SFM3: A Service-based Flow Traffic Measurement Management Model for IP Networks

Leobino Sampaio and José A. Suruagy Monteiro

UNIFACS — Salvador University
Computer Networks Research Group (NUPERC)
R. Ponciano de Oliveira, 126 – 41950-275
Salvador - BA - Brazil
Emails: {leobino,suruagy}@unifacs.br

***Abstract*— It is a great challenge to analyze all the data obtained from the variety of measurement tools existing today. This task is even more complicated when we need to extract useful information for managing networks belonging to different domains due to systems and equipment heterogeneity. This is also true with traffic flow measurement data. Although much effort has been placed in its standardization by IETF's IPFIX Working Group, still flow information data analysis falls behind. This goal can be achieved by the use of SOA architecture, through Web Services. This article proposes the SFM3 model, which provides heterogeneous systems integration using standard based technologies. We report on our experiences in implementing such model in our national academic IP Network through the use of Apache Axis Web Service, OSU Flow-tools and Cisco Netflow, by identifying, collecting and providing useful information in a platform independent way.**

I. Introduction

It's well-known the diversity of tools and techniques existing today with the purpose of measuring IP Networks in order to verify the quality of service that is being offered to its users. The reason for this variety is the complex technologies that exist in the administrative domains, resulting in each of them having its own hardware and software solutions. In addition, evaluating accurately specific parameters is a hard task when it has to be done by only a single tool.

Measuring with the use of different tools has its advantages once it is possible to reach higher accuracy levels in the results. On the other hand, data handling by management applications becomes a challenge if they do not follow access and presentation standards. Beyond the uniform network data access, would be interesting for the management applications to have access to the measurement's information that belongs to different domains as well as correlating them. For flow traffic measurements, for example, this necessity becomes even more evident, since in general it crosses the boundaries of a single administrative domain.

In practice, there are many existing solutions for flow traffic measurement infrastructures that generically have been based on the RTFM model [1] and implemented by NeTraMet [2], NetFlow [1], and sFlow [3]. One of the main characteristics of these solutions is the presence of a flow collector element that implements its proper methods of communication with the measurement devices. For this reason, in order to prevent a great diversity of solutions for this data exchange, the IPFIX group (IP Flow Information Export) [4] have been working in the standardization of this communication.

Although groups like IPFIX have been working on these matters, one of the questions that remain open is how management applications can obtain useful information from the data stored by these collectors and devices. Especially because it is out of the scope of IPFIX activities and there are a large number of proprietary solutions that are not adaptable to new environments and systems.

In this direction, the Web Service standards have been widely used as a key solution to problems related to applications interoperability and integration. This trend has influenced some groups to implement middleware, libraries, and tools that provide a cooperative use of geographically distributed resources like a single unit or just one management environment. Examples of such systems are E2EpiPEs (End-to-End Performance Initiative Performance Environment System) [5], INTERMON (Advanced architecture for INTER-domain quality of service MONitoring, modeling and visualization) [6] and MONALISA (MONitoring Agents using a Large Integrated Services Architecture) [7].

However, the majority of these efforts have been directed to active measurement tools and there has been very little effort on passive measurements tools, especially of traffic flows. Hence, the main objective of this paper is to address significant aspects related to these subjects, proposing the SFM3 (**S**ervice-based **F**low traffic **M**easurement **M**anagement **M**odel) that focus on the use of Web Services. In this model, the XML language is seen as a key technology to exchange traffic flow measurements information through the network with the use of standards-based technologies and coping with aspects related to interoperability. This paper also presents our experience developing a services framework for RNP (The Brazilian Research and Education Network) which backbone [8] connects 27 nation-wide Points of Presence (PoPs).

The remainder of this paper is organized as follows. Section II describes the related work. Section III presents the SFM3 model and its framework. Experiments are presented in Section IV. Finally, Section V concludes the paper.

II. Background and Related Work

A. IPFIX — IP Flow Information Export

Nowadays, there are a number of incompatible IP flow information export systems. The IETF's IPFIX working group [4] was created with the goal of standardizing an IP flow information export system. Its activities include the development of a basic common IP traffic flow technology that will be available on most of the devices, helping the development of generalized management and flow analysis tools. However, the scope of IPFIX efforts is restricted to flow identification and its transport to the collectors and there are currently no mechanism to access the data stored on them.

B. SOA — Service-Oriented Architecture

The Service-Oriented Architecture [9], [10] is a programming paradigm where the software provide its functionalities in the form of services. This characteristic brings a notable flexibility in building applications that will use resources belonging to different administrative domains. In summary, the architecture is composed of three components: the service provider, the service registry, and the service client. In general, SOA have been implemented by using the Web Services technology that will be explained next.

C. Web Services

Web Services is an initiative led by the World Wide Web Consortium (W3C) where applications use/provide services through the use of standard XML-based technologies, enabling the communication between two computer applications without human interference. Due to the ubiquity of the web and its widespread adoption, one of the advantages of this approach is that the communication occurs over ubiquitous protocols like HTTP and facilitates the interoperability through most firewalls policy constraints.

Basically, there are three key technologies in the Web Services framework. They are: WSDL (Web Services Description Language) [11] that is a XML format used for describing Web Services; SOAP (Simple Object Access Protocol) [12] is the lightweight communication protocol that provides the messages transport; and UDDI (Universal Description, Discovery, and Integration) [13] that makes possible the automatic service description and discovery.

D. ESOA — Extended Service Oriented Architecture

The need for issues like service management, orchestration, transaction management and coordination, and security made Papazoglou [14] propose an extended layered architecture that utilizes the basic SOA constructs at its bottom layer and classifies the other services in terms of composition and management, as shown in Fig. 1.

III. SFM3 — Service-based Flow traffic Measurement Management Model

Currently, flow traffic measurements have been done with the use of proprietary solutions that require rude collected data access methods by flow measurement tools. In general, the analysis applications need to implement case-by-case solutions to have access to these data, having to be concerned with specific details of each environment platform. An ideal scenario would be the one where the analysis applications would not need to implement these access methods, and its only concern would be to request the information of interest.

In order to make possible such a scenario, it would be needed that all the applications use a standard language for data representation and transport in networks. Looking at addressing these aspects is that the SFM3 model shown in Fig. 2 was proposed.

This model follows some principles of the ESOA architecture in which services components functionality is distributed in three layers: Basic services, Composition and Management. In addition, this approach proposes not only the functionality classification but also the components themselves, according to their role in the task of dealing with the provided information.

SFM3 assumes Web Services as the main technology and all of the information is represented by a XML schema and are transported between the components using the SOAP protocol. Also, each component has a set of services that is provided to the other components. These services are published at UDDI public directory infrastructure allowing automatic discovery and request.

The components of the Basic Services layer has a fundamental role in the SFM3 approach, since it is where all of the data and information comes from. In this layer, the measurement device identifies the traffic flows and exports them to the collector, which selects the flows of interest and exports them to a DBMS software. Once the information about a flow is requested by other components, there is a component responsible for communicating with the DBMS software, processing the data, and providing the right requested information. In the composition layer is located the components responsible for providing services that are the result of the combination of other services. In the management layer are located the components responsible for activities related to the management of flow resources, access control, and presentation to others management applications.

This division in layers, facilitates the access to the information about the traffic flow of the net, since now there is a responsible component for each type of information and such information are available through services that use standards based technologies.

Thus, independently of the technology that a given net domain is using for flow collection, storage, and manipulation, what matters is the interface of its components, the type of manipulated information, and form of access to this information (all described in WSDL documents).

The following example illustrates how the platform independence and the multiple layers make a difference. For instance, an administrative domain can use collectors and flow meters through the use of the NeTraMet package, storing the collected information in a hash file to speed up the consultations to the collected data. Then, this domain sets a group of services to determine the behavior of these collectors and meters

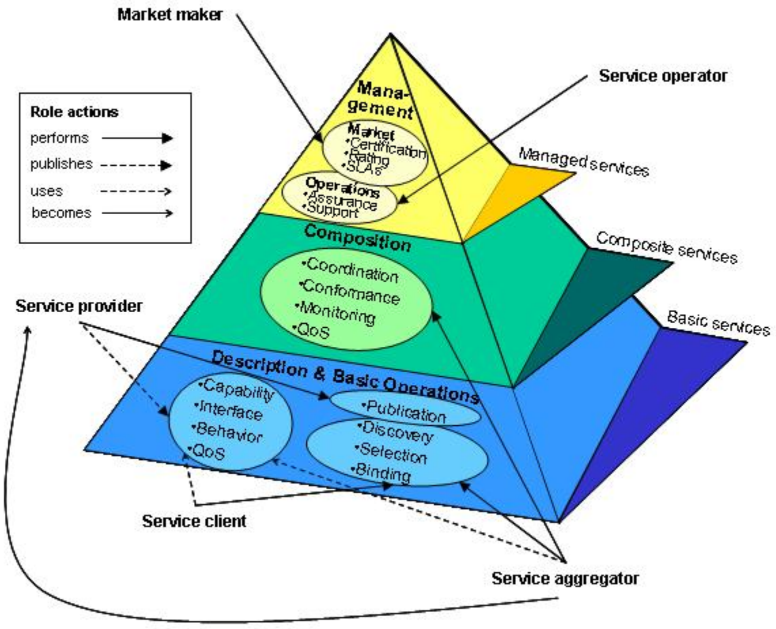


Fig. 1. Extended SOA [14].

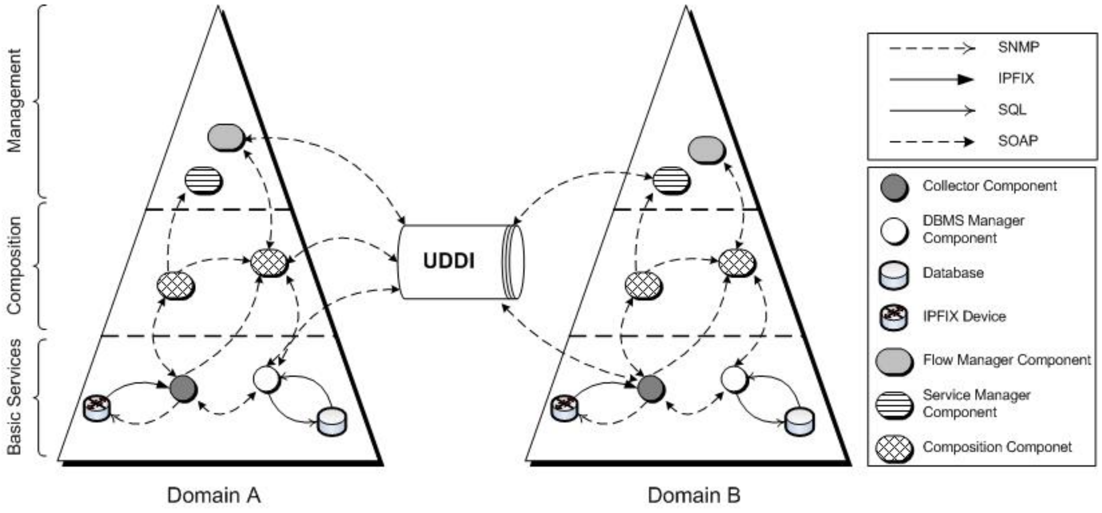


Fig. 2. SFM3 Model.

(configured through the SRL's rules). Notice that for the users of these services, it is of little importance the technology that is being used. Moreover, this domain sets another group of services that gives specific information on the flows (e.g., total of octets between two subnets in a given time interval). On the other hand, another administrative domain can use Netflow for identifying the flows that go through its Cisco routers, and uses the Flow-tools package for the data collection and storage of the flows information in MySQL databases. This last domain already implements other forms of stored information recovering and other mechanisms to determine the collector behavior; still it offers the same set of services.

Thus, as it can been seen in Fig. 3, this model proposes a framework of services between layers, in which the software components deal with traffic flow information in a completely platform independent manner.

IV. Prototype Implementation

This section is dedicated to describing the technical details of a services framework prototype implementation for RNP's backbone. In the following subsections it will be discussed the implementation of the collector component, the DB component, and the client management application, SaManTa.

A. Prototype description

This prototype was implemented in RNP's backbone using Netflow to identify the traffic flows that pass through the backbone routers. OSU's flow-tools package [15] was used to

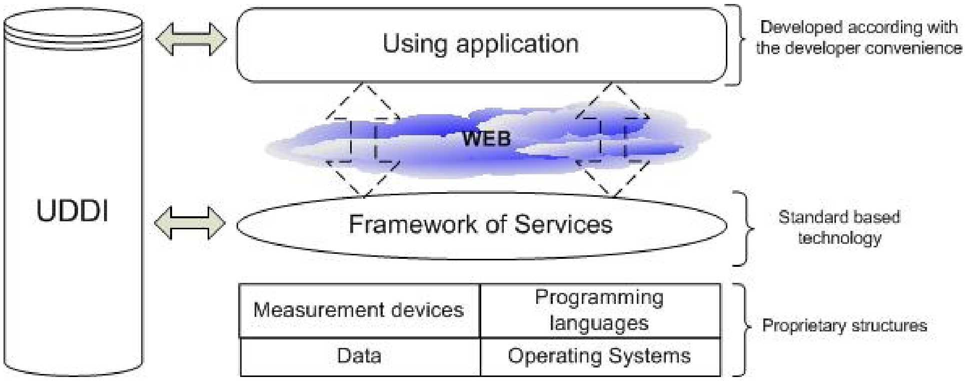


Fig. 3. Framework of services between components layers.

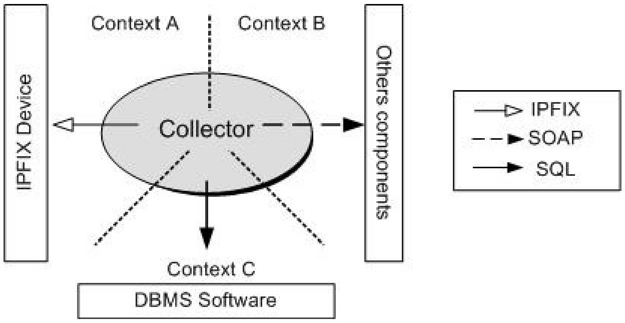


Fig. 4. Collector component contexts.

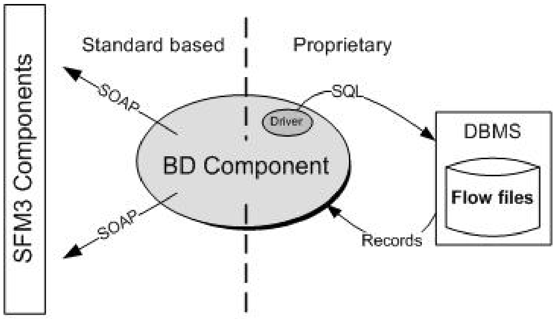


Fig. 5. DB component contexts.

collect, process, filter, and export the flow information received from the routers. To deploy Web Services the Apache Axis Java Web Services [16] was used, and the MySQL database server was used to store the selected flows.

B. Collector component description

The collector component is responsible for communicating with the measurement devices. For this communication, the collector component must use the IPFIX standards. Moreover, this component should implement functionalities that allow the selected flows of interest to be exported to database servers.

In the prototype implementation, the collector was implemented as shown in Fig. 4, where basically three context were involved. One that is related to the relationship with the measurement device; another that involves the export of flow records to a DBMS server; and finally another where it is established the relationship with other components.

As described in Fig. 4, the collector uses the flow-tools package to capture the flow records exported by Cisco's Netflow. This component provides services that determine its behavior. These services set values in a configuration file that is used by a script that sets the parameters passed to flow-tools software (e.g. flow-capture, flow-cat, flow-nfilter, flow-tag, and flow-export).

During the flow records collection process, this script sets flow-capture parameters like: the configuration compression level; the directory path for flow files records; the number of times to create new files per day; and the IP addresses from the allowed measurement devices. The same happens for the selection of flows records. The script sets flow-nfilter processing parameters like filter primitives, valid matches and filter file names. Finally, in the process of exporting data to MySQL, the collector reads its configuration files to catch database information and export flow-records according to flow-nfilter's parameters (also stored in the collector configuration file). Fig. 6 resumes the technologies used by the collector.

C. DB component description

The DB component does the processing of data stored in the database and provides the results as useful information to analysis applications. Such characteristic gives much flexibility to client applications that don't need to deal with specific aspects related to DBMS software (Fig. 5). In addition, all the data is available on demand according to the client applications needs.

In order to implement these aspects of the DB component in the prototype, it was necessary to use MySQL JDBC driver to access the DBMS server. The data model created to store the flows was based on Cisco's Netflow fields and some index was created for a better database query performance.

All of the services made available by the DB component are based in queries previously prepared to extract specific information about the flow records. These queries are executed by a DAO (Data Access Object), giving a high level of abstraction to the class responsible for the service (Fig. 7).

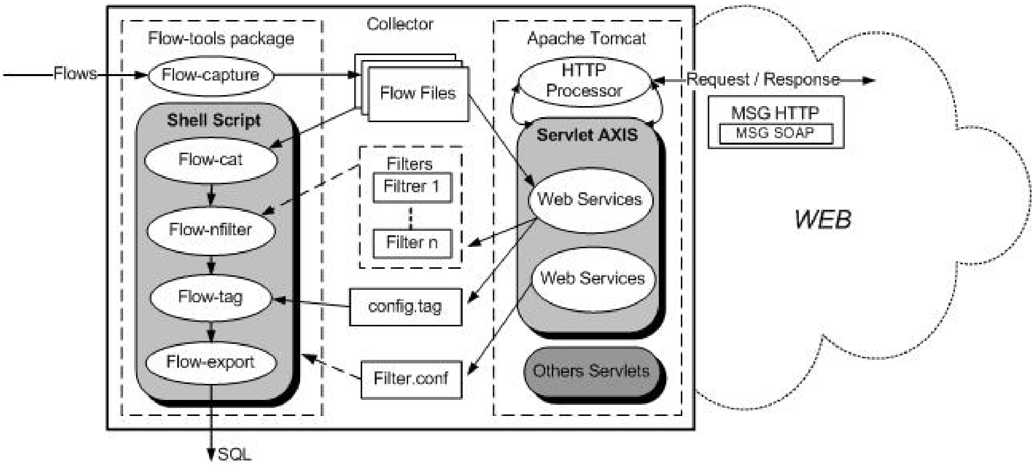


Fig. 6. Technologies used by the collector component.

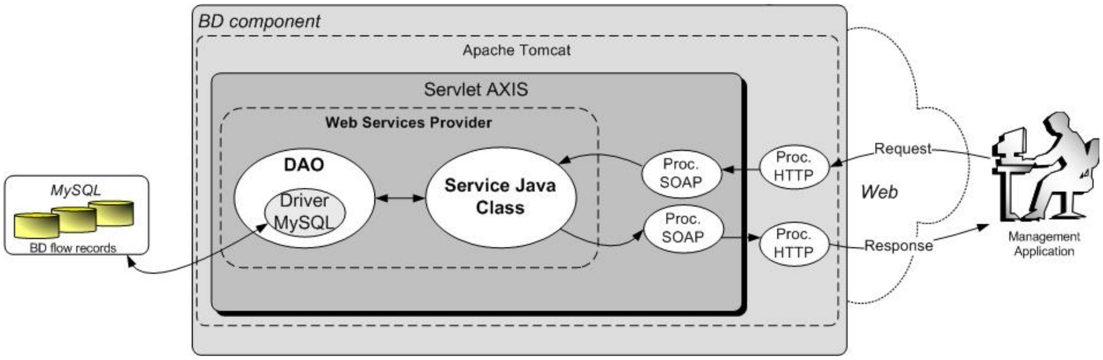


Fig. 7. Technologies used by the DB component.

D. Developed Services

A set of services were created for each component of SFM3's Basic services layer aiming at making available flow information as well as managing the collectors distributed among several PoPs.

The following services are some examples of services created for the collector component:

- **Automatic configuration of flow collection rules:** These services establish the criteria for which flow records should be exported to the database servers;
- **Real time reception from new measurement devices:** Once can exist more than one measurement device for a collector, these services was created in order to specify which device is allowed to export flow records;
- **Data removal from flows records with a given characteristic:** After collected and exported to database, the files with flow records stay in the file system of the collector and need to be removed. Therefore these services establish the these criteria;
- **Anonymization of flows records with a given characteristic:** One of the security concerns with the flow measurements is about the privacy of the information collected. With anonymization, the flow records can be re-agrouped according with subnets;
- **Delivery of preliminary statistics about specific flows records:** These services allow the collector to give some statistics information about a specific group of flow records;
- **Data base servers configuration:** The collector export flow records to the database. These services configure specific database parameters, like user name, password, and IP.

Services were created for the DB component aiming at recovering the information stored in mySQL. For example:

- **Flow records database maintenance:** These services is responsible to manipulate the flow records stored in the database, dropping some of them according with some criteria;
- **Maintenance of measurement devices, domains and points records:** Once the flow records is exported to the database by more the one collector these services deal with all the information necessary to control the collectors and the information exported by them;
- **Maintenance of indexing structures:** In order to make fast the queries about the flows records in the database it is necessary to create indexing structures. These services deal with it these structures, updating, removing and creating new ones;

- **Flow statistics generation:** These services are responsible for manipulating flow records stored in the database in order to give statistical information.

*E. SaManTa — a **S**ervice-b**A**sed **MAN**agement tool of **T**r**A**ffic flows*

SaManTa is a tool developed in Java aiming at integrating in a single environment all traffic flow management functions, based on Web services. Such set of management functions includes collector configuration, statistical data visualization, and traffic characteristics graphical visualization.

Due to the fact that SaManTa is completely based on Web Services, it was used mainly the APIs which belong to the *Java Web Services Developer Pack 1.3* [17], distributed by Sun Microsystems [18]. This package includes all the needed APIs for the development of Web services clients and servers, such as JAX-RPC (Java API for XML-Based Remote Procedure Calls), JAXM (Java API for XML Messaging), SAAJ (SOAP with Attachments API for Java), JAXR (Java API for XML Registries), JAXB (Java API for XML Binding), and JAXP (Java API for XML Processing).

In the sequence, it is described some SaManTa functionalities which use the services implemented in the prototype for the collector and DB components.

1) Collectors configuration module: This module is responsible for passing the collector working parameters, specifying among other things, the flow types that have to be exported to the database. It is possible to reach the configuration options through the main menu, as it can be seen in Fig. 8.

2) Traffic Matrix Module: One of the results that can be obtained from flow information is the traffic matrix. In other words, it can be shown the total amount of octets and packets among each source and destination PoP, detailing the service types and/or protocol types.

To this purpose, this module exhibits a window with configuration options for the traffic matrix. Among the options are the unit type (octets, packets or flows), the PoPs involved and the period of time. After these choices are made the data is shown as illustrated by Fig. 9 (A) and (B).

V. Conclusions and future work

The perspective of using a great variety of IP network performance evaluation tools has motivated several initiatives in integrating management environments. In the specific case of traffic flows, there is not a standard access model to the data stored in flow collectors, which in is great majority are managed mainly via proprietary solutions.

These factors motivated the elaboration of an access and data management model for the flows information divided in layers, called SFM3. This model presents different flow information presentation levels which varies from raw data to specific data of a given user.

Each SFM3 model layer can be seen as a service framework for the next level, making easier the process of developing management applications as well as their integration. Based on the description of each participating component, a Java prototype was developed using Web services technologies. Through the experiments with this prototype it was possible to observe and evaluate the feasibility of implementing the elements described in the model. In one of the experiments it was confirmed the interoperability due to the use of Web services. Because they are based on technologies such as XML, they are adequate to the majority of proprietary environments. With the development of the traffic matrix application in the second experiment, it was possible to verify the advantages in using services oriented approach for traffic flow measurements in relation to measurements management.

The experience presented in this paper can be used as a basis for future works in this area. Mainly regarding the transformation of the information received by the intermediary layer and its transformation in useful knowledge for management. This knowledge will allow a more proactive behavior of the monitoring/management systems, making possible adaptative solutions with less human interference.

Acknowledgments

The authors want to thank all the support given by the Brazilian Research and Education Network (RNP) and the Brazilian National Council for Scientific and Technological Development (CNPq) to the QoS2 WG and IQoM Project.

References

[1] N. Brownlee, C. Mills, and G. Ruth., "RFC 2722: Traffic flow measurement: Architecture," Oct. 1999, Status: INFORMATIONAL.

[2] N. Brownlee, "RFC 2123: Traffic flow measurement: Experiences with NeTraMet," Mar. 1997, Status: INFORMATIONAL.

[3] P. Phaal, S. Panchen, and N. McKee, "RFC 3176: InMon corporation's sFlow: A method for monitoring traffic in switched and routed networks," 2001.

[4] IPFIX, "IPFIX: IP flow information export," http://www.ietf.org/html.charters/ipfix-charter.html, 2003.

[5] Internet2, "E2E piPEline: End-to-end performance initiative performance environment system architecture," http://e2epi.internet2.edu/E2EpiPEs/e2epipe11.pdf, 2003.

[6] INTERMON, "INTERMON: Advanced architecture for INTER-domain quality of service MONitoring, modelling and visualisation," http://www.ist-intermon.org, 2003.

[7] MONALISA, "MonALISA: Monitoring Agents using a Large Integrated Services Architecture," http://monalisa.cacr.caltech.edu, 2004.

[8] RNP2, "RNP2: Backbone map," http://www.rnp.br/backbone/, 2003.

[9] W3C, "Web services architecture," http://www.w3.org/TR/ws-arch/, Aug. 2003, Status: Working draft.

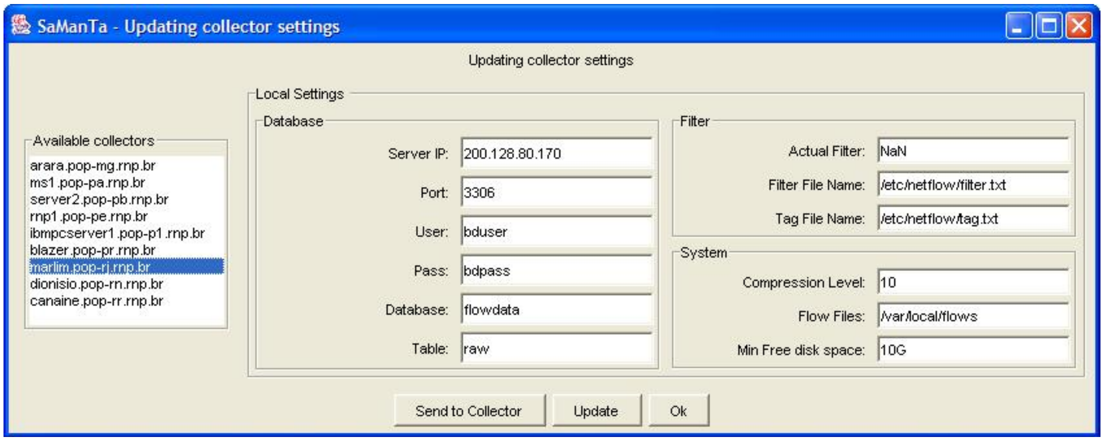
[10] M.P. Papazoglou, "Service-oriented computing: Concepts, characteristics and directions," Dec 2003, Keynote for the 4th International Conference on Web Information Systems Engineering (WISE 2003), to appear in IEEE CS.

[11] Roberto Chinnici, Martin Gudgin, Jean-Jacques Moreau, and Sanjiva Weerawarana, "Web services description language (WSDL) version 1.2 part 1: Core language," http://www.w3.org/TR/2003/WD-wsdl12-20030611, June 2003, W3C Working draft.

[12] W3C, "Soap version 1.2 part 0: Primer," http://www.w3.org/TR/2003/REC-soap12-part0-20030624, June 2003, W3C Recommendation.

[13] OASIS, "Uddi version 1 specifications," http://www.oasis-open.org/committees/uddi-spec/doc/contribs.htm#uddiv1, June 2002.

[14] M. P. Papazoglou and D. Georgakopoulos, "Introduction: Service-oriented computing," *Communications of the ACM*, vol. 46, no. 10, pp. 24–28, Oct. 2003.



SaManTa - Updating collector settings

Updating collector settings

Available collectors

arara.pop-mg.rnp.br
ms1.pop-pa.rnp.br
server2.pop-pb.rnp.br
rnp1.pop-pe.rnp.br
ibmpcserver1.pop-p1.rnp.br
blazer.pop-pr.rnp.br
marlim.pop-rj.rnp.br
dionisio.pop-rn.rnp.br
canaine.pop-rr.rnp.br

Local Settings

Database

Server IP: 200.128.80.170
Port: 3306
User: bduser
Pass: bdpass
Database: flowdata
Table: raw

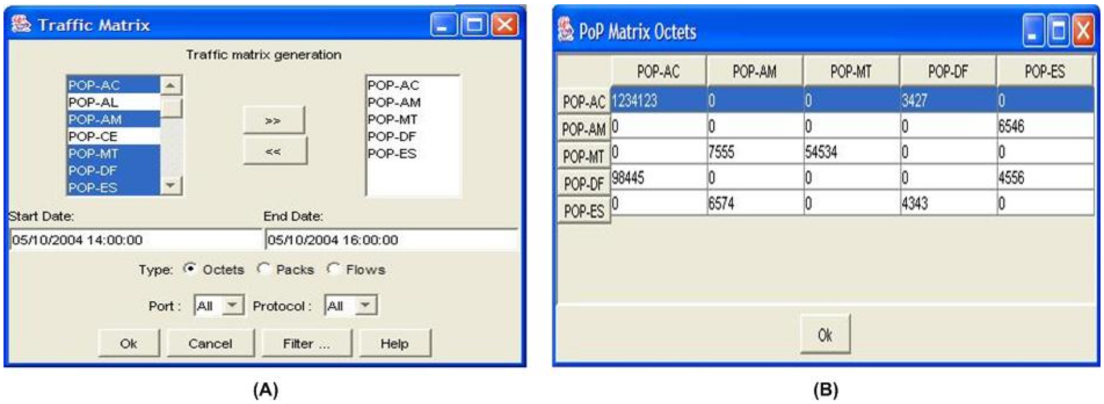
Filter

Actual Filter: NaN
Filter File Name: /etc/netflow/filter.txt
Tag File Name: /etc/netflow/tag.txt

System

Compression Level: 10
Flow Files: /var/local/flows
Min Free disk space: 10G

Fig. 8. Collectors configuration through SaManTa.



Traffic Matrix

Traffic matrix generation

POP-AC
POP-AL
POP-AM
POP-CE
POP-MT
POP-DF
POP-ES

POP-AC
POP-AM
POP-MT
POP-DF
POP-ES

Start Date: 05/10/2004 14:00:00
End Date: 05/10/2004 16:00:00

Type: ● Octets ○ Packs ○ Flows

Port : All Protocol : All

PoP Matrix Octets

	POP-AC	POP-AM	POP-MT	POP-DF	POP-ES
POP-AC	1234123	0	0	3427	0
POP-AM	0	0	0	0	6546
POP-MT	0	7555	54534	0	0
POP-DF	98445	0	0	0	4556
POP-ES	0	6574	0	4343	0

Fig. 9. Generating a traffic matrix.

[15] Steve Romig, Mark Fullmer, and Ron Luman, "The OSU flow-tools package and CISCO netflow logs," in *Proceedings of the Fourteenth Systems Administration Conference (LISA-00)*, Berkeley, CA, Dec. 3–8 2000, pp. 291–304, The USENIX Association.

[16] Apache, "Apache Axis," http://ws.apache.org/axis, 2004.

[17] Sun Microsystems, "Java Technology and Web Services," http://java.sun.com/webservices, 2004.

[18] Sun Microsystems, "Sun microsystems," http://www.sun.com, 2004.