



Advanced architecture for INTER-domain quality of service MONitoring, modelling and visualisation



INTERMON-IST-2001-34123

INTERMON Deliverable 4
“Architecture Components and Interactions”

Work-package No. / Title	WP 3 - Deliverable 4: Architecture Components and Interactions
Deliverable Type	Public
Planned Issuing Date	
Distribution	INTERMON Consortium (all INTERMON members)
Document Identifier	Architecture Components and Interactions
Filename	IM-WP3-v109-CINI-D4-ArchitectureComponents.doc
Version	v 1.09
Editor/Author	[CINI] Mauro Gargiulo
Contact Person(s)	Mauro Gargiulo (mauro.gargiulo@napoli.consorzio-cini.it)

Author(s)	Consortium partner	
	[BUTE]	
	[CINI]	M. Gargiulo, R. Canonico, S. D'Antonio, G. Ventre
	[FOKUS]	C. Schmoll, E. Boschi
	[NEC-E]	M. Molina, S. Tartarelli
	[SAG]	
	[SR-ANC]	I. Miloucheva
	[TID]	P. Aranda Gutiérrez
	[TILAB]	F. Saluta, G. Foddis
	[T-NOVA]	A. Kock
	[UniBe]	M. Scheidegger
	[UniD]	J. Seger, S. Michaelis
	[WIT]	P. Malone, M. Ó Foghlú

Change History

V100	first version after Turin Meeting
V101	With further contributions from WIT
V101	Further comments by CINI
V102	Further Contributions from CINI
V103	Contributions from WIT
V104	Editing
V105	Contributions from Fokus
V106	Contributions from SR
V107	Contributions from UniBe about Simulation
V108	Salzburg meeting review
V109	Name changed into Deliverable 4

Table of Contents

1	Architectural components	6
1.1	Global Controller	9
1.2	Local Controller	10
1.3	Tool Manager	11
1.4	Tool wrappers	11
1.5	Data containers	12
1.5.1	Local DB.....	12
1.5.2	Global DB.....	12
1.5.3	Document Repository.....	12
1.6	Advanced Functionalities.....	13
1.6.1	Visual Data Mining.....	13
1.6.2	Simulation	13
1.7	Auxiliary Modules	13
1.7.1	AAA.....	14
1.7.2	Naming Service	14
1.7.3	Data Post Processing.....	14
2	INTERMON components interaction.....	14
2.1	Communication technologies	14
2.1.1	Client - Global Controller communication.....	14
2.1.2	Global Controller – Local Controllers communication	16
2.1.3	Local Controller – Tool Manager communication.....	16
2.1.4	Tool manager – Tools / Data Sources communication.....	16
2.1.5	Global Controllers communication in the inter-domain scenario	17
2.1.6	Database access.....	17
2.2	Service Level Indication.....	17
3	Use cases.....	17
3.1	User – Global Controller interaction.....	18
3.2	Global Controller – Local Controller interaction	19
3.3	Global Controller – Global Controller interaction.....	20
3.4	Tool usage: Metering.....	21
3.5	Local Controller: Data Post Processing	22
4	References	23

List of Figures

Figure 1: The INTERMON architecture: components and modules.....	7
Figure 2: Real databases within the INTERMON general architecture	8
Figure 3: Real tools used within the INTERMON architecture	8
Figure 4: The INTERMON architecture: ISP's federation	9
Figure 5: A possible INTERMON XML service result	15
Figure 6: Client – Global Controller interaction.....	18
Figure 7: Authentication and service request diagrams.....	19
Figure 8: Global Controller – Local Controller interaction	20
Figure 9: Global Controller – Global Controller interaction	21
Figure 10: Metering and data collection	22
Figure 11: Data post processing at Local Controller.....	22

Executive Summary

The aim of the INTERMON project is to enhance the inter-domain QoS and traffic analysis in large-scale, multi-domain Internet infrastructures.

To this purpose, the INTERMON project ([IM-D1], [TA]) 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 fulfilment, in both an intra-domain and an inter-domain scenario. 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 [IM-D5]. However, due to the sensible and costly nature of data stored in the INTERMON databases, the basic INTERMON architecture [IM-D2] has been extended with a set of new components which aim at providing a controlled access to such data, in the form of services.

In particular, the following services will be provided:

1. Monitoring services for QoS and traffic
2. Inter-domain topology discovery
3. Inter-domain modelling and simulation
4. Visual Data Mining with algorithms for traffic matrices and spatio-temporal QoS analysis

The current document, **Deliverable 4**, is an intermediate outcome of work done in Work Package 3 (**WP3**), “**Specification of integrated inter-domain QoS analysis architecture**”. It states the current view of the INTERMON architecture useful for next steps in both the implementation and integration phases. It also reflects latest advances in work done by other Work Packages.

The goal of this Deliverable is to explain and to extend the previous INTERMON architecture (which is included in the new one) in its interfaces, interactions and roles so to make it more scalable, manageable and understandable. The new architecture can actually be considered as a general framework which allows to deploy an ever more complete monitoring activity. Hence, this document is mainly focused on components roles without providing technical details.

This deliverable consists of three chapters and is completed with a list of *References*.

This deliverable is organised in the following sections.

Section 1 describes the INTERMON architecture and its components. Interactions among architecture components are described in Section 2, while Section 3 presents the main use cases.

1 Architectural components

The INTERMON architecture is made of four main architectural components (Figure 1):

1. a **Global Controller** (GC), which is unique within a given Internet Autonomous System (AS) and it is responsible for the INTERMON service provisioning. The GC is responsible to provide monitoring-related services (e.g. SLA monitoring, Modelling, Network planning support, Visual Data Mining, etc.) to the INTERMON users. The proposed architecture is “service-oriented” (i.e. service requests pass through authentication and each system activity will be logged) because monitoring needs resources (i.e. time, cpu-cycles, bandwidth, storage), hence it has a cost.
2. one or more **Local Controllers** (LCs) are associated to each Global Controller. Within an AS each LC is responsible for a specific portion of the network. They select the proper Tool Manager(s) and provide them with the appropriate configuration parameters; furthermore, Local Controllers perform some local data mining to export more compact views of the network status to the Global Controller or to perform some basic data aggregation on the collected data sets;
3. several **Tool Managers** (TMs), each responsible for controlling a given set of tools of a common type (e.g. CMToolkit, BGP-4, IPFIX); in particular, Tool Managers translate a task description (provided by a Local Controller) into a tool specific input and can adapt the tool output to the INTERMON database structure.
4. optional: **Tool Wrappers** (not shown in Figure 1 but close to the interested tool) to add support for (secure) remote control and data collection to an existing tool according to the INTERMON tool-usage protocol (Section 1.4).

A more detailed description of these components is stated in following sections.

The INTERMON architecture present to users a common graphical interface which is located within the Client GUI in Figure 1. Such an interface interacts with the Visual Data Mining (VDM) module in order to implement the data presentation functionality and to extend system interaction capabilities.

The architectural components described above rely on a distributed information base implemented by a two-layer hierarchy of DBMSs. The upper layer of the hierarchy consists in the **Global DB**, which stores a high-level description of the network status (i.e. traffic and QoS) and topology, and is directly controlled by the Global Controller. The bottom layer of the hierarchy consists of one or more **Local DBs**, one per Local Controller. A Local DB contains a detailed view of the part of the network which pertains to the associated Local Controller.

The distinction between Local and Global DB is due to a different information grain and it is useful to address scalability issues. In fact, whereas a Local DB contains fine grained information related to a specific network domain (e.g. an ISP), the Global DB contains coarse grained data resulting from both integration and summarization of data stored into the different Local DBs located within its AS.

We can assume that the internal structures of Global DB and Local DB are very similar as they are supposed to store the same kind of data (also if with a different grain).

The current set and status of high-level tasks in the system and their subtasks are data strictly related to each task/subtask, therefore they have not to be stored into neither the Global DB nor the Local DB which are strictly reserved to monitoring data. Moreover, there is a **Document Repository** which contains only service reports delivered to users. DBs and Repository use different storing techniques: while the former are based on a relational DBMS, the latter is based on a XML database. Finally, the **Visual Data Mining** module allows a graphical interaction with the INTERMON architecture; hence it translates user

interactions with graphical elements of the GUI into new service requests and produces graphical results from the service outputs.

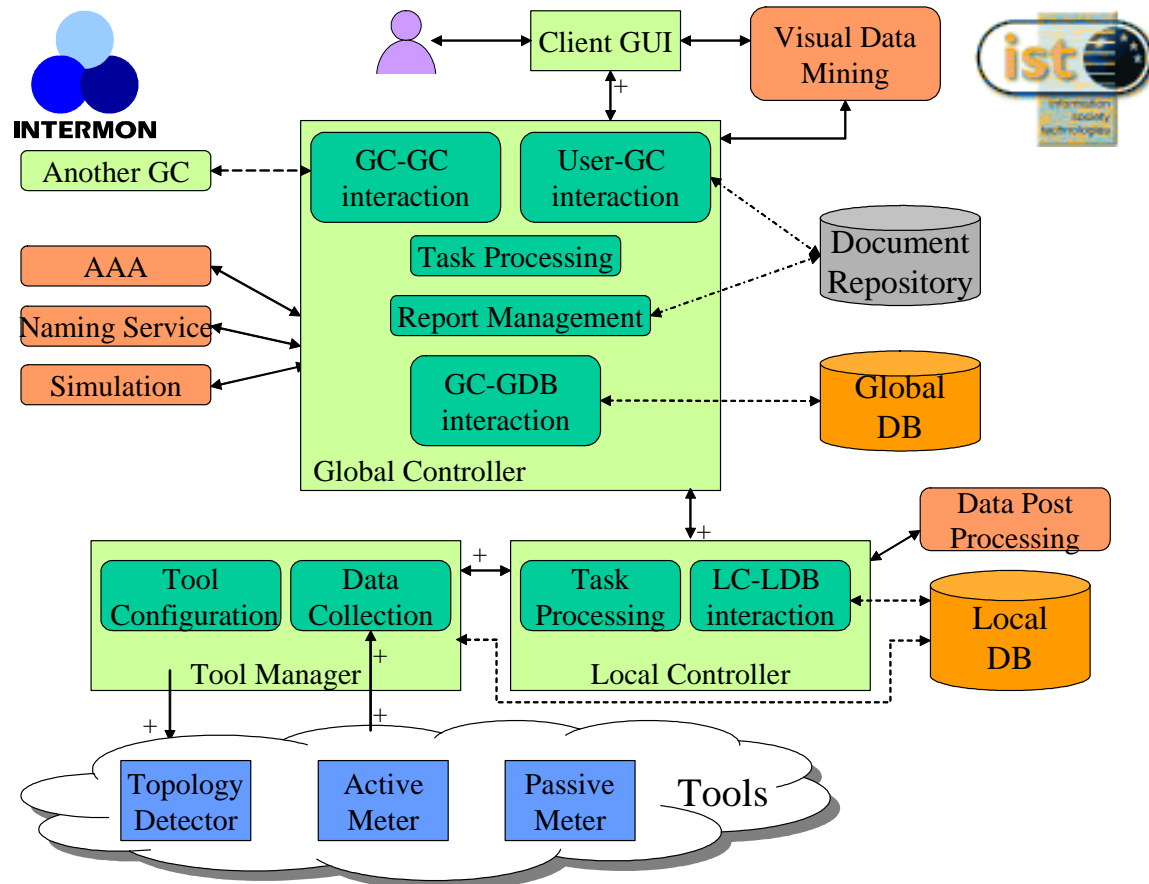


Figure 1: The INTERMON architecture: components and modules

Graphical elements in Figure 1 should be interpreted according to the following colours legend:

- **Light-green:** Components (Client GUI, Global Controller(s), Local Controller(s), Tool Manager(s))
- **Dark-green:** Functionality (User-GC interaction, GC-GC interaction, Task Processing, Report Management, GC-GDB interaction, LC-LDB interaction, Tool Configuration, Data Collection)
- **Orange:** Data Bases (Global DB, Local DB)
- **Pink:** Advanced functionalities (Visual Data Mining, Simulation) and Auxiliary modules (AAA, Naming Service, Data Post Processing)
- **Gray:** Official documents repository (Document Repository)
- **Blue:** Tools and data sources (Topology Detector, Active Meter, Passive Meter)

Figure 1 presents a general description of the INTERMON architecture; therefore, there are some general component which have to be mapped to real ones, according to [IM-D1], [IM-D5] and the other project deliverables, as described in the following:

- Topology Detector: is a tool based upon BGP-4 protocol;
- Active Meter: CM Toolset
- Passive Meter: IPFIX and MonRes tools

- Global DB and Local DB: they are logical entities, to be replaced with a set of real databases (IM-DB for traffic data and CM for QoS).

A real implementation of the INTERMON architecture is depicted in Figure 2 and Figure 3.

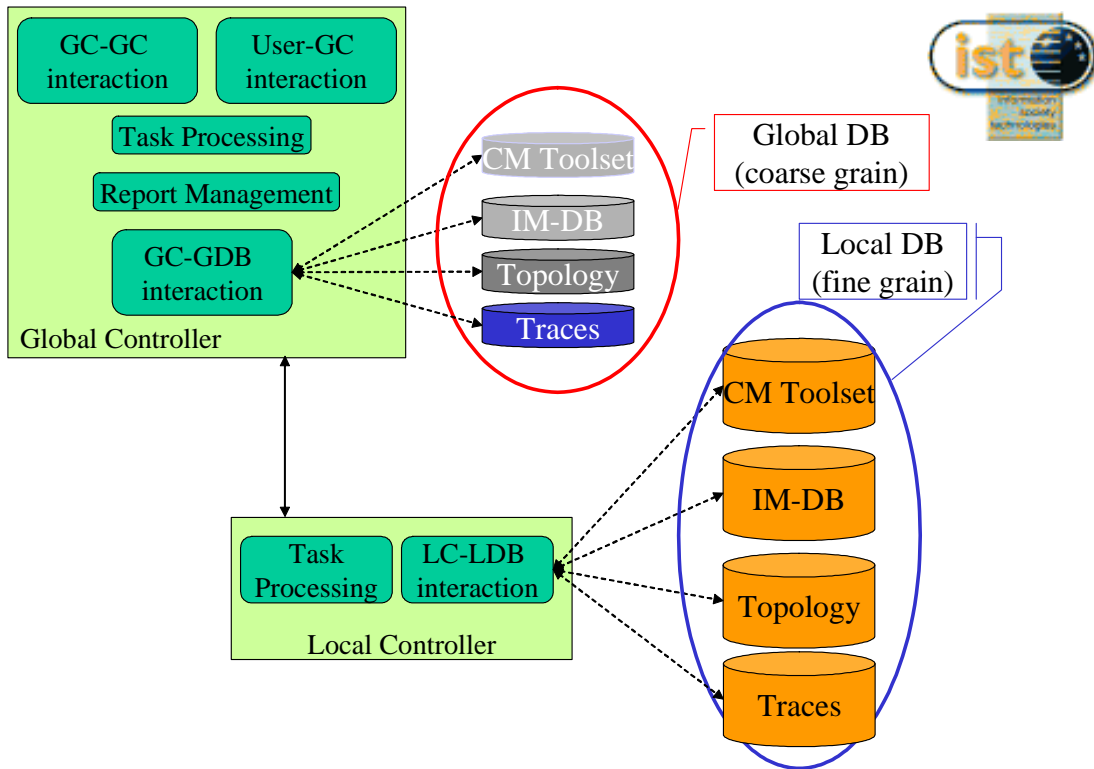


Figure 2: Real databases within the INTERMON general architecture

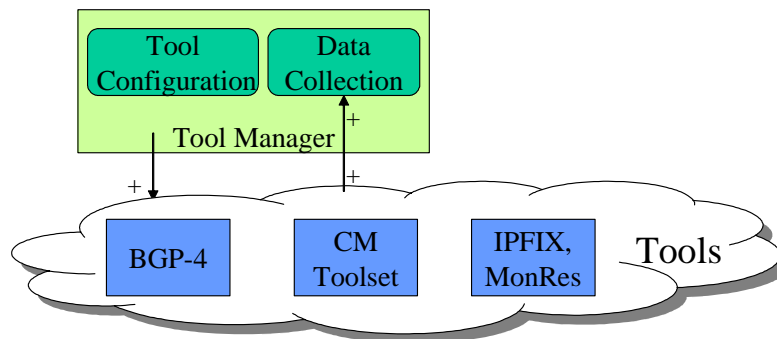


Figure 3: Real tools used within the INTERMON architecture

Figure 4 shows the INTERMON architecture in a real-world scenario, where several ISPs are federated into a single Autonomous System. The Global Controller is responsible for the monitoring activity of the whole AS and it provides such services by cooperating with all of ISP belonging to the AS.

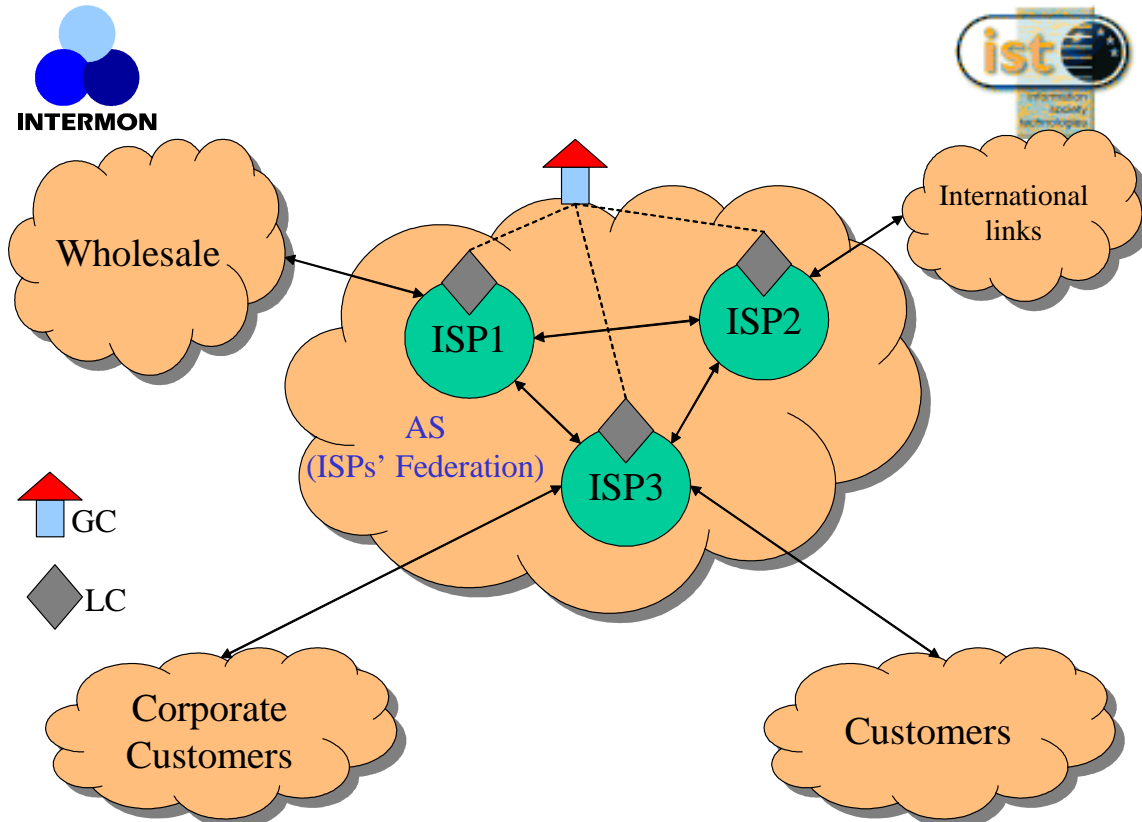


Figure 4: The INTERMON architecture: ISP's federation

On the other hand, in a simpler scenario, it is possible to have an AS consisting of a single entity. In this case, two solutions are possible: with both Global and Local Controller as well as with only the Global Controller. The former is especially suitable to address scalability issues within large domains; the latter is appropriate for small and simple domains.

1.1 Global Controller

The Global Controller (GC) is the central component of the INTERMON architecture. It is (at least logically) unique within a given Autonomous System. Each AS is represented by a Global Controller which provides the monitoring related services. As to the topology issue, each AS represents a single entity with a unique (public) BGP-4 identifier also when composed by the confederation of several ISPs (which will be Autonomous Systems with *private* BGP-4 identifiers). The internal topology of each AS is private and is not supposed to be exchanged among neighbouring Global Controllers.

A service involving several ASs could be provided by exchanging data among the respective GCs through a well defined INTERMON interface (see GC-GC interaction in Section 2.1.5). By doing this, several issues could be addressed, as explained in the following:

- scalability: a number of GC instances can share the overall workload thus leading to better system response time;
- reliability: a big GC, which provides monitoring service for several ASs, represents a single point of failure for a large number of users. If it crashes, all of the ASs lose their GC functionality;

- security: by using a GC for each AS, it is easier to control what information is allowed to be shared with the other ASs;
- a smaller GC is easier to be implemented than a bigger one.

Its role is to accept user requests and to implement the INTERMON services in collaboration with the other INTERMON architecture components. The main functions the GC must implement are the following:

- **User-GC interaction**, which consists of service selection/specification and results provisioning. Such results depend on the service request made by the user and they are provided in a neutral XML format to be parsed, filtered, mapped and translated by the Visual Data Mining module;
- **GC-LC interaction**, which is needed in order to let the GC ask for the required data from lower-layer components (i.e. Local Controllers);
- **GC-GC interaction**, which is needed in order to let the GC collect data from peer components (i.e. other Global Controllers);
- **Task Processing**, which splits an user service request into several synchronized tasks, to be executed either in sequence or in parallel. The execution of each of these tasks may involve other GCs or LCs, and may invoke auxiliary modules made directly available to the GC itself (e.g. AAA, Naming Service, Visual Data Mining, Simulation);
- **Report Management**, which consists in producing and storing service results in the form of XML documents (possibly linked to other tool-specific data files); furthermore, such a function should provide a secure and controlled access to these documents from legitimate users;
- **GC-GDB interaction**, is the functionality allowing the “open” access to a number of different databases. Hence, it is responsible of both adapting data to specific DBs schemas and retrieving information to be forwarded to the VDM module or to another Global Controller.

As Figure 4 shows, the Global Controller interacts with one or more Local Controllers, whose role is detailed below.

1.2 Local Controller

The Local Controller (LC) is the intermediate component of the INTERMON architecture. Within a given Autonomous System one or more Local Controllers may exist. A number of motivations lead to the decision of deploying several LCs into a single domain:

- in networks built as interconnected “islands”, each relying on a different technology (e.g. ATM, MPLS, Diffserv, ...), one LC can be deployed for each “island”;
- in networks spanning over a very wide geographical region, it may be natural to elect an LC for each sub-region (e.g. one for each country of the EC);
- by balancing the entire load over several Local Controllers, better performance can be expected;
- if the same dataset is replicated over several Local Controllers, suitable policies can be implemented to increase the availability of the whole system.

Each Local Controller accesses its own Local DB. The main functions the GC must implement are the following:

- **GC-LC interaction**, which is needed to let the LC accept requests from the Global Controller;

- **Task Processing**, which splits a GC request into several synchronized tasks, to be executed either in sequence or in parallel. The execution of each of these tasks involves one or more Tool Managers, which are responsible of collecting and providing monitoring data to be stored into the Local DB related to the Local Controller;
- **LC-LDB interaction**, is similar to the equivalent functionality within the Global Controller.

Optionally, a **Data Post Processing** external module could be activated by a Local Controller to summarize data collected in the Local DB.

1.3 Tool Manager

Each Local Controller relies on a set of Tool Managers to populate its Local DB with up-to-date data reflecting the current status of the network. Since many kinds of tools are available, assigning to each Tool Manager the responsibility for a set of homogeneous tools seems a natural choice. Hence, at least three different kinds of Tool Managers will be developed:

- one for Passive Meters,
- one for Active Meters, and
- one for interfacing with a suitable topology discovery tool.

A Tool Manager plays four fundamental roles:

1. it communicates with its LC in order to both receive subtasks for the tools it is controlling and to notify the LC about the status of tools and subtasks.
2. it remotely commands the tools for which it is responsible, by means of a proper tool-specific configuration client;
3. it collects data from the tools and adapts their tool-specific format into a new format that is compliant with the INTERMON database structure;
4. it forward collected data to its Local Controller. Optionally, it could store collected data directly into the Local DB through an ODBC-based interface.

1.4 Tool wrappers

The Tool wrappers communicate with the tool configuration part and the data collection part in the Tool Manager. It is useful for tools that do not provide an XML interface compliant with the Tool Manager.

A wrapper can connect to an associated Tool Manager for exchange of both control and monitoring information. Also the Wrapper handles and supervises the running and termination of the tool and accompanying programs (e.g. output filters). The Wrapper scans the output and prepares these information for delivery to a remote Tool Manager.

1.5 Data containers

1.5.1 Local DB

The Local DB provides data storage functionality to the Local Controller which is responsible for its management. It contains information related to a specific network domain (e.g. an ISP).

Actually, the Local DB is composed by several DBs, i.e. CM for QoS and IM-DB for traffic and topology. As a Local DB is supposed to store a number of fine grained data, it is possible to improve system scalability by using several Local DBs within a single AS.

The Local Controller is responsible for the management of the Local DB. It is in charge of storing both data collected by Tool Managers and data resulting from a further processing activity. As said in Section 1.3, optionally, also the Tool Manager could directly access to the Local DB in order to improve system scalability by reducing the Local Controller workload.

1.5.2 Global DB

The Global DB provides data storage functionality to the Global Controller which is responsible for its management. It contains coarse grained data, resulting from both integration and summarization of data stored in the different Local DBs within the AS.

While just one entity of the INTERMON architecture (i.e. the Global Controller) is allowed to interact with its instance of the Global DB (through an ODBC-like interface), several data sources provide data to be stored in it. Such data sources are the following:

- Local Controllers located in the same domain as the GC to which the Global DB belongs: they provide data collected from meters deployed in the part of network they control;
- the Global Controller to which the Global DB belongs, as a result of further processing and/or Simulation activities;
- other Global Controllers which cooperate to provide a monitoring service in an inter-domain scenario.

Finally, data stored in the Global DB is exploited to build the service result which is stored into the Document Repository.

1.5.3 Document Repository

It contains documents to be forwarded to the users as result of the monitoring activity. Such a repository is meant to decouple end user and GlobalDB. It stores all the service results produced as a consequence of user requests and, as such, maintains a history of the user-system interaction. Such a decoupling between users and DB is the most natural complement to the enforcement of the AAA policies within the INTERMON architecture.

The Document Repository should also support simple query activities for document retrieving (e.g., user may wish to request all results which he/she requested in the previous day/week/month, or documents concerning topology discovery between two dates). As it is supposed to store only XML documents and related attributes (i.e. service type, customer, issue data) a XML DBMS seems a better solution than a relational DBMS one.

Information about the associations among documents, users and tasks should be stored into the document repository itself as a set of document attributes.

1.6 Advanced Functionalities

This section explains the Advanced functionalities developed within the INTEMON project.

1.6.1 Visual Data Mining

The visualisation considerations can be viewed as a Publisher/Viewer model [Wood96]. Such a task can be divided in three subtasks (Filtering, Mapping and Rendering) combining to translate raw data (request results) to a more user-friendly output.

How these tasks are distributed is a task for the VDM module and is independent of the Global Controller. While the filtering and mapping of this raw data are usually tasks internal to the VDM module, the rendering task might belong to the VDM module or might be carried out entirely on the client side. This ability to shift the barriers of the Publisher/Viewer model between the VDM module and the client application allows for a more flexible environment in which the user can operate and will provide vastly improved performance in the case where the client application can, for example, make use of specialised hardware to achieve complex image rendering. At the same time this provides the ability for a considerably lighter client to make use of the same system.

1.6.2 Simulation

Measurements of network characteristics can only provide information about the current and past state of a network. However, network management and network planning tasks often require prediction of network behaviour for the near future and what-if analysis. The modelling and simulation toolkit provides the user with tools to performs these tasks.

By relying on measurements stored in the database, or by requesting special measurements, the toolkit can automatically create and parameterize models representing entities of various levels in the network (autonomous systems, aggregate traffic sources or even whole clusters of ASs), considering characteristics like packet loss or delay distributions. These models can then be combined to form simulation scenarios, automatically and manually. Various modelling and simulation approaches are provided by the toolkit, in order to cover a wide range of application areas. All of them have a special focus on scalability, since the scenarios within the INTERMON context are expected to be very large, to the extent where traditional simulation approaches would not suffice.

The user interacts with the modelling and simulation toolkit by sending requests, which are then forwarded by the Global Controller. Resulting parameterized models, simulation scenarios and simulation results can be stored in the database for later use or reuse, by the modelling and simulation toolkit or the visual data mining module.

1.7 Auxiliary Modules

This section briefly describes the main functionalities of the auxiliary modules used by the INTERMON architecture.

1.7.1 AAA

It is useful for verification of the identity of a user or an entity such as a Global Controller. It is also used to check whether or not a given entity has the rights to perform a requested action (authorization). Such an action could be to control a component in the system or to retrieve some information from a database or repository. In addition the AAA component might be used for user profiling. It is used by both the GC-GC and User-GC interactions in order to identify the entity requesting a new service (which leads to the creation of a new document in the repository) or asking for some already available information (stored in a pre-existing document in the repository).

1.7.2 Naming Service

The purpose of a naming service is to provide a binding between names and objects. Clients of naming services use this facility to locate objects by name. Within the context of INTERMON architecture the naming service will enable objects to discover the location of other objects, with which communication is required. For example the Global Controller will locate the AAA object with which it wishes to communicate by requesting a reference to the object location via the naming service. Objects which wish to be discovered by client requests to the naming service need to first register their name and location to that naming service. A naming service must at minimum provide two basic functions, firstly the ability to bind names to objects and secondly the ability to look up objects by name.

1.7.3 Data Post Processing

In order to exploit a specific service request made by the Global Controller, the Local Controller may analyze or summarize information stored within the Local DB by mean of the Data Post Processing module.

2 INTERMON components interaction

2.1 Communication technologies

2.1.1 Client - Global Controller communication

For both user convenience and as a proof of the service fulfilment, the INTERMON *service result* is an XML document, which is stored in a document repository for later access and billing references. In Figure 5 a first attempt of XML document representing an INTERMON service result is reported.

```

<INTERMON_document type="result">
<UID>1924D493-9C5F-47DF-B16D-05A9A62A2217</UID>
<userid>IM000001</userid>
<notify_to>roberto.canonico@unina.it</notify_to>
<notification>required</notification>
<creation_timestamp style=1>
  Tue, 22 Oct 2002 18:14:53 +0200 (MET DST)
</creation_timestamp>
<expiration_timestamp style=1>
  Tue, 29 Oct 2002 18:14:53 +0200 (MET DST)
</expiration_timestamp>
<pgp_signature version="GnuPGv1.2.0">
  iD8DBQE9qxASaUz3f+Zf+XsRAih2AJ938e5hFc85Tvv+TXp4toSAQv1QsQCg/hG5ZMIiUpUe1
  GLq7AOliGR4MbowTMZ
</pgp_signature>
<IM_data>
<VoIP_QOS>
  <time>
    <start>10:00:00.000
    </start>
    <end>10:01:00.000
    </end>
    <steps>120
    </steps>
  </time>
  <step>
    <lost_packets>
      5
    </lost_packets>
    <end_to_end_delay>
      20
    </end_to_end_delay>
    <jitter>
      2
    </jitter>
  </step>
  <step>
    <lost_packets>
      6
    </lost_packets>
    <end_to_end_delay>
      21
    </end_to_end_delay>
    <jitter>
      1
    </jitter>
  </step>
</VoIP_QOS>
</IM_data>
</INTERMON_document>

```

Figure 5: A possible INTERMON XML service result

The client of the Global Controller can be either the Visual Data-Mining module or the end user application. In the former case the Visual Data-Mining module might supervise the entire visual data-mining process. The later case would imply that the application program has the ability to perform all the necessary filtering, mapping and rendering on the data to create the required output. In the case of a thick client, a likely scenario is a hybrid of these two cases; e.g. the visual data-mining module performs the filtering and mapping and passes resulting data to the application, which has a visualisation plug-in to perform the rendering.

When the Global Controller returns a *service result* to the Visual Data-Mining module it is unaware of the boundaries of the publisher/viewer model. This is determined by the particular “client / VDM module” relationship and will most likely be defined in the originating *service request* from the client to the visual data-mining module.

Finally, service output should be made accessible to end users via a proper access key, which may be valid only for a given time.

At this level, the client-GC interaction must be designed by taking into account that INTERMON services may produce their results in two different ways:

- **immediate results**, that can be obtained as a fast reply to the user request (e.g. after a simple SQL query into the Global DB), and hence delivered in the same session of the user request;
- **deferred results**, that require a certain amount of measurement and/or computation activities and therefore should be delivered to the user in an asynchronous way. In such a scenario, upon a service request the system should provide an acceptance notification as soon as possible (i.e. synchronously). Moreover, it should provide other methods to check the current status of a task and some intermediate results later on.

For immediate results, the service output will also be delivered to the client GUI, together with its access key (for later accesses). In case of deferred results, the user will be notified (e.g. via email) about the service completion and will be provided with the document access key.

2.1.2 Global Controller – Local Controllers communication

In the INTERMON scenario, each AS will setup a single Global Controller supported by a number (*federation*) of Local Controllers. Several techniques can be used by the Global Controller to communicate with its Local Controllers.

In particular three possible solutions have arisen, as follows:

- a CORBA-based platform, which implements a virtual bus among intra-domain components;
- a SOAP/HTTPS-based solution as we have to deal with a lot of asynchronously acting tools and a loose coupling between components (e.g. a Tool Manager could choose from a pool of currently available and unloaded Simulations);
- a message-oriented middleware solution such as the OMG COS Notification Service or Sun Microsystems JMS (Java Message Service)

Obviously, different solutions can be chosen within different ASs.

2.1.3 Local Controller – Tool Manager communication

Communication at this level can be implemented with similar techniques as for the previous case.

2.1.4 Tool manager – Tools / Data Sources communication

Since Tool Managers have to interact with several different tools and data sources, the communication logic for their interaction with the tools is tool-specific. It is envisaged that one tool manager will only control tools of a kind. Therefore, the Tool Manager has to be implemented in a variety of specific tool managers, one for each kind of tool used.

2.1.5 Global Controllers communication in the inter-domain scenario

In the case of inter-domain services, a Global Controller needs to gather information from other Global Controllers belonging to peering ASs. This kind of interactions will be performed by exchanging XML documents, whose format needs to be specified by INTERMON. Transport of these XML documents can be performed either by means of the SOAP over HTTP stack or through middleware solutions, such as CORBA or on a message-oriented model. The choice among these alternatives warrants further investigation, which should address scalability, interoperability and security issues.

2.1.6 Database access

The INTERMON architecture relies on traditional DBMS systems to implement both the Global DB and the Local DB, in order to exploit the well-established features of such systems. Therefore, to access these databases, INTERMON components will issue SQL queries via ODBC/JDCS connections.

2.2 Service Level Indication

As explained before, in the INTERMON architecture there are a number of XML-based documents exchanged among the architecture components. In order to avoid misunderstanding a proper naming schema has to be defined for those documents.

By mirroring the SLA/SLS hierarchy, it is possible to introduce the Service Level Indication (SLI) document schema, as described in the following:

- Each Local Controller exports data to its Global Controller by using a **Local SLI** document (such data are not raw data but rather they are the result of the LC Post Processing);
- In a intra-domain scenario, the Global Controller collects one or more Local SLIs (provided by its LCs) in order to define a new document named **Global SLI** (as a result of a further processing) to be stored in the Document Repository;
- In a inter-domain scenario, the Global Controller collects both Local SLIs and Global SLIs (from other GCs) in order to define its Global SLI;
- The Global SLI relies on a proper document template named **Template SLI** which contains information about which methods have to be done and which parameters have to be collected
- Finally, the Template SLI document is also useful in order to define the final document, namely **User SLI**, which is to be forwarded to the user.

3 Use cases

In the following we represent, in the form of UML use-cases, some typical interactions between INTERMON system components.

3.1 User – Global Controller interaction

At the highest level of the INTERMON architecture, the user client interacts with the Global Controller to access the INTERMON services. In Figure 6 we depict a typical interaction between the client application and the Global Controller to access an INTERMON service. As Figure 6 shows, after a preliminary authentication and service selection phase, the Global Controller may need to interact with other Global Controllers (in the inter-domain scenario) and/or with one or more Local Controllers and other high level auxiliary modules to gather sufficient information to satisfy the user request. Once this data is available in the Global DB, the Global Controller collects it (through an SQL query) and produces a new report which is stored in the Document Repository and (in the case of *immediate results*, [Section 2.1.1]) forwarded to the client application (where it can be further processed for visualization).

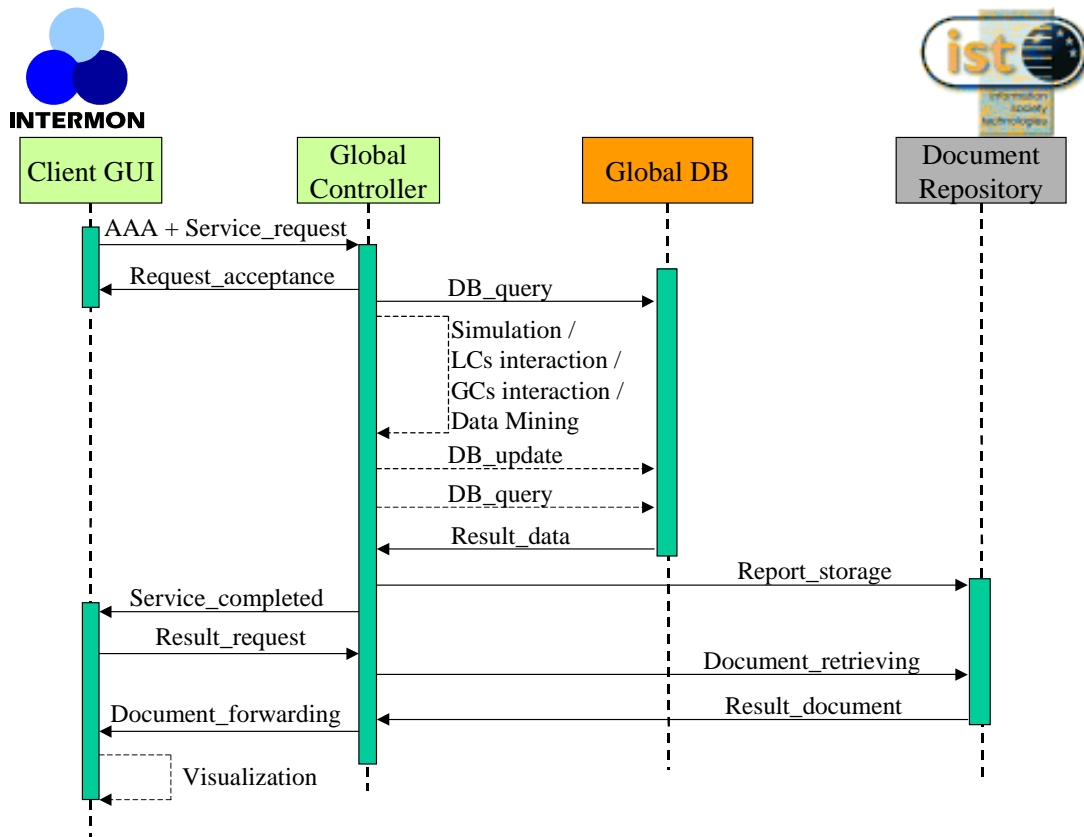


Figure 6: Client – Global Controller interaction

Figure 7 shows a detailed Sequence Diagram related to authentication and service request phases.

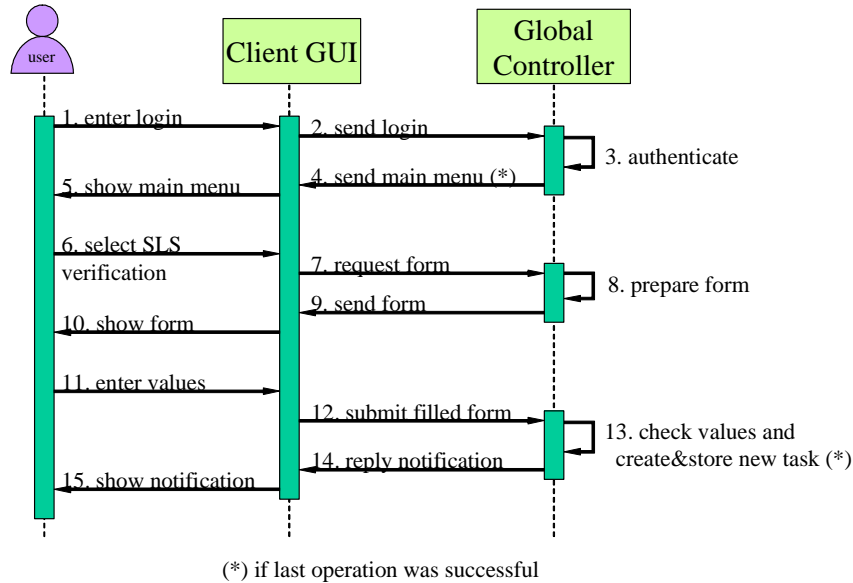


Figure 7: Authentication and service request diagrams

3.2 Global Controller – Local Controller interaction

To gather all the information needed to satisfy an user request, a Global Controller may need to interact with one or more of Local Controllers, which, in turn, may activate one or more Data Sources (through specific Tool Managers) to collect up-to-date information from the network. This kind of interaction is depicted in Figure 8. After that a proper Local Controller has been selected, the Global Controller sends a data request to the Local Controller (for instance, in the form of an XML document). This request is translated into a database query for the Local DB. If the required data is not already available in the Local DB, or it is expired, a proper Tool Manager is selected and a data request is issued. The Tool Manager, in turn, selects which of the available tools is more suitable to satisfy the data request and translates it into a proper tool-specific configuration which is sent to the Data Source. Once data has been collected from the Tool Manager, it is stored in the Local DB and the Local Controller is notified that requested data is available. At this time, the Local Controller may send a new database query for the Local DB and prepare with the received data a result message for the Global Controller.

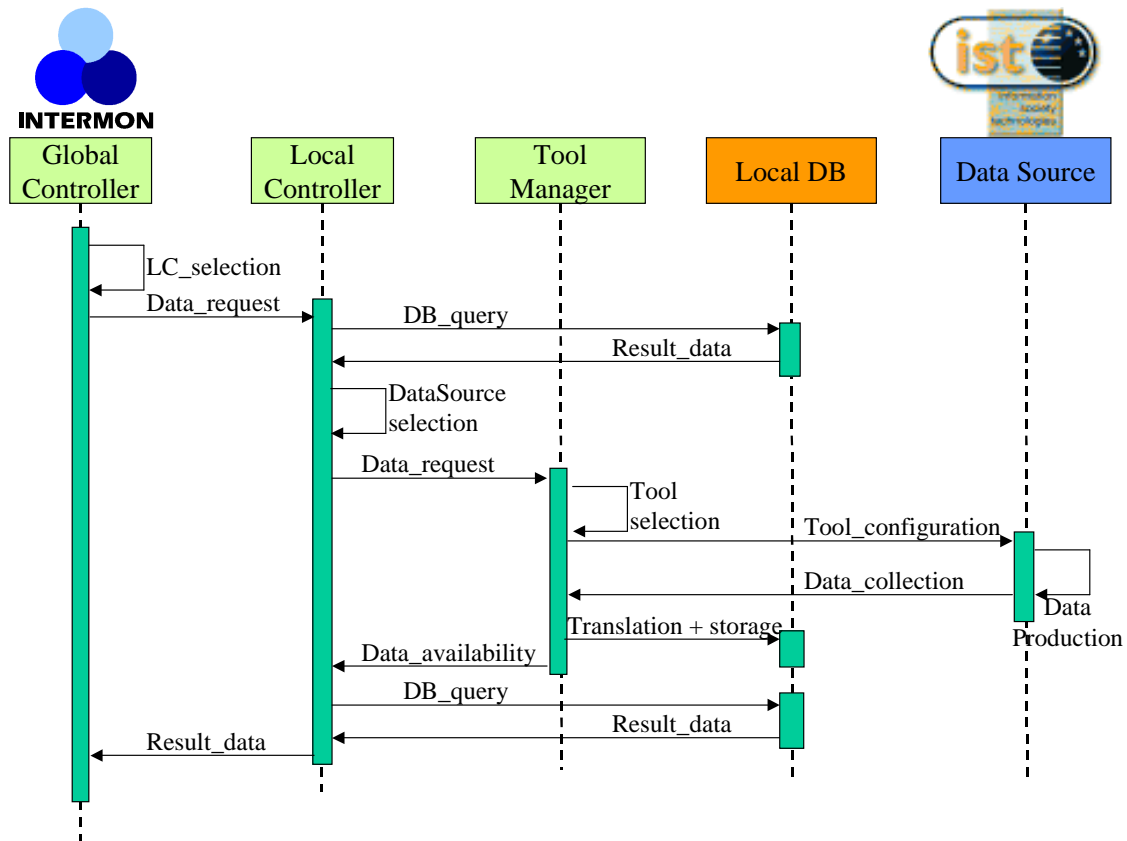


Figure 8: Global Controller – Local Controller interaction

3.3 Global Controller – Global Controller interaction

In some circumstances (e.g. in the inter-domain case), a Global Controller may need to query another Global Controller to satisfy an user request. This kind of interaction is depicted in Figure 9, in which an user request triggers three tasks in parallel: two tasks require data from Local Controllers (exactly in the same way we described in the previous subsection), while the third one involves a different Global Controller. The Task Processing functionality of the Global Controller is responsible of orchestrating and synchronizing the execution of different tasks. In the case illustrated in Figure 9, once the three tasks have been completed, the Global Controller produces the service report and notifies the client application that the service is completed.

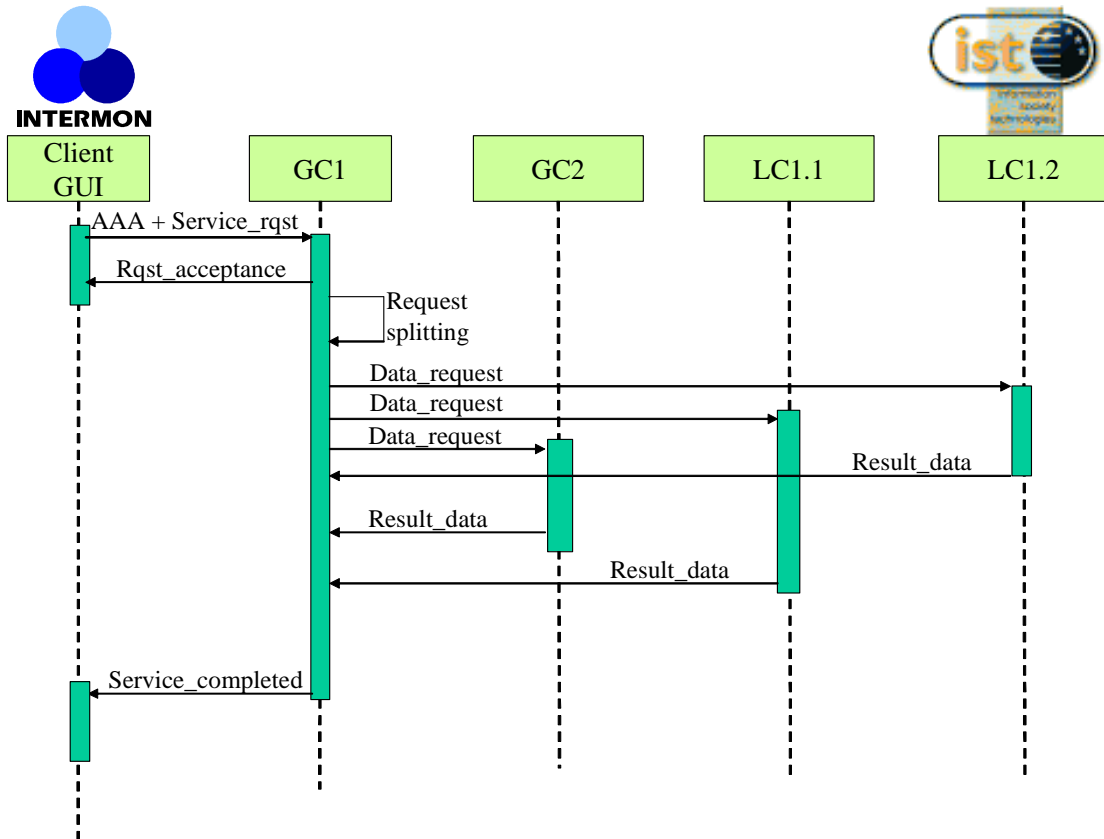
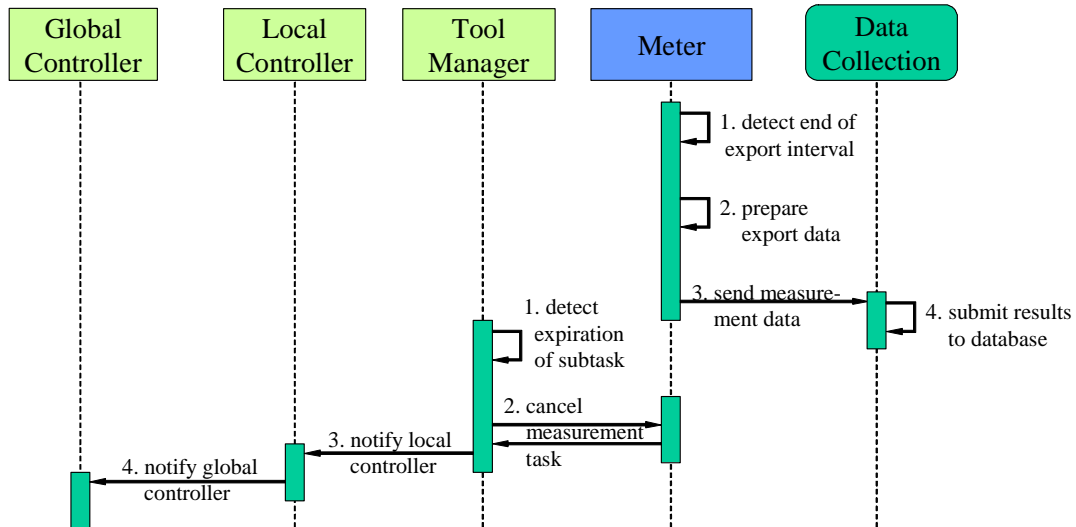


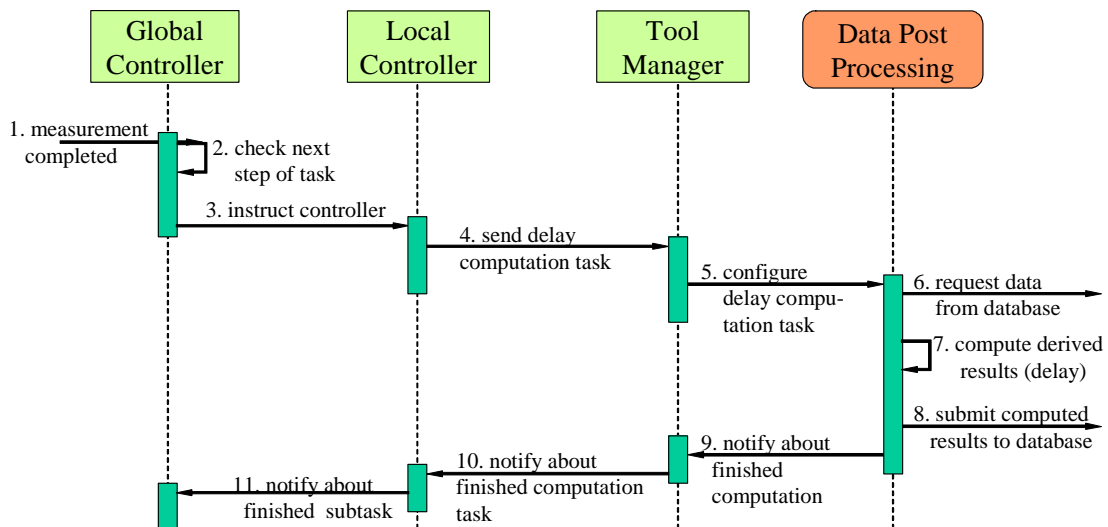
Figure 9: Global Controller – Global Controller interaction

3.4 Tool usage: Metering

Figure 10 depicts the metering and data collection phases.


Figure 10: Metering and data collection

3.5 Local Controller: Data Post Processing


Figure 11: Data post processing at Local Controller

4 References

- [IM-D1] Requirements analysis, INTERMON Deliverable 1, INTERMON Consortia June 2002.
- [IM-D2] INTERMON Deliverable 2, "*Architecture and Specification*", INTERMON Consortia, August 2002.
- [IM-D5] INTERMON Deliverable 5, "*Specification of inter-domain data base with policy-controlled data collection*", INTERMON Consortia, September 2002.
- [TA] INTERMON Project Technical Annex
- [Wood96] Wood, J.; Broodlie, K.; Wright, H.: *Visualisation Over The WorldWideWeb And Its Application to Environmental Data*. Proceedings Visualization'96, IEEE Computer Society Press, Los Alamitos, 1996, P. 81-86.