid simulator include end-to-end QoS evaluation of single flows— simulated using traditional packet-based models—over a complex backbone network, or the effect of changes in a backbone network (e.g. addition/removal of links, capacity changes, big changes of network load due to new SLAs, etc.) on flows traversing the domain.

4.1.2 RTC-FSIM

The Rate and Time Continuous Fluid Simulation (RTC-FSIM) is a novel modeling approach developed within Intermon [2, 3]. It is aimed at inter-domain scenarios that are characterized by large topologies and a high degree of traffic aggregation. In order to avoid the scalability problems of packet-based simulations, the RTC-FSIM model uses a fluid flow abstraction where traffic is modeled as chunks of fluid flows. In classical fluid flow simulators the individual packet arrivals are aggregated to flow arrival/departure events that trigger the event based simulator. However, as has been shown in [4] the so called "ripple-effect" can lead to an explosion of simulation events such that the event rate is even higher than in equivalent packet based simulations.

In the RTC-FSIM model, a transition from a discrete load process to a continuous fluid process is made. To keep the most important process characteristics—mean, variance, and autocorrelation—the continuous fluid process is derived from the discrete fluid process by a newly developed iterative algorithm. With this abstraction inter-domain links become continuous queuing systems and the dynamic relations between incoming fluid-traffic, service link rate, buffer occupation, loss rate, etc. are described by differential equations. RTC-FSIM can be tightly integrated with the evolving IPFIX standard to implement measurement based simulations.

The performance of the RTC-FSIM simulator, which is currently implemented in a Matlab/Simulink environment,

---

[1] Recent versions of OpenVPN even support tunneling over TCP.

does not suffer from the ripple-effect but mostly depends on the performance of the differential equation solver.

RTC-FSIM is very well suited for measurement based simulations. It is particularly useful in very large scale inter-domain scenarios as the simulation speed is independent of the link speed and the amount of aggregated traffic. Additionally, accuracy can be traded against performance by adjusting the sampling interval for the differential equation solver. On the basis of the measured traffic the delay performance can be evaluated for scenarios like the removal of links, rerouting, change of capacity, additional load.

4.1.3 Inter-IP

The INTER-IP tool evaluates the delay performance for a traffic relation (i.e. a flow identified by the source and destination IP address and the service class on which it is mapped) that crosses multiple domains. Two different types of service class are supported: one strict priority class and up to five classes serviced in a round-robin way (this model is similar to the MDRR scheduling discipline implemented on Cisco giga-routers [5]). For the traffic mapped onto the strict priority class the tool evaluates the mean packet delay, whereas for the traffic mapped onto the classes serviced in the round-robin way the mean volume transfer delay is computed. Two different parameters are used because each one is more effective than the other for describing the user perception of two different types of application. For delay-sensitive applications like Voice-Over-IP or interactive applications the packet delay is the relevant performance parameter. On the other hand, for applications based on large volume transfer, usually using TCP, the relevant parameter is the throughput obtained, or the volume transfer delay.

The delay calculation is totally analytical and based on a proprietary queuing model for the links. The formula resolving this model is in closed form. This allows for near real-time performance evaluation and suggests using this tool when rapid comparison of different alternatives (e.g. routing alternatives for a flow) has to be performed. Another possible application is the preliminary study of a scenario that we want to simulate with the much more time and resource consuming simulation tools, in which we skim off the most interesting simulation cases that would not be easily identifiable "a priori."

4.1.4 TSSIM

The Time Series Simulator (TSSIM) is an efficient inter-domain simulator, which works on the time series of aggregated traffic data, and evaluates the QoS state of the network in terms of throughput, drops, delay and jitter [6]. Since the simulator is based on aggregate load information,

the simulated QoS values represent an average for the traffic as a whole. The simulated network consists of network domains. Each domain is represented by its border routers, which process the time series that represents the input traffic and produce output time series that are forwarded to the successive border routers in the same or neighboring domains. On the domain level the simulator assumes that the input traffic from and the output traffic to a given domain have no effect on each other. A border router uses an input model and an output model, which can be one of the following:

NullModel: This model simply copies the input to the output introducing only a static delay and distributing the output among the neighbors. This model is primarily used as input model since the inside of a domain is supposed to be ideal except a static mean delay.

LeakyBucket: This models a simple FIFO queue with limited server capacity and queue length. It handles aggregate load information and produces throughput, drop, delay and jitter state information (the latter two at low accuracy).

Class4LeakyBucket: This models the same queue as the LeakyBucket model but handles aggregate and packet level load information, too. It produces the same QoS state as the LeakyBucket model but at a higher accuracy.

Utilizing these models, TSSIM can be used to analyze the effects of adding or rerouting traffic in the network.

4.2 Creation of Simulation Jobs

The Intermon toolkit provides a GUI for the specification of a simulation request. The GUI consists of four tabs whereby the content of one tab depends on the type of simulation tool chosen.

In the first tab named "Common parameters" (see Fig. 4(a)) the user selects the simulation tool to be used among the four alternatives integrated into Intermon. Parameters like a name, description, comment, and starting time for the simulation can also be entered.

The second tab named "Specific parameters" (see Fig. 4(b)) gives the user the opportunity to set the configurable parameters of the simulation tool selected in the previous step. The content of this tab is dynamically adapted according to the type of simulator chosen. As an example, the RTC-FSIM simulators allows the choice of a sampling interval for the solver of the differential equations.

In the third tab named "Topology Chooser" (see Fig. 5(a)) the user selects the base topology of the simulation from a list of topologies that have been created before. The topologies are typically produced by Intermon's
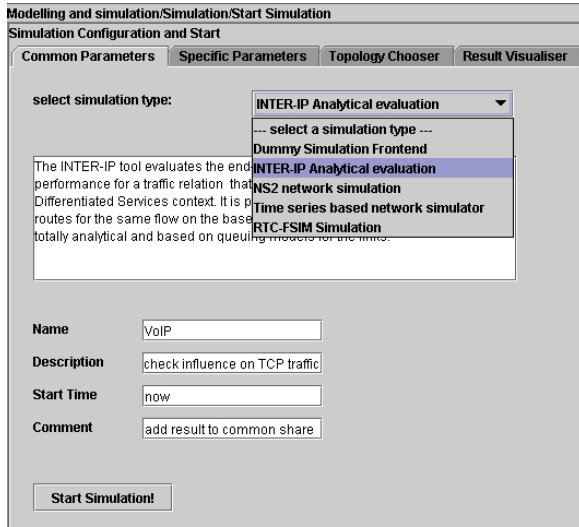
(a) "Common Parameters" tab
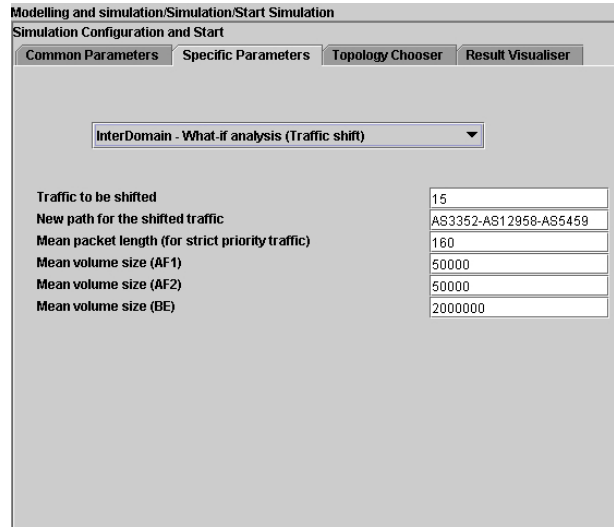
(b) "Specific Parameters" tab

Figure 4. Screenshots of the simulator GUI

InterRoute tool, which analyses BGP messages to detect the topology around itself. When a list entry is selected a 3D visualization of the topology is shown immediately. Depending on the simulation tool selected in the first tab, the user can now apply certain changes. Examples for such changes are: removal of a link to simulate the effect of rerouting, increasing/decreasing the capacity of a link, or installing an additional link. The user is provided with these configuration possibilities in the form of simulator specific pop-up dialogs (see Fig. 5(b)).

When the specification of the simulation request is finished the user presses the "Start Simulation" button on the first tab to submit the simulation job. Thereupon the GUI transforms the user input into an simulation request message, which is a XML document. The message consists of three main sections:

Topology information: The topology is represented as a `<BGPTopologyTree>` that is used in many places within the Intermon toolkit. It is normally produced by InterRoute and describes the Autonomous Systems and their connections.

User applied changes: All changes that the user has applied in the GUI are stored in a `<changes>` section which consists of one or more `<actions>` that describe the modification. An exemplary `<changes>` section that contains a single user modification (a link removal) is shown in Fig. 6.

Simulator specific parameters: Each simulation ap-

```
<changes>
  <action type="remove">
    <object type="Link">
      <Link>
        <EndPoint type="AS">2000</EndPoint>
        <EndPoint type="AS">3500</EndPoint>
      </Link>
    </object>
  </action>
</changes>
```
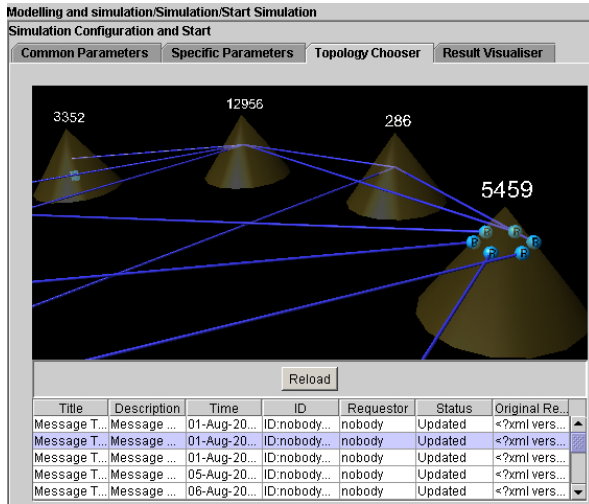
Figure 6. Example XML description of a link removal action

proach has some simulator specific parameters that are encoded within a corresponding XML element whose tag name uniquely identifies the simulator. Within this section, parameters are always stored as property/value pairs. Fig. 7 shows an example.

4.3 Returning Simulation Results

On completion of a simulation task the simulation manager returns the simulation results and status information to the global controller using a unified XML format reply message (Fig. 8).

As the first item the reply contains the same message id as the simulation request earlier received from the GC. This enables the GC to identify the matching request message.

(a) "Topology Chooser" tab
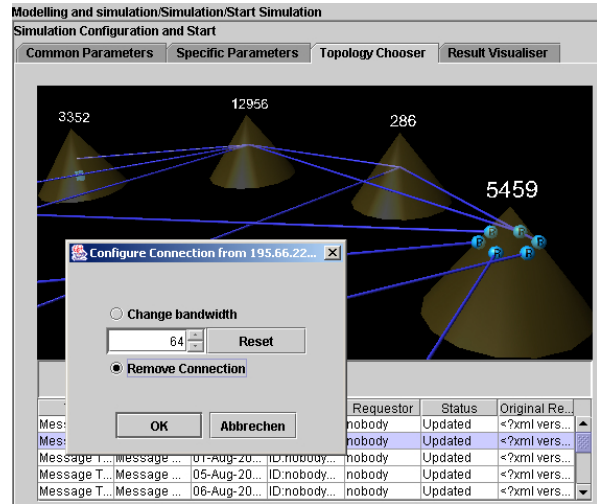
(b) A simulator specific dialog pops up

Figure 5. Screenshots of the "Topology Chooser"

```
 <RTC-FSIM>
   <Property name="Simulation_time" value="150"/>
   <Property name="Sampling_interval value="0.1"/>
 </RTC-FSIM>
```

Figure 7. Example XML description of simulation specific parameters

```
<Simreply id="Message ID">
  <source> ns2 | interip | rtcfsim | tssim </source>
  <comment> …Description…    </comment>
  <duration> …Processing time...    </duration>
  <status> ok | broken | running | canceled </status>
  <results>
    <subresult storage="extern" >
      <comment> …Description…    </comment>
      <content> …Keywords…</content>
      <url> …URL of results...    </url>
    </subresult>
    <subresult storage="inline">
      <comment> …Description…    </comment>
      <content> …Keywords…    </content>
      <data>
        …Results as CDATA…
      </data>
    </subresult>
  </results>
</Simreply>
```

Figure 8. Format of the simulator reply message

It also includes some descriptive comment on the content of the reply and indicates the duration and the status (e.g. ok, broken, running, canceled) of the simulation. However, the most important part of the reply is the results section, which shows the actual outcome of the simulation. As the simulators may produce several different types of results, the results section can consist of several subresult sections, each describing a certain measure (e.g. the time series of bandwidth or delay). The subresults have their own descriptive comment and identify the type of their content (e.g. bandwidth, delay, jitter, etc.). The actual simulation result data can either be included in the reply in-line, or it can be stored in a separate file externally on an FTP, HTTP, or other server. In the former case, the subresult section includes a data section, which can contain the results in a simulator specific format. In the latter case, the section contains a URL pointing to the data file storing results in a simulator specific format.

5 Visual Data Mining

The visualization component is a central unit within the Intermon system. This unit has two components: the topology visualization and the visual data mining (VDM) unit. The topology visualization was designed to give a visual representation of topology data, e.g. the BGP topology data coming from the InterRoute tool. This area is out of the scope of this article. The other visual component, visual data mining, was created to interact with simulation and measurement tools. It aims to provide an easy way to present data sets from the simulation and measurement

tools. To support deeper data analysis the data mining unit can be activated and individually changed on runtime.

To be independent from specific data of different tools every tool has to provide an import filter, which transfers proprietary data into a common VDM format. Visual data mining algorithms are specified in a XML format. Writing these algorithms by hand is very error prone, so a tool has been developed to guide the user in the creation of these XML descriptions in order to make customization easy.

5.1 Basic Pipelining Concept

The visual data mining pipeline concept is based on the traditional visualization approach in [7, 8]. Its basic idea is to divide the visualization into the three stages filtering, mapping and rendering (see Fig. 9), a combination which is often called a F-M-R pipeline.

Figure 9. FMR pipeline

Within the filtering stage, raw data from a database (or directly from a data producer) is prepared for visualization. The transfer of data to an appropriate 2D or 3D geometrical object is performed by the mapping unit. In the rendering unit, the visual model is displayed on the viewer's screen.[2]

The reference model, based on Robertson and de Ferrari, is an enhancement of the traditional F-M-R pipeline [9]. The main difference to the original model is the integration of a feedback loop, which is initiated by the viewer and can influence the visualization pipeline at several points to make the model more interactive.

5.2 Visual Data Mining Approach

Our Visual Data Mining (VDM) approach is based on the above mentioned reference model by Robertson and de-Ferrari. But to include the data mining aspect into the system the filtering unit was exchanged by a full-blown data mining module able to analyze the data in detail before it is visualized (see Fig. 10). In order to have a configurable and flexible data mining module the system was created from building blocks. These building blocks, called "filters," contain basic data mining algorithms that can be connected together to form a more complex module.

Every pipeline starts with an import filter that converts the data sets coming from the Intermon tools into a com-
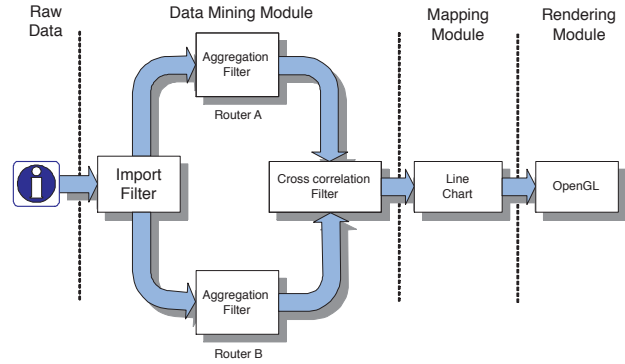
Figure 10. Example VDM pipeline

mon VDM format, which is used throughout the VDM System.

In the example in Fig. 10 the data mining algorithm splits the data into two branches aggregating the data sets for router A and B, respectively. Then, a cross correlation filter calculates the correlation between these two datasets and maps the data to a line chart, which is rendered to a picture.

5.3 Pipeline Generator

The pipeline generator is a tool capable of creating even complex visual data mining algorithms. It can even be used by people who are not experts in visualization and data mining. The graphical user interface offers all available building blocks and helps the user in the process of organizing them into visualization pipelines. Fig. 11 shows the creation of the example pipeline described in Fig. 10.
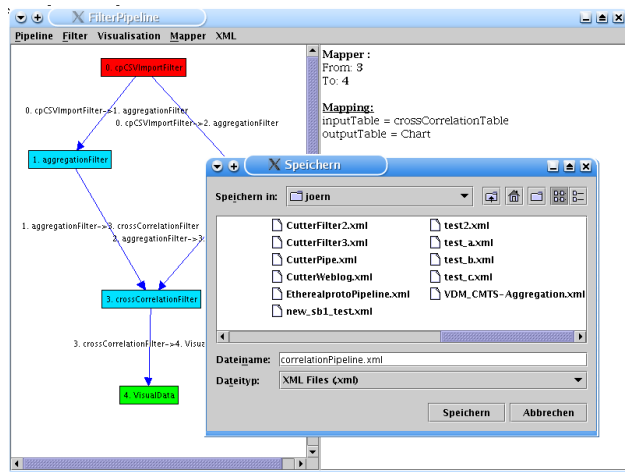
Figure 11. Pipeline generator

---

[2]In most literature concerning visualization, the "viewer" is the operator who performs the visualization.

6 Summary

In this paper we presented the modeling and simulation framework and the visual data mining component of the Intermon project, and their integration in its global messaging architecture and GUI. Special focus was given to the XML-based messaging scheme creating a unified interface to four fundamentally different modeling and simulation approaches with individual parameter sets and output formats, both in the user GUI and towards the data mining component. The distributed nature of the system creates organizational an infrastructural challenges, which are addressed with a global control architecture using the Java messaging system, and a VPN setup to enable and secure communication between the distributed components.

Acknowledgements

References

[1] Baumgartner, F., Scheidegger, M., Braun, T.: Enhancing discrete event network simulators with analytical network cloud models. In: International Workshop Inter-domain Performance and Simulation, Salzburg. (2003) 21–30

[2] Bergholz, G.: Signalflußsimulation für Nachrichtenverkehrsmodelle (german). Technical Report SR-ANC-B0I1, Salzburg Research (2002)

[3] Haber, P., Bergholz, G., Hofmann, U., Miloucheva, I.: Time and rate continuous multiclass fluid simulation model for inter-domain traffic flow simulation. In: First international workshop on Inter-domain performance and simulation (IPS), Salzburg. (2003)

[4] Liu, B., Figueiredo, D.R., Guo, Y., Kurose, J.F., Towsley, D.F.: A study of networks simulation efficiency: Fluid simulation vs. packet-level simulation. In: Infocom. (2001) 1244–1253

[5] Cisco White Paper: Understanding and configuring MDRR/WRED on the cisco 12000 series internet router (2002)

[6] Mahr, T., Dreilinger, T., Vidacs, A.: Time series based simulation architecture. In: First international workshop on Inter-domain performance and simulation, Salzburg. (2003)

[7] Haver, R.B., Nabb, D.A.M.: Visualisation idioms: A conceptual model for scientific Visualisation Systems. In: Visualisation in Scientific Computing. IEEE-Computer Society Press, Los Alamitos (1990) 74–93

[8] Schuhmann, H., Müller, W.: Visualisierung – Grundlagen und allgemeine Methoden (german). Springer Verlag, Heidelberg (2000)

[9] Robertson, P.K., Ferrari, L.D.: Systematic approaches to visualization: is a reference model needed. In: Scientific Visualisation. Academic Press, Los Alamitos (1994) 287–305