G E N I

Global Environment for Network Innovations

Milestone 4
GENI Real-Time Measurements Software Architecture

Document ID: GENI-MS4-ERM-June09-v1.0

June 01, 2009

Prepared by:
F. Fidler, C.P. Lai, and K. Bergman
Dept. of Electrical Engineering, Columbia University New York,
500 W. 120th Street ,
New York City, NY 10027
under Project Nr. 1631
“Embedding real-time measurements for cross-layer communications”

TABLE OF CONTENTS

1 Document Scope

This section describes this document's purpose, its context within the overall GENI project, the set of related documents, and this document's revision history.

1.1 Executive Summary

This technical note presents the results obtained in work package "Milestone 4: Develop a software architecture" of Project Nr. 1631, "Embedding real-time substrate measurements for cross-layer communications".

This milestone deals with the development of a measurement framework based on GENI real-time measurement requirements and other developments/resources within the GENI prototyping activities. It included interacting with other prototype efforts to identify and leverage relevant activities, software architectures, and products.

In Section 2 we give an overview on the concept of a unified measurement framework (UMF) for GENI as proposed in our previous milestones [erm09_1, erm09_2]. Section 3 deals with a number of software architectures dedicated to network measurements which could serve as an interface between the UMF, the control framework, and the GENI experimenter. In Section 4 we assess several network management protocols and data exchange formats with respect to their ability of exchanging measurement and control information between the substrate's performance monitors and the UMF, as well as between the UMF and the GENI control frameworks.

1.2 Related Documents

The following documents are related to this document, and provide background information, requirements, etc., that are important for this document.

1.2.1 GENI Documents

Document ID	Document Title and Issue Date
GENI_QR_ERM_Apr09	1Q09 Status Report
GENI-INF-PRO-S1-CAT-01.3	GENI Infrastructure Substrate Catalogue
GENI-MS1-ERM-March09-v1.1	Technical Note 1, Milestone 1
GENI-MS2-ERM-March09-v1.0	Technical Note 2, Milestone 2

1.3 Document Revision History

The following table provides the revision history for this document, summarizing the date at which it was revised, who revised it, and a brief summary of the changes. This list is maintained in chronological order so the earliest version comes first in the list.

Revision	Date	Revised By	Summary of Changes
1.0	01 June 09	F. Fidler	Initial draft

2 Introduction

To evaluate the GENI requirements for real-time user access to measurement data, we have previously assessed the capabilities of GENI's (future) infrastructure with respect to real-time measurements [erm09_1]. As pointed out in [erm09_2], interfacing available performance monitors (PMONs) to the control framework and to the access point of the GENI researcher in a straightforward way may lead to the following obstacles:

- Depending on the number of available PMONs, the number of required interfaces might become very large.
- Designing the interfaces requires detailed, vendor-specific knowledge about the control mechanisms of the PMONs and of how measurement data is exported. Therefore, some abstraction is highly desired.
- A certain amount of editing and preparation of the measured data prior to the delivery to the researcher could be desirable.
- Extensibility of the measurement framework and manageability is difficult if each PMON has to be controlled individually.

Therefore, rather than interfacing every performance monitoring device within the substrate directly with the control framework and the experimenter, we recommended the design and use of a unified measurement framework (UMF) [erm09_2]. The UMF represents a universal measurement platform which can be accessed by the control framework and the researcher via a limited number of well defined interfaces (cf. Figure 2-1). The main tasks and functionalities of the UMF are:

- Acquiring measurement data from the various performance monitoring devices within the optical substrate,
- abstraction of measurement capabilities and equipment from several manufacturers,
- provide a single point of access,
- basic processing of the measurement data, e.g., to extrapolate signaling data for the researcher,
- provisioning of some storage capacity for non-time-sensitive measurements,
- interfacing to the researcher via a unified measurement interface which allows requesting and controlling certain measurements and delivers the measured data, and
- interfacing with the control framework so that the measurement framework (or a subset of PMONs) can be allocated to a requesting GENI researcher and reconfiguration of the slice is made possible.

As such, the UMF presents a uniform view and an abstraction of the measurement capabilities within the substrate and makes them accessible/sliceable to a control framework.

While some hardware related implementation aspects of the UMF were previously discussed [erm09_2, erm09_3], the adequate design of a specific software architecture/framework to enable the efficient exchange of information to a UMF is still an open question. Within this milestone, we therefore evaluate a number of existing software measurement architectures (e.g. SILO [silo09_1], perfSONAR [perfsonar09_1], OMF [rutgers09_1],…) which are already associated with GENI, as well

as analyze appropriate data exchange formats (e.g. XML, …) and networking protocol languages (e.g. SNMP, TL1, …) with respect to their applicability within a unified measurement framework.

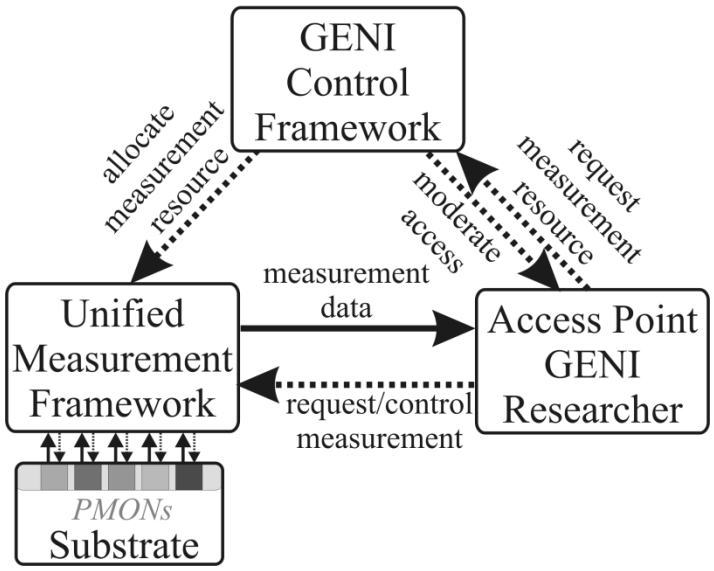

Figure 2-1: Schematic of interaction between performance monitors (PMONs) of substrate, unified measurement framework (UMF), control plane, and GENI researcher.

3 Software measurement frameworks

3.1 SILO

Our ongoing cooperation with the Renaissance Computing Institute (RENCI)/BEN provided us with some insight concerning their NSF funded "Services Integration, ControL and Optimization" (*SILO*) architecture [silo09_1] for the future internet. The objective of this project is to formulate a framework for a non-layered internetworking architecture in which complex communication tasks are accomplished by combining elemental functional blocks in a configurable manner. This internetworking architecture consists of

- Building blocks of fine-grain functionality,
- explicit support for combining elemental blocks to accomplish highly configurable complex communication tasks,
- control elements to facilitate (what is usually referred to as) *cross-layer communication*.

SILO takes a holistic approach to network design, allowing applications to work synergistically with the network architecture and physical layers so as to meet the application's needs within resource availability constraints. It is also already positioned to take advantage of hardware-based performance-enhancing techniques such as embedded real-time measurement based cross-layer optimization. In that sense, SILO already pursues goals [stevenson07_1] which are in accordance with our unified measurement approach, such as

- flexibility, extensibility, and scalability,
- support for a scalable unified architecture, and
- explicit cross-service interactions and optimization.

The fundamental building blocks in the SILO framework are so-called *services*. A service is a well-defined and self-contained function performed on application data, and addresses a separate, simple and reusable function. Hence, the architecture provides a much finer granularity than current protocols which typically embed complex functionality. Each service provides an upper and lower data interface (as in today's layered approach), but also provides control interfaces which are referred to as *knobs*. Finally, each service provides a list of partial ordering constraints with respect to other services. Beyond these constraints, any set of services can be composed into a stack (i.e. a *silo*) in any order. A *method* is an implementation of a service that uses a specific mechanism to carry out the functionality associated with the service. Ideally, a silo structure and all related state information persists for the duration of a connection. A *control agent* is an entity residing inside a node, which is responsible for composing a silo for an application stream, and appropriately adjusting all the service- and method-specific knobs. Composing a silo refers to determining the subset of services it contains, their order in the stack, and the method implementing each service.

With respect to cross-layer communication [baldine07_1], the SILO architecture can be viewed to "operate in layers, control across layers." As of today, services and methods are required only to provide a minimal interface, hiding internal details. Whereas traditional protocols are only required to provide invocation methods (APIs), protocols in the SILO architecture are required to provide a minimal control interface as well. Beyond this, the methods can be designed and implemented in isolation. As the control agent has a unique view into the knobs of every method in the silo and can embody all the integrated control concerns, "cross-layer" (or, more appropriately, "cross-service") can become part of the architecture.

Currently, the SILO prototype APIs are implemented as a series of user-space components implemented in *C/C++* interconnected using traditional UNIX mechanisms (e.g. UNIX sockets, shared memory, message queues etc). The SILO ontology is an *XML* based description of the relationships between SILO services and methods used to create and operate SILOs. It describes interfaces between services as well as service and method control interfaces.

Real-time measurements within SILO

The real-time measurement capabilities within GENI's infrastructure [erm09_1] could be interfaced and be accessible as a collection of API calls within the previously described SILO architecture. This would represent a new component to the SILO framework, one that is specific to the operating environment and allows information to flow into the SILO framework (for subsequent use by an experimenter using SILO) from any framework that supplies measurement data (in this case, the UMF). The API may be fronted by specific 'measurement' SILO services with control interfaces (knobs) to adjust the measurement accuracy or time-scale. Additional ordering constraints would define where to fit the measurement service(s) into the SILO stack.

3.2 PerfSONAR

PerfSONAR [perfsonar09_1] is a framework that enables (multi-domain) network performance information to be gathered and exchanged in a multi-domain, federated environment. The goal of perfSONAR is to enable ubiquitous gathering and sharing of this performance information to simplify network management and to allow next-generation applications to tailor their execution to the state of the network, for example to enable cross-layer communication. PerfSONAR's services act as an intermediate layer, between the performance measurement tools and the diagnostic or visualization applications. This layer is aimed at making and exchanging performance measurements between networks, using well-defined protocols. Similar to SILO, perfSONAR is a services-oriented architecture, meaning that the set of elementary functions have been isolated and can be provided by different entities called *services*. Some major perfSONAR services [hanemann05_1] which are in accordance with our goals for a UMF are:

- Measurement point service: Creates and publishes monitoring information
- Measurement archive service: Stores and publishes monitoring information
- Lookup service: Registers all participating services and their capabilities
- Authentication service: Manages access to services via tokens
- Transformation service: Offers custom data manipulation of existing archived measurements
- Topology service: Offers topological information on networks

Measurements currently include:

- OWAMP: This tool is used to run active tests to collect metrics such as one-way latency, packet loss, or delay variation [boote09_1].
- NPAD: The user provides a target data rate and round-trip-time (RTT) and NPAD attempts to determine if those values are possible, by testing the infrastructure on a limited portion of the path [mathis08_1].
- Traceroute: Traceroute is a network tool used to determine the route taken by packets across an IP network.

- NDT: This diagnostic tool [carlson03_1] attempts to determine what kind of performance the user should expect, and what the current limiting factor is. It allows the end users to test the network path for a limited number of common problems, such as inadequate TCP buffer sizes or duplex mismatches.
- BWTCL: This allows for throughput testing as well as for delay, jitter, and packet loss measurements [boote09_2].
- PingER: 'Ping end-to-end reporting' [matthews00_1] supports network characteristics including throughput, availability, latency and jitter, which provide a broad spectrum of determining end-to-end network performance. PerfSONAR provides a web service interface based on XML that allows queries on the PingER data.

Real-time measurements within perfSONAR

As the examples given in the previous paragraph show, perfSONAR is currently focused on publication of network metrics, but promises to be flexible enough to handle new metrics such as our proposed physical layer performance monitoring capabilities.

With respect to this goal of monitoring link status and providing physical layer performance metrics for GENI, the following functionalities of perfSONAR are of special interest:

- SNMP: The perfSONAR SNMP based measurement archive is able to expose data collected via variables from the Simple Network Management Protocol (SNMP) found on networked devices and stored in Round Robin Databases (RRD) archives. The measurements are collected through external means. PerfSONAR provides a simple interface that is capable of exposing these files. A web service front end provides a uniform method of access using the perfSONAR XML protocols and delivers the data in an unambiguous manner to the user. This capability of publishing SMNP data via perfSONAR could be used to query for all SMNP data from performance monitoring devices (such as the Polatis fiber switch or Infinera DTN, cf. [erm09_1]) along the path.
- Topology: The topology service helps facilitate sharing of network topology. Networks deploy the service and register an XML representation of their network topology with it. Interested clients can then query the service to discover the data they are interested in. The perfSONAR topology service relies on an XML database, Oracle XML DB, to store the topology information. Detailed information about the physical layer infrastructure could be used for example in cross-layer based protection and routing protocols, where information about physically disjoint paths is required [erm08_1].
- Status collector: This service can be configured to obtain status information directly from switches and routers. Currently, the status collector supports querying SNMP devices, Ciena CoreDirectors, Nortel HDXcs and Nortel OME6500s. This service should be extended to other available hardware within GENI's infrastructure (as summarized in [erm09_1]).

The used protocols are based on SOAP XML messages and follow the recommendations of the Open Grid Forum (Network Measurement Working Group) [ogf09_1]. The protocol assumes a set of services types, defines the protocol standard (syntax and semantics) by which services communicate, and allows anyone to write a service. The XML scheme shown in Figure 3-2 is used to represent measurement data. It segments the measurement data presentation in two parts: the metadata and the data. Metadata describes the type of measurement data, the entity or entities being measured, and the

particular parameters of the measurement. The data itself is simply a timestamp and a vector, or array, of measurement values.

Figure 3-1: XML scheme used by PerfSONAR to exchange measurement information [perfsonar09_1]

3.3 ORBIT

One task of the GENI funded project "Control, Measurement, and Resource Management Framework for Heterogeneous and Mobile Wireless Testbeds" (a.k.a ORBIT) is to extend Winlab's OMF (cOntrol and Management Framework) [rutgers09_1] by a measurement framework, so that it can support experiments across heterogeneous testbed resources, with a specific focus on mobile testbeds [geni09_3].

OMF is a generic framework for managing networking testbeds, additionally allowing for the collection of experiment metrics. While the experiment is being executed, data is measured and collected according to the user's description. Measurement points are defined inside the C/C++ application source code or automatically based on XML configuration files. The measurement data (from each client) is transmitted using XDR encoding [sun87_1] and saved to a database (Orbit Measurement Library OML), i.e. a server, for later analysis. A new database is created for every experiment. Within the database, a table is created for each set of measurements collected. SQLite can be used to manipulate the results. Alternatively, a custom script that can export the results to a text file is available. The exported data can then be used to generate graphs or charts for analysis.

Real-time measurements within OMF

The OMF architecture incorporates a number of important functionalities which are relevant for enabling real-time physical layer measurements [erm09_2, singh05]:

- OML data filters provide a standard way of reducing the amount of collectable data to be stored for further analysis. Those filters can be configured and used without rewriting the application code to provide a flexible way to change the data collection and data pre-processing behavior.
- Database queues are used to store the received packets. Using a queue significantly improves the scalability by providing a buffer to avoid packet loss when dealing with experiments that generate bursty data, bearing in mind the slow XDR coding and SQL insertion process.
- An SQL module is used for storing the decoded values in the SQL server for post experiment analysis and data persistency.

Currently it is intended to extend the ability to dynamically steer the experiment control by enabling access to the metrics for further (pre-)processing, which would move the OMF environment more into the direction of enabling real-time measurement based cross-layer experimentation.

3.4 GIMS

Within the GENI funded project "Instrumentation and Measurement for GENI" [geni09_2], a prototype implementation of a *network* instrumentation and measurement system should be developed, tested, and employed. The primary objectives of the design of the GENI Instrumentation and

Measurement System (GIMS) [barford06_1] which are also in accordance with the requirements for performing real-time measurements by means of a UMF include:

- Ubiquitous deployment
- No or at least measureable impact on experiment
- Extensibility
- The ability to also perform physical layer measurements
- Access control
- Archival system for storing measurements
- Abstraction capabilities by means of a central repository
- Tools for facilitating end-user, i.e. experimenter, access.

Within GIMS, the specification of GENI resources is planned to be performed by using two abstract resource types – *links* and *nodes*. *Links* are the physical media (e.g. fiber, air) that connect nodes. *Nodes* are programmable components on which slices can be instantiated and experiments conducted. The assumption is that the combination of these resources forms an infrastructure that will support experiments from the physical through application layer and that measurements will be required throughout.

Real-time measurements within GIMS

The objective of the initial set of instrumentation and measurement systems for GENI is restricted in scope and more specific in focus in order to ensure that sufficient capability is available and operational in a timely fashion. To that end, the specific focus is on *passive packet capture in GENI*. Passive packet capture means the ability to gather, save and analyze packets from *taps* on links in the GENI infrastructure.

Measurement system design specifications of this project are in a very early stage (cf. [geni09_2]), so that GIMS' abilities of providing physical layer performance metrics for GENI and potential required extensions cannot yet be fully evaluated. However, it is anticipated that a UMF could eventually take advantage of the data archival and user interface developed in this project (cf. [barford09_1]).

4 Network management protocols and data exchange formats

In the following chapters we address a number of standard network management protocols and data exchange formats which could be used to exchange measurement information between the PMONs, the UMF, the GENI control frameworks, and software measurement architectures as described in Section 3.

4.1 SNMP

As stated in [erm09_1, geni09_1] a great number of performance monitoring devices incorporated in the substrate (e.g. the optical power monitors in the Polatis fiber switches or the Infinera DTNs) allow the retrieval of their measurement information via SNMP [case90_1]. Usually, software SNMP agents (running on the equipment) expose management data on the managed systems as variables, but the protocol also permits active management tasks, such as modifying and applying a new configuration. While configuration and control operations are used only when changes are needed to the network infrastructure, monitoring operations are usually performed on a regular basis.

SNMP is used in perfSONAR for collecting data, such as status information, from networking equipment like switches and routers. As already identified in [erm09_1], the following GENI substrate equipment allows exchanging measurement information by means of SNMP:

Table 4-1: Real-time measurement capabilities of GENI node equipment using SNMP networking protocol.

Network equipment	measurement capability	deployment	SNMP version
Polatis fiber switch	optical power	BEN	SNMPv3
Infinera DTN ROADM	bit error readout	BEN, ProtoGENI	SNMP (version unknown)
Adva Optical Networking transponder	bit error readout, optical power	MAX, GpENI	SNMPv3
Ekinops transport platform	bit error readout	GpENI	SNMPv2c
Ciena CN4200 switch	bit error readout	GpENI	SNMPv1, SNMPv2c SNMPv3

4.2 TL1

Transaction Language 1 (TL1), a vendor-independent network management protocol, represents another class of instructions widely used to manage network elements and its resources. TL1 can be accessed via an industry standard command line interface (CLI) for managing network elements. Its messages are in plain ASCII text, resulting in easy readability and greater interoperability.

Like SNMP, TL1 is supported by all GENI equipment with measurement capabilities as given in Table 4-1 and - as required for use within the UMF - provides a machine-readable interface definition.

TL1 has a well-defined set of management services for performance, fault, security and other areas of management. For instance, an operator has standard ways to set up performance schedules and receive performance reports from any vendor's TL1-manageable network equipment. Of course, also SNMPv3 provides solid approaches to network equipment identity, discovery, and security.

4.3 SCPI

For accessing individual performance monitoring devices (such as the optical power monitor of the Polatis fiber switch) and interfacing them e.g. with a server incorporating a NetFPGA card (as suggested in [erm09_2]), one could consider SCPI. The “Standard Commands for Programmable Instrumentation” language was developed aiming at a *common* interface language between computers and test instruments. The SCPI Standard is built on the foundation of IEEE-488.2, Standard Codes and Formats. It requires conformance to IEEE-488.2, but is a pure software standard. The SCPI syntax is ASCII text, and therefore can be attached to any computer test language, such as BASIC, C, or C++. It can also be used with test application environments such as LabWindows/CVI, LabVIEW, or HP VEE. SCPI is hardware-independent, which means that SCPI strings can be sent over any physical instrument interface like GPIB, RS-232, VXIbus or LAN networks.

4.4 XML

The interfacing efforts for the UMF will require the development of drivers to access performance monitoring equipment as well as supporting the creation of drivers for the control framework and other measurement frameworks and architectures used by the experimenter. To improve reusability of the software developed and guarantee interoperability, it is essential that the abstraction of the interface to the commercial equipment be sufficiently flexible to accommodate different GENI substrate nodes. This feature can be attained by the use of XML (Extensible Markup Language) based interface and data structure definitions. Self-describing XML representations provide the basic syntax that can be used to share information between different kinds of entities as XML data is stored in plain text format [xml09_1]. This software- and hardware- independent way of storing data allows different incompatible systems to share data without needing to pass them through many layers of conversion. This also makes it easier to expand or upgrade.

XML is currently used in SILO for storing information about service capabilities and dependencies [baldine07_1], in perfSONAR (in the form of RELAX-NG [zurawski06_1]) for the storage and exchange of performance measurements (separating relatively constant metadata, e.g. an identifier for the performance monitor, from rapidly changing measurement information), and ORBIT also uses definitions stored in XML configuration files to initialize and configure measurement points within their framework.

4.5 NDL

The Network Description Language (NDL) is currently investigated to be used for example within the control framework ORCA (Cluster D) to produce a topology knowledge base (TKB) [geni09_4]. The advantage of a TKB is that it provides easily accessible information about the network upon which the management and control planes can build. NDL is a method for representing information about resources and its capabilities [ham08_1]. It provides a common framework for expressing metadata so that it can be exchanged between applications without loss of meaning. Information is expressed using triplets:

- Subject: The resource being described.
- Property: The property the statement describes.
- Object: The value of this property according to the statement.

A set of triplets is called graph and a common textual form to express such a graph is for example RDF/XML, i.e. the graph is encoded in the XML format. This kind of description can be used by applications that need only an overview of the network, e.g. its measurement resources, and not all diagnostic information. This is exactly what is required by the control frameworks.

One related example to physical layer monitoring using NDL is described in [nl09_1]. Here, NDL is used for lightpath monitoring in NetherLight, which is an open optical infrastructure [nl09_2]. To monitor the lightpaths, NDL to specifies their topological details, and actively queries the network elements involved. The output is stored in a network state database with alarm and configuration information. This enables to correlate the configuration data with alarm information and determine whether a specific lightpath is up or down. If a failure is detected somewhere in the lightpath route, this can be clearly indicated using a visualization of the lightpath.

4.6 RSpec

In other control frameworks like ProtoGENI (Cluster C) [proto09_1] or PlanetLab (Cluster B)[peterson05_1], RSpec (resource specification) is the data structure of choice to describe a particular

collection of resources within GENI [geni09_5]. Using RSpecs to transfer information about the topology and its resources can be used in two ways:

- Advertisement Pipeline: Advertisements contain information about the physical nodes. Additional information about components may come from other sources, such as measurement services. Measurement services are independent services which monitor nodes and can provide information which may be relevant to the use of those nodes. This information can be static information, like motherboard chipsets or memory capacity, or it may be transient, currently such as load or network usage; in future implementations, this may also include physical layer measurements. Transient data is taken care of by extensions. Because such a service provides its data as an extended RSpec, gathering measurement data can take the form of annotating an advertisement.
- Request Pipeline: Request RSpecs contain a complete, partial, or empty mapping between the virtual resources a client might desire and the physical resources available on component managers.

The RSpec schema is given in the RelaxNG syntax and thus can be translated to the W3C XML schema, which is widely supported. The schema for the core RSpec is defined using XML Schema version 1.0. Extensions to RSpec are also specified with XML and XML schema, and are included in RSpec documents using the XML namespace functionality [geni09_5].

5 Implementation example

Taking into account the information on potential software measurement frameworks, network management protocols, and data exchange formats for GENI as given in the previous paragraphs, we now want to give an example of how all these concepts could be implemented altogether in order to achieve a unified measurement framework. As illustrated in Figure Figure 5-1, standard network management protocols like SNMP, TL1, and SCPI should be used for accessing individual performance monitoring devices and transmitting measurement information to a hardware implementation of the UMF. For example, this could consist of one or more NetFPGA cards hosted in a server as suggested in [erm09_2]. The NetFPGA pre-processes the measurement information so that measurement data can be stored in databases (e.g. SQL), and accessed and exported to other services/software frameworks (SILO, perfSONAR, …) via XML based protocols. From the UMF, information about the resources and measurement capabilities should be sent to the GENI control frameworks by means of XML based/encoded data structures like NDL or RSpecs.

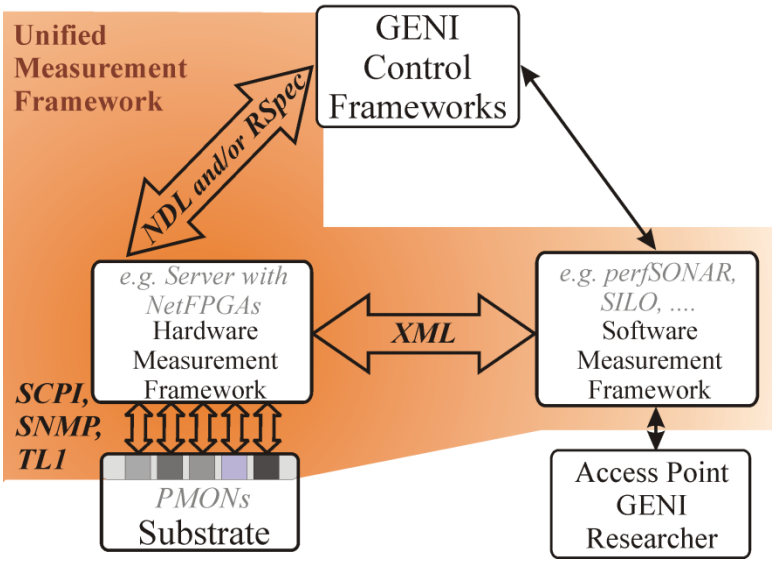

Figure 5-1: Example of UMF implementation concept showing hardware and software measurement frameworks, network management protocols, and exchange data formats and structures.

6 Summary and Conclusions

In this report we analyzed a number of software architectures which are dedicated to network measurements to assess their potential applicability within a unified measurement framework as proposed in [erm09_2]. While GIMS is currently in a very early development stage and ORBIT focuses mainly on wireless nodes, the most promising architectures, SILO and perfSONAR, are already used by some GENI substrates like BEN or MAX. Both ***SILO*** and ***perfSONAR*** also support standard protocols (as described in Section 4) and have proven to be compatible and working with GENI hardware.

Potential network management protocols to exchange measurement and control information between performance monitors and the UMF are SNMP, TL1, and SCPI, with ***SNMP*** being a protocol which is supported by all PMONs currently employed within the GENI substrates [cf. erm09_1].

For the information exchange between the UMF and a control framework either ***NDL*** (with ORCA) or ***RSpecs*** (with PlanetLab and ProtoGENI) has to be employed. Both are ***XML*** based data structures, which guarantees for readability and interoperability between different software and hardware systems.

7 Bibliography

[1] [erm09_1] F. Fidler, C.P. Lai, and K. Bergman, "Technical Note 1: GENI Requirements for Real-Time Measurements (Project Nr. 1631, Milestone 1)" (2009, February) [Online]. Available: http://groups.geni.net/geni/wiki/Embedded%20Real-Time%20Measurements
[2] [erm09_2] C.P. Lai, F. Fidler, and K. Bergman, "Technical Note 2: Specifications and Networking Protocols (Project Nr. 1631, Milestone 2)" (2009, February) [Online]. Available: http://groups.geni.net/geni/wiki/Embedded%20Real-Time%20Measurements
[3] [erm09_3] F. Fidler, "Unified Measurements", 4th GENI Engineering Conference (2009, April) [Online]. Available: http://groups.geni.net/geni/wiki/Embedded%20Real-Time%20Measurements
[4] [silo09_1] Net-Silos Team, "SILO Project – Services, Integration, controL and Optimization for the Future Internet (2009, May) [Online]. Available: http://www.net-silos.net
[5] [perfsonar09_1] perfSONAR (2009, May) [Online]. Available: http:// www.perfsonar.net
[6] [rutgers09_1] Rutgers University, "WINLAB – Wireless Information Network Laboratory" (2009, March) [Online]. Available: http://www.winlab.rutgers.edu/
[7] [stevenson07_1] D. Stevenson, R. Dutta, G. Rouskas, D. Reeves, I. Baldine, "On the Suitability of Composable Services for the Assurable Future Internet ", Proceedings of MILCOM'07, 2007.
[8] [baldine07_1] I. Baldine, M. Vellala, A. Wang, G. Rouskas, R. Dutta, D. Stevenson, "A Unified Software Architecture to Enable Cross-Layer Design in the Future Internet", Proceedings of ICCCN'07, 2007.
[9] [hanemann05_1] A. Hanemann, et al. "PerfSONAR: A Service Oriented Architecture for Multi-domain Network Monitoring", Service Oriented Computing – ICSOC'05, 2005
[10] [boote09_1] J. Boote, "One-way Ping (OWAMP)" (2009, May) [Online]. Available: http://e2epi.internet2.edu/owamp/
[11] [mathis08_1] M. Mathis, J. Heffner, P. O'Neil, and P. Siemsen, "Pathdig: Automated TCP Dagnosis", Proceedings of Passive and Active Measurement Workshop, 2008.
[12] [carlson03_1] R. Carlson, "Developing the Web100 Based Network Diagnostic Tool (NDT)", Proceedings of Passive and Active Measurement Workshop, 2003.
[13] [boote09_2] J. Boote and A. Brown, "Bandwidth Test Controller (BWTCL)" (2009, May) [Online]. Available: http://e2epi.internet2.edu/bwctl/
[14] [matthews00_1] W. Matthews and L. Cottrell, "The PingER Project: Active Internet Performance Monitoring", IEEE Communications Magazine on Network Traffic Measurements and Experiments, May 2000.
[15] [erm08_1] K. Bergman, "Embedding Real-Time Substrate Measurements for Cross-Layer Communication", 3rd GENI Engineering Conference (2008, October) [Online]. Available: http://groups.geni.net/geni/wiki/Embedded%20Real-Time%20Measurements
[16] [ogf09_1] Open Grid Forum, "Network Measurement Working Group"(2009, May) [Online]. Available: http://nmwg.internet2.edu/
[17] [geni09_3] Global Environment for Network Innovations – Wikipedia, "Control, Measurement, and Resource Management Framework for Heterogeneous and Mobile Wireless Testbeds (Project Nr. 1660)" (2009, March) [Online]. Available: http://groups.geni.net/geni/wiki/ORBIT
[18] [sun87_1] Sun Microsystems Inc., "RFC 1014 - XDR: External Data Representation Standards", (2009, March) [Online]. Available: http://www.faqs.org/rfcs/rfc1014.html
[19] [singh05_1] M. Singh, M. Ott, I. Seskar, and P. Kamat, "ORBIT Measurements framework and library (OML): motivations, implementation and features", Testbeds and Research Infrastructures for the Development of Networks and Communities (Tridentcom 2005), pp. 146-152, 2005
[20] [geni09_2] Global Environment for Network Innovations – Wikipedia, "Instrumentation and Measurement for GENI (Project Nr. 1628)" (2009, March) [Online]. Available: http://groups.geni.net/geni/wiki/MeasurementSystem
[21] [barford06_1] P. Barford (Eds), "GENI Instrumentation and Measurement Systems (GIMS) Specifications", GENI Design Document 06-12, December 2006.
[22] [barford09_1] P. Barford, M. Blodgett, M. Crovella, and J. Sommers, "Requirements and Specifications

for the Instrumentation and Measurement Systems for GENI" (2009, May) [Online]. Available: http://groups.geni.net/geni/attachment/wiki/MeasurementSystem/MeasurementSysSpec.pdf
[23] [geni09_1] GENI Project Office, "Spiral 1 substrate catalog", 05. January 2009
[24][case90_1] J.D. Case, M. Fedor, and M.L. Schoffstall "A Simple Networking Management Protocol (SNMP)", RFC 1157, May 1990
[25] [xml09_1] World Wide Web Consortium (W3C), "Extensible Markup Language (XML)", (2009, March) [Online]. Available: http://www.w3.org/XML/
[26][zurawski06_1] J. Zurawski, M. Swany, and D.Gunter "Scalable Framework for Representation and Exchange of Network Measurements", Testbeds and Research Infrastructures for the Development of Networks and Communities, 2006. TRIDENTCOM 2006, pp.417, 2006
[27] [geni09_4] Global Environment for Network Innovations – Wikipedia, "Deploying a Vertically Integrated GENI "Island": A Prototype GENI Control Plane (ORCA) for a Metro-Scale Optical Testbed (BEN) (Project Nr. 1582)" (2009, March) [Online]. Available: http://groups.geni.net/geni/wiki/ORCABEN
[28][ham08_1] J. van der Ham, F. Dijkstra, P. Grosso, R. van der Pol, A. Toonk, and C. de Laat, "A distributed topology information system for optical networks based on the semantic web" Optical Switching and Networking, Vol. 5, pp. 85-93, 2008
[29] [nl09_1] NL Light, "NetherLight NOC", (2009, March) [Online]. Available: http://noc.netherlight.net/
[30] [nl09_2] SurfNet, "NetherLight", (2009, March) [Online]. Available: http://www.surfnet.nl/nl/thema/netherlight/Pages/Default.aspx
[31] [proto09_1] ProtoGENI, "RSpec", (2009, March) [Online]. Available: http://www.protogeni.net/trac/protogeni/wiki/RSpec
[32][peterson05_1] L. Peterson, A. Bavier, M. Fiuczynski, and S. Muir, "Towards a Comprehensive PlanetLab Architecture" Technical Report, PDN–05–030, June 2005
[33] [geni09_5] GENI, "RSpecs", (2009, March) [Online]. Available: http://groups.geni.net/geni/wiki/GeniRspec