mPlane

an Intelligent Measurement Plane for Future Network and Application Management

ICT FP7-318627

Use Case Elaboration and Requirements Specification

Author(s):	ETH	B. Trammell (ed.), S. Neuhaus
	POLITO	M. Mellia, A. Finamore, S. Traverso
	FUB	F. Matera
	EURECOM	E. Biersack, A. Barbuzzi
	NEC	S. Niccolini, M. Ahmed, M. Dusi
	NETVISOR	T. Szemethy, B. Szabó
	FTW	P. Casas, A. Bär
	TID	D. Papagiannaki, Y. Grunenberger, I. Leontiadis
	FHA	R. Winter
	ALBLF	Z. Ben-Houidi, G. Carofiglio,
		S. Ghamri-Doudane, D. Perino
	ENST	D. Rossi

Document Number:	D1.1
Revision:	1.0
Revision Date:	31 Jan 2013
Deliverable Type:	RTD
Due Date of Delivery:	31 Jan 2013
Actual Date of Delivery:	31 Jan 2013
Nature of the Deliverable:	(R)eport
Dissemination Level:	Public

Abstract:

This document defines the requirements for the mPlane architecture on the background of a set of scenarios explored by the consortium, a survey of existing comparable measurement systems and platforms and applicable standards therefor, and a set of architectural first principles drawn from the description of work and the consortium's experience.

As mPlane is intended to be a fully flexible measurement platform, freely integrating existing probes and repositories with ones to be developed in the project, this document is primarily concerned with the definition of interfaces among mPlane components. While it does enumerate capabilities to be provided by these components, these are primarily intended to ensure the platform has the flexibility required to meet all the scenarios envisioned; the enumerations of measurements, metrics, data types, and other component capabilities are therefore not to be construed to limit the scope of work on components within the project to just those scenarios treated in this document; nor do the scenarios enumerated here define the capabilities to be demonstrated in the project's integrated trial.

Keywords: architecture, use case, scenario, measurement, platform

Disclaimer

The information, documentation and figures available in this deliverable are written by the mPlane Consortium partners under EC co-financing (project FP7-ICT-318627) and does not necessarily reflect the view of the European Commission.
The information in this document is provided ``as is'', and no guarantee or warranty is given that the information is fit for any particular purpose. The user uses the information at its sole risk and liability.

Contents

Document change record

Version	Date	Author(s)	Description
0.1	20 Nov 2012	B. Trammell (ETH) ed.	initial draft
0.9	28 Jan 2013	B. Trammell (ETH) ed.	final draft for review
1.0	31 Jan 2013	B. Trammell (ETH) ed.	final

1 Introduction

The Internet's diversity provides it with great flexibility and resilience, and has driven rapid innovation at the edge. However, the Internet has shown to be fragile to problems arising from interactions among networks and to misbehaving nodes at the edge. While the architecture of the Internet has long been split between the data plane (which carries packets) and the control plane (which controls routing), the mPlane project advocates the development and deployment of a third plane, the measurement plane. This document is the first step in the specification of the architecture of this measurement plane.

This document defines the requirements for the mPlane architecture on the background of a set of scenarios explored by the consortium, a survey of existing comparable measurement systems and platforms and applicable standards therefor, and a set of architectural first principles drawn from the description of work and the consortium's experience.

Section 2 lists first principles to be considered in defining architectural requirements. Section 3 enumerates and explores measurement scenarios representative of those we intend to address within the scope of the mPlane project. As mPlane is intended to be a fully flexible measurement platform, freely integrating existing probes and repositories with ones to be developed in the project, the development of requirements considered these scenarios as representative of the scenarios the platform may be applied to, in order to ensure the architecture is sufficiently flexible to address them.

These three sections provide the basis of section 5, which enumerates platform requirements, defining the responsibilities of the components and the properties of the interfaces between them. Following the requirements is section 6, which draws from the requirements and provides an initial sketch of the mPlane architecture to be defined in D1.3. This last section is to be taken as initial work, deferring many choices as future work, and may significantly differ from the final architecture specification.

Note that this document is primarily concerned with the definition of interfaces among mPlane components. While it does enumerate capabilities to be provided by these components, these are primarily intended to ensure the platform has the flexibility required to meet all the scenarios envisioned; the enumerations of measurements, metrics, data types, and other component capabilities are therefore not to be construed to limit the scope of work on components within the project to just those scenarios treated in this document; nor do the scenarios enumerated here define the capabilities to be demonstrated in the project's integrated trial.

2 First Principles

In addition to addressing the scenarios in Section 3, the requirements for the mPlane architecture will conform to a set of architectural principles identified by the consortium as important for meeting mPlane's goal to become a universal platform for Internet measurement.

First, we have identified three classes of component: i) probes, which measure; ii) repositories, which correlate, store, and analyze, and iii) a supervisor, which controls and provides learning-assisted iteration. Definitions of components and interfaces should conform to this broad division of responsibility, though it is possible that probe, repository, and supervisor functions be co-located or even supported by the same component.

Second, many of the envisioned scenarios have as a key feature *iterative measurement*: the results of one measurement determine the next measurement to perform, especially in the case of manual or automated root cause analysis applications. This is not to say that all mPlane workflows are strictly iterative; these iterative measurements often take place on the background of *continuous* passive or *periodic* active measurements.

The data and control flow within the architecture must therefore support both an inherently cyclic workflow in the ``foreground''; the management of a large set of continuous or periodic measurement in the ``background'', whether autonomously performed by probes or directed by supervisors; and the integration of these data streams into a coherent picture of the state of the measured network.

Third, the architecture should be completely described by a *small number of simply defined interfaces.* In other words, anything which implements the interface(s) appropriate to an mPlane component of a certain type *is* an instance of that component.

Finally, these interfaces should utilize existing *standards,* and be defined in such a way to allow future extension to these standards. Taken with the interface-oriented approach to architecture, this will allow the project to scale its impact during its three-year lifetime by utilizing, to the maximum extent possible, existing probe and repository technology.

3 Candidate Scenarios

This chapter describes the scenarios evaluated when determining the requirements for the architecture; each section contains the following items:

- A **scenario description** including available background information; what is the goal of this scenario?
- A **scenario narrative**, in the form of a story. How are measurements initiated? Are they continuous or on-demand? Where are the results reported?
- An analysis of the **measurements** to be performed by the probes, with attention paid in each case to the type of measurement performed (i.e. active, passive, or hybrid), the timescales over which they must be performed (i.e. on-demand, periodically?), and the **metrics** produced.
- An analysis of the data that must be **stored at, retrieved from, derived at,** or **made available from external sources** at the repositories, with attention paid to the **information elements** and **schemas** for each.
- An analysis of the **data analysis tasks** necessary to support each scenario.
- An analysis of the role of **iterative measurement** in this scenario, with attention paid to the information required by the supervisor to allow iteration.
- A short description of the **results** produced in the scenario.

These scenarios are primarily examined for their ability to cast light on the operations that must be supported by a measurement platform; they do not represent the use cases that will be handled during the demonstration.

3.1 Supporting DaaS troubleshooting

Running desktop-as-a-service (DaaS) solutions in remote data centers is an emerging means of delivering virtual PCs in an inexpensive, secure, and easy-to-maintain way. The fact that such solutions rely on the presence of connectivity between users and their virtual PCs poses a challenging operational question that mPlane can address: *what is the quality of experience of the user when running a particular application over the thin-client protocol?*

3.1.1 Scenario Description

Thin-client solutions allow users to connect to remote services and access content with a virtual PC as if they had physical access to the remote machine. Thin-client protocols, such as Microsoft Remote Desktop Protocol (RDP) and Citrix High Definition User Experience (HDX), allow graphical displays to be virtualized and served across a network to a client device, while application logic is executed on the server (e.g., editing documents or running multimedia applications). The reasons behind deploying thin-client solutions and hosting virtual PCs in datacenters are related to easiness of maintenance, cost optimizations and security. However, thin-client solutions were firstly designed for Local Area Network (LAN) environments, where bandwidth and delays between users and their virtual PCs are generally not an issue. With the advent of the cloud paradigm, hosting virtual PCs in remote data centers poses challenges in terms of whether or not the network has enough resources to sustain a good quality of experience (QoE) for end users also in Wide Area Network (WAN) environments.

As users perform real-time activities through thin-client protocols (e.g., reading and writing emails, web browsing, and watching flash content), the responsiveness of the network becomes a crucial parameter to determine the QoE perceived by end users when interacting remotely with their virtual PC. Knowing the kind of application users are currently running on their virtual PC would be the first step to shed light on the QoE they perceive given the actual network conditions: the entity of a bottleneck in the network may be tolerable for a user who is using the thin-client solution for reading a document on its virtual PC, whereas it may be unacceptable for a user watching multimedia content. However, several issues make the detection of the application running inside thin-client protocols a challenging task. First, the content over the thin-client connection is normally encrypted and toward the same server port (e.g., port 3389 for RDP) so that no port-based or pattern-matching mechanisms can be exploited to infer the application running in the virtual PC. Second, the data exchanged between the user and the virtual PC are actually the video bitmaps of what appears on the remote screen, and do not follow the request-response scheme of the specific protocol the running application is using. Last, thin-client protocols are closed source, so it is hard to infer how they actually work and whether they perform optimization depending on the content they are transferring.

The goal of this scenario is to exploit mPlane to understand whether the path between the thin client and the remote server has enough resources to sustain the rendering of the specific application. The idea is to use mPlane to collect and correlate information about the application being requested and the available network resources, in order to infer the quality of the service as perceived by the end-user and to pinpoint the reasons of any performance degradation.

3.1.2 Scenario Narrative

In this scenario we focus on the problems related to the QoE as perceived by thin-client users when they connect to remote data centers. Let us assume traveling workers that have to connect to a data center to have access to some documents. One of them initiates a thin-client connection (e.g., by means of the RDP protocol running over port 3889) and opens a document on the remote side. The other one instead accesses to a remote multimedia content. Even if they physically sit next to each other, the two traveling workers can experience different quality in accessing content on the data center, as several factors can affect their QoE, such as the resources available in terms of CPU and memory at either the thin-client or the data-center, the presence of network congestion along the path, the kind of application being accessed on top of the thin-client connection. Without a distributed knowledge of the network status, the user has no way to understand what is causing a bad QoE, other than blame it on the network. However, depending on the content being accessed, the network might have enough resources to actually serve the content in a satisfactory way; conversely, the problem might be due to the load on the data-center side.

The idea is to exploit mPlane probes to passively observe the connection between the user's terminal and the data center, and continuously monitor its state. For a given thin-client connection a probe can collect IP-level features of packets, such as packet size and rate, and classify, e.g., by means of statistical classification techniques, which category of application run on top of the thin-client protocols: the granularity of the application-identification algorithm can vary, from the class of applications it belongs to (e.g., Audio), to the specific application type (e.g., Skype).

Such extracted information (the application category that the user is running inside the thin-client connection) can then be correlated with the network metric that most affects users experience when using thin-client protocols, such as the Round-Trip Time and link utilization, to design a threshold-based system able to infer users' QoE.

The result of the measurements will be a feedback on the QoE currently being perceived by the user: this could trigger alarms that require the mPlane reasoner to react to, for instance by migrating closer to the user the virtual machine where the cloud service runs.

Service providers can make use of such system to continuously monitor users' satisfaction while using thin-client services, thus verifying that Service Level Agreement (SLA) requirements are met.

3.1.3 Metrics and measurements

Finding the root causes of quality degradation in the connection between the thin client user and the remote server requires the presence of at least one passive probe along the path that continuously or on-demand performs the following measurements on the connection: bit-rate and packet-rate at the IP-level, bit-rate and packet-rate at the TCP-level, TCP-payload length (histogram, mean and standard deviation), number of observed packets.

From each flow, probes can consider time epochs of fixed size and carry out the above measurements within epochs. Then, probes classify the epoch by means of statistical classification techniques, to infer which category of application run on top of the thin-client protocols: the outcome can be either a macro-category, such as multimedia content or interactive traffic, or the application type, such as Skype or web browsing.

Concurrently, probes monitor the network metric that most affects users experience when using thin-client protocols, such as Round-Trip Time, available link capacity, link utilization, and correlate

the information to infer users' QoE based on a threshold-based system.

Other possibly important information of interest for this scenario are route information from the user client to the data center hosting the virtual desktop server since this route information can be used to investigate performance degradation (e.g., delay increase) along specific portions of the path. The retrieval of such information can be triggered on demand by carrying out active measurements or queried for existing repositories. An important requirement here is that such information reflect the current state of the network and not an historical one, since the current state is the one currently affecting user performance.

3.1.4 Repository schemas and features

For each address-destination pair (virtual desktop client, virtual desktop server) running on a specific port (e.g., port 3889 for RDP) and for each time window (e.g., 5 or 10 seconds), probes can send the following information to a central repository owned by the ISP: histogram (or full log) of packet sizes of the connection, (optionally) connection rate (can be computed from packet sizes as well) and RTT (and/or available capacity, packet loss rate, etc.) samples (can be computed by analysis of TCP packets even from a passive probe in the middle of a connection).

Additionally, information on path route (e.g., for each address-destination pair, virtual desktop client, virtual desktop server) and RTT accumulated along the path (can be achieved via an active measurement tool such as traceroute or acquired by other provider network tomography tools) can be sent to the central repository.

The ISP may use this data to identify performance degradation problems in some portion of its network and proactively overcome them. In order to locate the problem, the repository should also store the logical position of the probes in the network.

3.1.5 Data analysis tasks

User connection is passively monitored via probes installed along the path, e.g., at the client access network, at the server data center, or in the ISPs offering their connectivity (either continuously or on demand for ``premium'' users in order to verify their SLAs). Data is computed locally on the probes starting from packet sizes (applications currently running are detected by a traffic classification algorithm) and from TCP connection (RTT is estimated): a local decision on whether or not generating an alert about the quality of such a connection is taken.

In this case information is transferred to the control entity that triggers (or retrieves) additional measurements about the route and the accumulated delay along it in order to drill down the network segment generating the impairment on the connection. If necessary, proper countermeasures can be applied, e.g., network administrators can reconfigure the path connecting the thin client and the server hosting the virtual machine, or cloud administrators could move the virtual machine to a data center closer to the user. Notice that defining and applying the proper countermeasures goes beyond the scope of the mPlane architecture, but that their study can assess and validate the usefulness of the mPlane.

3.1.6 Application of iterative measurement

Once the application currently running is known, the QoS requirements are evaluated with respect to passive measurements like RTT, available capacity on the path, packet loss rate, etc. When one of the above observables exceeds a certain threshold, additional measurements are triggered to identify the bottleneck along the path. This information can be then derived on demand via an active measurement tool (like a traceroute) or via network provider repositories (continuously storing information about the network status in various segments of the network). Such sequential measurement can be iterated until the root cause of the problem is identified.

3.1.7 Results

For each connection: classification of the application that a given user is running on her virtual machine(s), estimation of the RTT, bandwidth, loss-rate, associated to such a connection. Based on these, feedback is sent on the QoE currently being perceived by the thin client user, and detection of the causes of possible network impairment.

3.2 Estimating Content and Service Popularity for Network Optimization

Predicting the popularity trends of services and contents (both managed and user-generated) has a wide range of applications.

Indeed, popularity information plays a key role in efforts by operators to improve and optimize traffic/resource management. For instance, this information can be exploited to design ad-hoc caching mechanisms, or used to optimize the placement of caches in their network, or to dynamically adapt request routing. Those mechanisms may help to save limited resources such as bandwidth and storage capacity, increasing network performance and limiting costs for operators, thus optimizing resource allocation within the ISP network.

In addition, popularity information can be exploited to provide services and economic predictions. For instance, this information can be exploited to design and optimize a media advertising service, or to estimate the revenue of a given service/content, or to forecast macro-economic. Those services can represent an additional source of revenues, or help taking market decisions according to the predicted economic trends.

This scenario focuses on the analysis of traffic to estimate popularity at two levels: i) service-level, e.g., VoD, File-sharing, and ii) content-level for a given service, e.g. YouTube videos. Specifically, we investigate how service/content popularity information can be collected across the network, and exploited by an ISP to optimize its network resources.

3.2.1 Scenario Description

Network operators have a built-in incentive to understand how their networks are utilized. As today's Internet traffic is mostly content-oriented, e.g. video traffic and P2P, it is crucial for ISPs to understand how their customers generate and consume content. Furthermore, the pattern of access to content is ever changing due to the growth of Internet traffic generated by mobile devices. Considering these trends, it is no longer enough for operators to place large caches in few locations in the core of their networks. To keep up with the demand of mobile users and with the growth of content-oriented traffic, operators must either spend heavily to increase their bandwidth or push requested content close to users and redirect their requests to their nearby caches accordingly.

This scenario addresses this problem by using network measurements to identify which service or content items will be mostly requested and dynamically adapt caching and request routing to optimize resource allocation. This means: i) estimating the current content or service demand; ii) deploying transparent caches at the network edges; iii) optimizing caching and request routing according to the predicted popularity and cache locations. In addition, ISPs have the opportunity to adapt their IP routing rules by exploiting content popularity and locality information.

Specifically, we aim to investigate predictive caching algorithms that make use of the current content demand in the network to identify future demands and then selectively push them closer to the users (e.g. at the network edge caches). This can be achieved by using predictive algorithms that aim at anticipating the future growth of content popularity. With this knowledge, network operators can adapt their strategies (both at application and network layer) in order to minimize the bandwidth requirements on their core network, and to enhance their services by providing a better quality to their customers. In addition, we aim at quantifying the benefits that transparent caching

can provide to ISPs and users, and how caches should be deployed in the ISP network to provide those benefits. Finally, we are interested in the design of dynamic request routing algorithms that can optimally exploit deployed caches and replicated items.

3.2.2 Scenario Narrative

In this scenario statistics about content/service popularity are continuously collected and updated. Specifically, probes within the ISP network would continuously collect information about the amount of requests generated for every content/service, while the repository continuously update aggregated statistics.

In-network measurements are entirely passive and can be performed at different timescales. The analysis of near-real time request within a short time interval is required to quickly react to popularity changes of content/services. For instance, according to short/medium term popularity information, an ISP running an mPlane intelligent reasoner can take decisions about routing and caching strategies, thus optimizing the overall resource allocation of devices within the ISP network. Measurements aggregated on a longer timescale can for be used for content placement, i.e., to decide where to place content items across the network.

In addition, the reasoner and the repository can process measures to derive long term statistics, as content/service popularity characterization, their evolution over time, and locality information. These statistics can be exploited mostly for network planning, i.e. placement of transparent caches.

3.2.3 Metrics and measurements

Probes collect statistics at two layers:

1. packet-level network measurements, and
2. application layer statistics.

This information is collected for generic content types (e.g. images, videos) or for specific services (e.g. Flickr, YouTube). Measurements are based on passive observation of network traffic.

Packet level probes that are deployed within the ISP infrastructure (PoP for instance) exploit Deep Packet Investigation (DPI) techniques to retrieve information such as the content IDs, either under the form of URLs or under the form of a signature -- for example, a content ID may be computed by hashing selected bytes in the packet payload. This can be then combined with information about time and location to estimate the request rate of a given content/service at a given point in the network.

Application layer statistics will add information such as the size of the content being requested, aggregated ``viewing habits'' of the users, or how persistent is the request for a given service. Probes can passively collect such statistics by observing DNS and CDN traffic, or web browser activities (e.g. through a browser plugin).

The above measurements can be performed continuously, or issued on-demand. The latter case would allow the observation of content and services that are related to exogenous events.

3.2.4 Repository schemas and features

Probes may batch (in time windows) or export without pre-processing their measurements to the repository, thus enabling further analysis and combining content and service requests with other vantage points of the network.

Measurements can be characterized by different timescales and granularity: i) detailed measurements for the short or medium term (hours, days, weeks) and ii) aggregated and refined statistics for the long term (months, years).

In the case of short and medium term measurements, the necessary schema needs to store: i) content/service IDs, ii) the timestamp of the request, iii) the location of the observation and iv) the IP address, properly anonymized, of the hosts generating this request. The vantage point ID can be stored as surrogates for locating the traffic source.

In the case of long term measurements, the schema needs to store: i) content/service popularity characterization (e.g., fitting curve type), ii) the evolution over time (e.g., the parameters of the fitting curve), and iii) the location where the characterization was performed.

Based on the above repository, the reasoner will combine information from different probes and trigger further analysis. Information collected from several vantage points can be aggregated to give a more holistic view of the content popularity within a wider portion of the network.

3.2.5 Data analysis tasks

The user connection is passively monitored by probes placed along the path between the user and the content or the service being requested. Through the use of DPI techniques, probes extract from user traffic the content/service requests (associated to a unique content/service identifier), the timestamps of the requests and their location (network or geographical).

Probes collect information within a given timescale and then export such information to the repository, where a reasoner will apply prediction algorithms and trigger decisions for the ISP network. For instance, the reasoner can issue rules for optimizing routing or caching placements.

In addition, on a longer timescale, the reasoner can derive aggregate statistics about popularity of content and services.

3.2.6 Application of iterative measurement

Whenever users request a content or a service, probes along the path collect the necessary information to estimate the popularity of a given service, and send their measurements and statistics to the repository. Based on such results and on the current network conditions, the reasoner can iteratively decide to instrument the probes to increase or reduce the frequency at which they send reports, and adaptively modify the resolution of the measurements. The iteration of these measurements and their change in resolution will help the reasoner to optimize routing policy and will lead to a better usage of the caches within the network of the ISP.

3.2.7 Results

Based on the collected data, popularity analysis algorithms will derive service and content popularity rankings to select which contents are more suitable to be cached, where and when, and optimize application-level request routing accordingly.

3.3 Enhancing collaboration between ISPs and CDNs

The Internet is a network in which different actors (end users, business users, Content providers, ISPs, and CDNs) interconnect and exchange data between each other. Mainly for historical reasons, none of these actors has full control over the end-to-end path in the Internet. Instead, each of them acts selfishly within its own domain, and has no influence on other actors' choices, even when more collaboration can bring mutual benefits for all the actors. For instance, the end user buys reachability to the entire Internet from its ISP, and has no control over the route its traffic follows towards a given destination. It delegates this task to its access ISP who is, at his turn, only responsible of how the traffic flows inside its own domain, and not on the complete end-to-end path. Similarly, a CDN performs request routing without any collaboration with the ISPs to which it is connected.

This resulted in a situation in which CDN Server allocation and ISP traffic engineering are two uncoordinated traffic engineering tasks. On one hand, for each content demand, the CDN allocates a server according to a specific load balancing strategy and a routing policy scheme, in order to optimize the content delivery process. On the other hand, ISPs also use traffic engineering techniques to adapt to the traffic load evolution and optimize their traffic delivery.

This scenario explores monitoring as an enabler for a smarter collaboration between these actors, for a more efficient content delivery and ISP resource allocation.

3.3.1 Scenario Description

The idea in this scenario is to monitor the ISP network, and extract a set of useful information that could help the CDN to perform an informed routing of the content requests. Different time scales could be considered as far as the extracted and exposed information is concerned. In the simplest case, it could relate to the network load and path conditions between the Content servers and the users. One can also imagine scenarios in which the network proactively signals upcoming events that the content provider or the CDN could take into account in his decisions. An example of such events is upcoming planned maintenance that can impact the quality of certain paths.

Today, content providers or CDNs make their own end-to-end measurements, so they have information about the path conditions between servers and end users. Therefore, it is important that the information that the ISP monitors, infers and provides to the CDN goes a step further beyond what these end-to-end measurements can provide. Consequently, the scenario should focus on information that the ISP can infer, and that the Content provider or the CDN cannot infer by its own means. For instance, the ISP is aware beforehand of planned events and is able to predict which paths it would impact. The ISP also knows beforehand about its traffic engineering policy changes, its network upgrades and new interconnection agreements. Besides, if a failure happens, the ISP should have a better view on the nature of this failure, and whether this failure is only transient or would last longer. All this knowledge could be shared between the ISP and the CDN in order to make the routing decisions more informed.

In addition, the CDN can also provide some statistics to the ISP about content/service popularity, or information about its server selection strategy. This data can be exploited by the ISP for on-line or off-line traffic engineering, in order to optimize its resource allocation. As for the previous case, the ISP can infer such information by means of passive measures. However, the CDN can provide additional information, such as global content/service popularity (also outside a given ISP), or planned release of a new popular service/content, or the placement of a new server.

3.3.2 Scenario Narrative

This scenario aim at creating collaborations between the ISP and the CDNs or the content providers that serve content to its end users. The rationale behind is that both parties have mutual interests from a collaboration: content providers can take advantage of measurement performed by the ISP to provide a better coverage or their service while at the same time the ISP has a better control of how the content provider is using its own network.

The ISP can focus on specific CDNs for example considering at first only the top big players, that is the one responsible for a high fraction of their volume. Probes within the ISP network need to continuously monitor the quality of paths between its interconnection points and its end users. This can be done using both passive and active measurements. The ISP's supervisor is responsible for collecting such information. Such information collected by the supervisor needs to be complemented with information about the network topology as well as information provided by the network administrators such as planned maintenance events, network upgrades and upcoming agreements. The mPlane intelligent reasoner needs to take this information into account in order to predict the possible upcoming changes that might affect path conditions. The information inferred by the reasoner can be either sent in an event-driven fashion (e.g. signal an upcoming change) to the CDN or accessed on-demand by the latter.

Similarly, the CDN collects statistics about content/service popularity. This data is aggregated by the CDN supervisor and coupled with topological information, planned events, and content request and caching strategies. The ISP supervisor can then access part of this data either on-demand or continuously.

The ISP and the CDN can establish then a communication channel through their supervisors and exchange information about the network's state on the base on some predefined policies (e.g., data are normalized, aggregated, etc.).

3.3.3 Metrics and measurements

We summarize the different information that needs to be inferred and monitored as follows.

- The network load. This measure is mostly performed off-line and can be exploited for long-term network planning, traffic engineering, or to establish agreements between ISPs.
- The path conditions between the servers and the users. Note that this information can be derived already by the CP or the CDN. This on-line measurement can be exploited to adapt CDN traffic engineering according to actual network condition.
- The network conditions between the interconnection points facing the CDN and the users. This can be exploited to adapt ISP traffic engineering according to actual network condition.
- Tracking and characterizing network failures and path condition changes.
- The CDN server selection: mapping between content items and servers. ISP can monitor servers selected by the CDN in order to optimize its traffic engineering. If a fine-grained information about the mapping between content items and server can be monitored, ISPs can exploit such information to adapt application-level request routing or inform the CDN about how to optimize its selection policy.

- Information beforehand about planned events like planned maintenance, provisioning, and new interconnection agreements
- Content or service popularity as shown in section 3.2.

3.3.4 Repository schemas and features

Most of the information that needs to be monitored in this use case is real time. This concerns in particular the network load and the path conditions between the interconnection point and the users. This information can be collected directly from the probes, and eventually exported without pre-processing to the repositories. The repositories would therefore need to keep track of: i) the set of metrics that characterize the path conditions, ii) the probe location, and iii) the set of end users for which this path condition applies.

The repositories need to be fed as well with an updated information of the ISP network topology (probes can monitor the IGP protocol) as well as information about future upcoming events that might affect the network conditions.

Finally, the repositories need to keep track of the information necessary for content or service popularity as explained in section 3.2.

3.3.5 Data analysis tasks

Path conditions can be passively or actively collected by probes and exploited locally to take on-line decisions. On a longer timescale, collected data can be sent to aggregation points, possibly pre-processed. The network load is passively measured locally by nodes and collected at aggregation points. Data analysis tasks that are relative to popularity can be performed exactly as described in section 3.2.

3.3.6 Application of iterative measurement

The intelligent reasoner can re-perform measurements to make sure for instance that the predicted changes in the network conditions happened according to what was expected. It can also measure the effect of the change in the CDN policy (upon a recommendation by the ISP) on the new network path conditions.

3.3.7 Results

This scenario should provide the status and a characterization of paths between servers and users in the ISP network. It also proactively detects upcoming changes in these path conditions, and verify that these changes occurred according to what was predicted. The knowledge that is acquired thanks to the measurements is exchanged between the ISP and the Content provider or the CDN so as to enhance the content delivery performance.

3.4 Active measurements for multimedia content delivery

The goal of this scenario is to demonstrate how ISPs can monitor the quality of popular multimedia content (e.g. YouTube videos) delivery in their networks. mPlane technology (probes and the reasoner) provides ISPs the ability not only to continuously monitor delivery performance, but also the means necessary to pinpoint the cause for performance degradations.

3.4.1 Scenario Description

In today's Internet users are increasingly using broadband real-time and VoD services. One of the best example for this is YouTube, even if technically speaking YouTube does not represent a real-time service, since the player can buffer the video stream and usually the stream is not a live feed, but nevertheless the users expect near-zero initial buffering time and constant playback without interruptions to allow the buffering to catch up. The playback of a video with regular resolution (like 360p) requires good quality broadband service, and viewing high definition (720p and 1080p) video (supported since 2009), requires an even better connection.

YouTube and similar services are increasingly regarded essential (some governments even using them for official communication), thus users, ISPs and content providers want to know how the provided service lives up to the expectations. Providing seamless video streaming is especially challenging, but at the same time we must not forget about the Web portal service itself (through which the streaming is made available to users). In this scenario this encompasses loading the YouTube page, searching for videos, browsing among the comments, etc.

3.4.2 Scenario narrative

A group of friends share the URLs for YouTube clips. For some of them, playout quality degrades from time to time, and for some clips are not accessible at all sometimes, thus they notify their ISP(s) about the problem.

Performance degradation can be caused by a number of reasons, and pinpointing it is a matter of luck today. The problem might be in the access network, the core network, at the YouTube streaming and Web servers or the reason even might be a problem on end-user's computer. In this use case we focus on active measurements running on probes placed at crucial points in the network. With the help of these probes ISPs will be able to get an overall view of the services delivered by their network on one hand, and on the other hand they will be able to aid their root cause analysis processes by dynamically reconfiguring the tests running on the probes. The measurements will be coordinated by the ISP's supervisor in both cases (after the initial, manual trigger from customer service). In the first ("data collection") case they will run periodically, while for the root cause analysis case they will be activated as on-demand iterative measurements. In both cases the probes will receive control flows telling them to start data collection for a specified set of sub-services with the specified collection parameters, like video buffer time, etc. Measurement results are uploaded to Repository component(s), evaluated by the Reasoner, and the end results are presented to the ISP's engineers who will take the necessary steps to remedy the problem (e.g. increase trunk or CDN bandwidths, notify CDN operators about mis-configured delivery topology etc.)

3.4.3 Metrics and measurements

As mentioned above, the service should be decomposed into several sub-services and mPlane should define metrics for each of them. Since in mPlane there might be different probes with different capabilities, some of the measurements might be applicable to all probes, while some of them might be only be applicable to a part of them. The sub-services that we define for YouTube are the following:

- **Base service**: loading the starting Web page, including all of its components, like image files, CSS definitions, etc. Since the monitored Web portal is obviously outside of the monitoring authority, special mechanisms have to be applied to deal with changes on the portal (e.g. different page layout).
- **Video searching**: searching for videos using different search criteria. Since videos are constantly added to the system, this test should only determine whether the search returned or not and it should not look for a particular video in the result set.
- **Interactivity**: Providing feedback, posting comments etc. are important portal services - checking these automatically is challenging if probes are not allowed to post themselves.
- **Video streaming**: we are going to define metrics for the characterization of multimedia stream transport over TCP (HTTP). mPlane primarily collects network-related measurements (e.g. bitrate, losses, jitters, no signal periods etc.), and infers QoE metrics from those, depending on codec characteristics. In cases where the probe does have the CPU resources and ability to execute the codec, finer-level analysis is possible, but this cannot be always guaranteed.

Only active probes are utilized by this scenario: they will be measurement modules initiating Web transactions themselves (mostly running on dedicated probes) and collect availability and total response time measurements for the different sub-services. For video streaming quality, more detailed measurements are going to be specified. The probes will have predefined buffering characteristics: how much video should be available before playout begins and how much buffer is needed for continuous playback. Many CDNs use sophisticated, multi-level cache deployments to accelerate content delivery. Taking advantage of mPlane's distributed architecture, measurements from multiple probes could be later used by the Reasoner to discover CDN cache topology and identify bottlenecks. Thus, apart from content-specific measurements (e.g. stream quality as delivered), probes are also going to collect information regarding the delivery architecture (servers participating in the delivery, number of HTTP redirects etc.).

3.4.4 Repository schemas and features

The measurement repository will contain an object schema reflecting service decomposition and relationships between them (all other sub-services depend on the base service). Each object in the schema will store their relevant metrics. Since the mPlane system will consist of many probes, the repository should also store information regarding the logical position of the probes in the network topology, so that the reasoner can perform network path and topology analysis.

3.4.5 Data analysis tasks

The raw data collected by the probes will be pre-processed locally and then collected by the respective Repository. Repositories perform basic aggregation functions (such as normalization, finding best/worst/avg values as well as baselining) and the processed data is analyzed by the Reasoner. The Reasoner will be able to detect anomalies in the results (deviations from normal behaviors in continuous measurements), and initiate on-demand measurements when needed (e.g. during peak hours or at locations sharing the same trunk line). The Reasoner may also re-configure continuous measurements to better focus on relevant metrics (e.g. occurrence of playout stuttering, active bandwidth probing) in order to use probe/network resources more efficiently. Knowing the probes' (physical and logical) locations, the network topology, the measurements sequences and their results, the Reasoner will be able to narrow down the problem location.

3.4.6 Application of iterative measurement

The probes deployed in the network are going to be capable of constant monitoring of the service, allowing generic statistical behavior profiling. On top of this the reasoner will be able to initiate new iterative measurements when analyzing the results coming from a probe. Since the reasoner knows about the logical locations of the probes in the network, it can decide to start measuring the same video stream with the same or with different resolution settings if a probe detects some performance degradation, and by doing so helping to determine the root cause of the problem.

3.4.7 Results

The results will be an overall performance indication of the ISP's network capability to deliver multimedia content, as well a problem pinpointing mechanism to indicate places in the network where bottlenecks are degrading this performance.

3.5 Quality of Experience for web browsing

The goal of this scenario is to demonstrate how mPlane can be used to monitor and find the root causes of end-user quality of experience degradation in Web browsing. This section shows how a browser plugin, network probes, and the Reasoner can be used not just to measure the quality of experience but also to pinpointing the cause of performance degradation.

3.5.1 Scenario Description

One common way to search and access information available in the Internet is via a Web browser. When clicking on a Web page, the user expects that the page gets rendered quickly, otherwise he will loose interest and may abort the page load. The causes for a Webpage to load slowly are multiple and not easy to comprehend for an end-user.

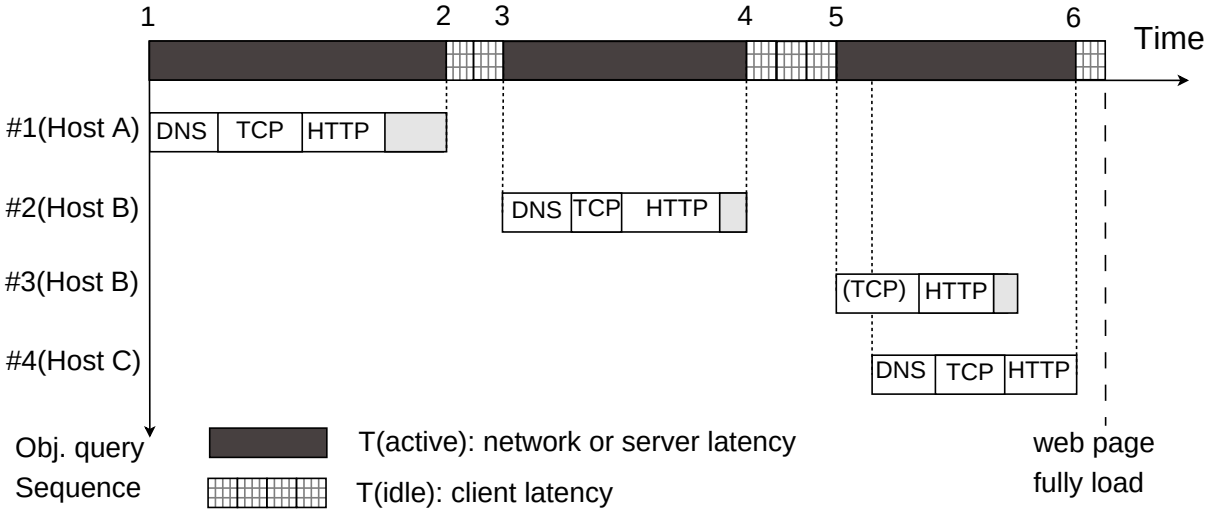


Figure 3.1: Timeline of object downloads that are part of a single Web page

Fig. 3.1 illustrates the underlying behavior when browsing page: the main object usually comes first. After that, the web browser can parse the page structure and load all the objects refereed to in the web page. In order to reduce download times, parallel connections can be also used. After the page content is completely downloaded and rendered, the ``load'' event is fired by the browser and the status of the Web pages becomes **fully loaded**. For each object the normal sequence of events is as follows: (DNS): the URL needs to be resolved using DNS; (TCP) then the TCP connection is established between the client and the server; (HTTP) now the HTTP request can be sent, which will be followed by the transfer of the object from the server to the client where it will finally be[1].

3.5.2 Scenario Narrative

When a user browses the Web he may experience a poor rendering performance or the Web page may not be rendered at all. In this case, the user may want to trigger a root cause analysis in order

[1] Although modern web browsers can trigger other objects download even after the page status is fully loaded, here for simplicity we do not consider those cases and focus on the ones that have occurred before page fully loaded.

to find about the reasons for the poor performance. In this case a plugin will be installed in his browser and then the user will be asked to access the same Web page once more. The plugin will record all the metrics shown in Fig. 3.1, perform the root cause analysis and return the result in a new browser window.

An ISP or a telecom regulation agency which wants to continuously supervise the QoE of Web access could also use the root cause analysis tool. In this case the tool would run on one)or more) dedicated machine(s) and actively browse in a periodic manner a set of predefined Web pages.

3.5.3 Measurements

In its simplest form, the root cause analysis uses only on passive and on demand measurements done by the browser plugin. However for a more full fledged and deeper analysis additional information may be required: examples are a traceroute from the client machine to the server in order to determine the end-to-end path, or a bottleneck bandwidth measurement.

As we see in Fig. 3.1, we need to measure page load times and its various factors is carried out in the browser using a plugin. This measurement is purely *passive*. However, as we will see below, during the diagnosis phase, additional active measurements can greatly help in identifying the causes of poor performance. The minimal set of metrics is the ones in Fig. 3.1; additional ones depend on the nature of the active measurements.

3.5.4 Repository

The measured data will be transmitted to a central repository owned and managed by the ISP. These data are of interest not only for the client who experienced poor performance but also to other stakeholders who may mine these data to develop a kind of ``early warning'' system for performance problems. The ISP may use this data to identify recurring performance problems of its network and take, if possible, corrective actions (e.g. adding more network bandwidth internally or at peering links) or upgrade its DNS capacity. The company hosting the Webpage may be interested in knowing that some clients have experienced poor download performance of its Webpages.

The measurement repository will contain an object schema reflecting service decomposition and relationships between them (all other sub-services depend on the base service). Each object in the schema will store their relevant metrics. Since the mPlane system will consist of many probes, the repository should also store information regarding the logical position of the probes in the network topology, so that the reasoner can perform network path and topology analysis.

3.5.5 Data analysis Tasks

The different measurement results will be submitted to simple arithmetic operations and the results compared against different threshold values; these checks are implemented as a custom made decision procedure.

One possibility is to ``combine'' the measurement results of several clients, e.g. of all the web clients of the different devices in the same home, in order to improve the potential of identifying more precisely the cause of the performance impairment. For instance if the WiFi at home is overloaded and some of the end systems access the Internet via WiFi while others are connected to the home

gateway via Ethernet, the use of multiple devices should allow to distinguish between congestion of the WiFi link as compared to congestion of the access link of the ISP.

3.5.6 Application of iterative measurement

A number of components are involved in generating transmitting and rendering the content of a Web pages. These components are: i) the PC of the client, ii) the local access link, iii) the remaining part of the Internet, and iv) the servers. A slowdown at any of these components will affect the page load time. The goal is to identify which of these components bears the major responsibility for the slow web page load. To carry out this task, using additional information can be very useful.

The tool could request access to either i) a ``reference'' Web page for which it is known that the Web server is close to the end client or ii) a Web page that provides typically very low access times such as the Web pages of major Web search sites. In both cases this could allow to decide if the ``bottleneck'' is near the client or deeper in the Internet or even at the server side.

3.5.7 Results

The primary results will be individual Web page load times with possibly some breakdown concerning metrics such as DNS look-up, TCP handshake times and HTTP request times.

3.6 Mobile network performance issue cause analysis

The main objective of this scenario is to demonstrate how mPlane can be used to monitor and discover problems with end-user quality of experience of mobile connectivity. More specifically, we will explore how a combination of probes that monitor various part of the network can be used to identify the key causes of quality degradation in mobile video streaming.

3.6.1 Scenario Description

Mobile data usage is increasing rapidly in part due to a growing release of mobile devices and the availability of a wide variety of mobile phone applications. For instance, a large number of users nowadays use their mobile devices to watch video on demand (e.g., YouTube) while on the move. However, the resulting mobile data traffic explosion is inducing significant strain on mobile operators as it often results in degraded performance. Finding the origins of these problems is not a trivial task as there is a wide variety of possible causes that may range from poor signal conditions to issues related with the end-service provider (e.g., the CDN).

3.6.2 Scenario Narrative

In this scenario we will focus on the possible problems that are related to experiencing video-on-demand on mobile devices. Lets assume that a user receives a notification from a friend via a popular social network to watch a viral video. The user clicks on the link, the dedicated video application is launched and the video starts buffering. After some, long, initial wait, the video starts but pauses frequently due to insufficient network performance. As the user has no way to find what is causing this issue it is likely that a complain might be filled (or even posted online) concerning the perceived poor quality of the mobile network provider. However, the causes of this problem are often rather complex and, therefore, measurements in various locations might be required in order to locate their origin:

- **Phone issues:** The user's phone might not be able to correctly load and display the video for various reasons: inadequate CPU or memory, missing codecs, lack of caching mechanisms, poorly configured drivers, lack of hardware acceleration or even the wrong video quality was selected by the application.
- **Signal issues:** The user might be in an area with poor cellular reception (low SNR, with only limited physical modulations usable) or where no high-speed data connectivity is available (lack of HSPA support, limited physical bitrate available on the radio channel).
- **Channel congestion issues:** In some cases (e.g., football games, concerts) there might be a very high demand at a given geographic area that may affect a number of local users.
- **Radio network controller issues:** Incorrect settings at the RNC can result in long delays to connect to the network or alter the latency significantly.
- **Mobile ISP backbone congestion issues:** The mobile operator might be running low in backbone capacity (e.g., when microwave links or old wired technologies are used) resulting in lack of capacity to carry data until the base station.

- **Core network issues:** The issue might be generated by a congested or badly configured core network (routing issues), lack of peering points between the provider and the video hosting service.

- **Host service issues:** This includes issues with the content distribution network or the servers that support the service (e.g., congested video servers).

3.6.3 Metrics and measurements

As a combination of problems can result in poor user experience, probes in various locations of the network are required to monitor the service and identify possible issues:

Application probe: provides on-demand information as perceived from the application point of view. The metrics include the buffer status, bitrate, packet loss, jitter, codec characteristics, application logs, etc. These metrics are generated either by instrumented applications on the phone or by using open-source platforms such as FireFoxOS (that enable full instrumentation of the video player and its underlying APIs).

Mobile OS probe: offers on-demand information considering the device capabilities and the device status (CPU, Memory). Once again, an open-source OS may be used (e.g., FireFoxOS or Android) so as to enable access to the device status. Furthermore, an important aspect of this probe is to measure the cellular network conditions (associated cell tower, signal strength, bit errors, transmit characteristics, power state of the device, etc). The amount of information that can be collected is bounded by the data that can be extracted from the cellular modem driver (i.e., Samsung Galaxy S2 devices support the extraction of such information).

Mobile ISP probe: captures, both passively and actively, information about each terminal (e.g., traffic statistics, type of connection, SNR). Furthermore, probes at each basestation passively capture aggregated information such as number of associated devices, overall traffic, channel utilization, scheduling/QoS policies, etc. The capabilities of this probe will be assessed using a cellular test facility that provides a cellular-like infrastructure (at a lower scale). Finally, information about the ISP's backbone such as capacity, transfer rates and latency can be measured using acceleration proxies and middle boxes installed in the network.

Core Network probe: measures, similarly to the ISP's backbone, the core network performance and the peering utilization. This probe can tailored depending on the amount of information that can be extracted from routers deployed in the core network.

Service probe: passively collects information such as service uptime, number of served clients, network congestion, CDN selection at the service provider. Furthermore, the CDN provider can also measure the caching hit-rates, service latencies and report any issues at the distribution network.

3.6.4 Repository schemas and features

Data repositories are distributed across the different network domains (mobile network, ISP, peering, video hosting) with respect to the security and anonymity of the users and in order to protect trade secrets of the mobile ISPs and all parties involved in the service delivery. A solution to this is to only provide high-level aggregated data or even just binary results (problem, no-problem detected).

3.6.5 Application of iterative measurement

When a problem is detected a number of hierarchical steps are taken. Firstly, inquire with the mobile application about the nature of the issue and identify any issues within the user's OS or the connection (signal, type of modulation). If the mobile application/phone side is not the cause of the issue the probes within the mobile ISP can identify any congestion at the user's basestation or within the backbone. Afterwards, the core network and the service provider can be polled to identify any routing or peering issues. Finally, probes within the service provider will be used to check that the resource is accessible (e.g., the CDN network). Notice that in order to accomplish accurate measurements other mobile devices in the vicinity of the user might be also probed to measure the local network conditions. Due to the fact that service guarantee is fundamentally unbalanced (customers have signed contracts with the mobile ISP but not with transit/CDN providers), part of the measurements or diagnosis can be delegated to other entities on behalf of the end users.

3.6.6 Results

The results will possibly provide the mobile network operators with a detailed outlook concerning the performance of their network. For instance, they could get detailed measurements about their backbone capabilities (e.g., latency, used bandwidth), signal coverage for each geographic location as reported by the users' mobile probes and the peering capacity with the service providers. Furthermore, as data collected by mPlane involve multiple parties, this can also allow the end-users and mobile operators to identify the exact causes of potential problems that may affect the performance of their service.

3.7 Anomaly detection and root cause analysis in large-scale networks

3.7.1 Scenario Description

Internet services and technologies are in continuous evolution. New applications and communication paradigms are constantly developed and deployed in the network impacting on the traffic carried by the network. CDN and other cloud-based services handle clients requests using complex policies. Tracking changes in traffic patterns is fundamental for ISPs, network administrators as well as for CDN providers, to understand how Internet services use the network (e.g., at specific time of the day there is a traffic shift due to load balancing), to characterize the behavior of their users (e.g., due to low performance users stop to use a service), and to optimize or troubleshoot their systems when the detected changes disrupt the normal operation QoS of their network.

In this scenario anomaly detection plays a key role to identify relevant changes in traffic patterns and other specific metrics. This analysis is not limited merely to the identification of such variations but the system has to provide root cause analysis capabilities to identify their origins.

Some examples of the changes we want to detect and diagnose include changes in RTT to specific content, changes in the average download speed, service unavailability, changes in the traffic volume distribution, etc.

In general, any temporal and/or spatial changes in any metric should be detected and trigger further analysis to identify the root causes.

3.7.2 Scenario Narrative

In this scenario, the mPlane is continuously collecting predefined measurements over time and from different vantage points. Both active and passive measurements can be considered. Measurements are stored in the repository where, at predefined time scales, are processed to detect the aforementioned changes. Once an anomaly is detected, alarms are raised and could eventually automatically trigger the intelligent reasoner analysis and the corresponding iterative measurements. When an anomaly is detected, the corresponding alarms are logged and presented to the mPlane user, and the most plausible causes for that anomaly are shown.

3.7.3 Workflow

Fig. 3.2 reports a scheme of the data analysis tasks needed to perform traffic anomaly detection and root cause analysis. We can identify 5 different key aspects for the system design:

Data sources: input data are obtained from user devices (e.g., PCs, smart phones, etc.), home devices (e.g. Wi-Fi access point, set top boxes, etc.) and network devices (e.g., probes deployed to monitor ADSL residential customers, routers, etc.). All these devices correspond to *probes* and run a software to monitor some metrics of interest. In particular, this software handles different modules for different metrics. In general, data sources expose measurements for some specific scenario. External data repositories can also be considered as valid

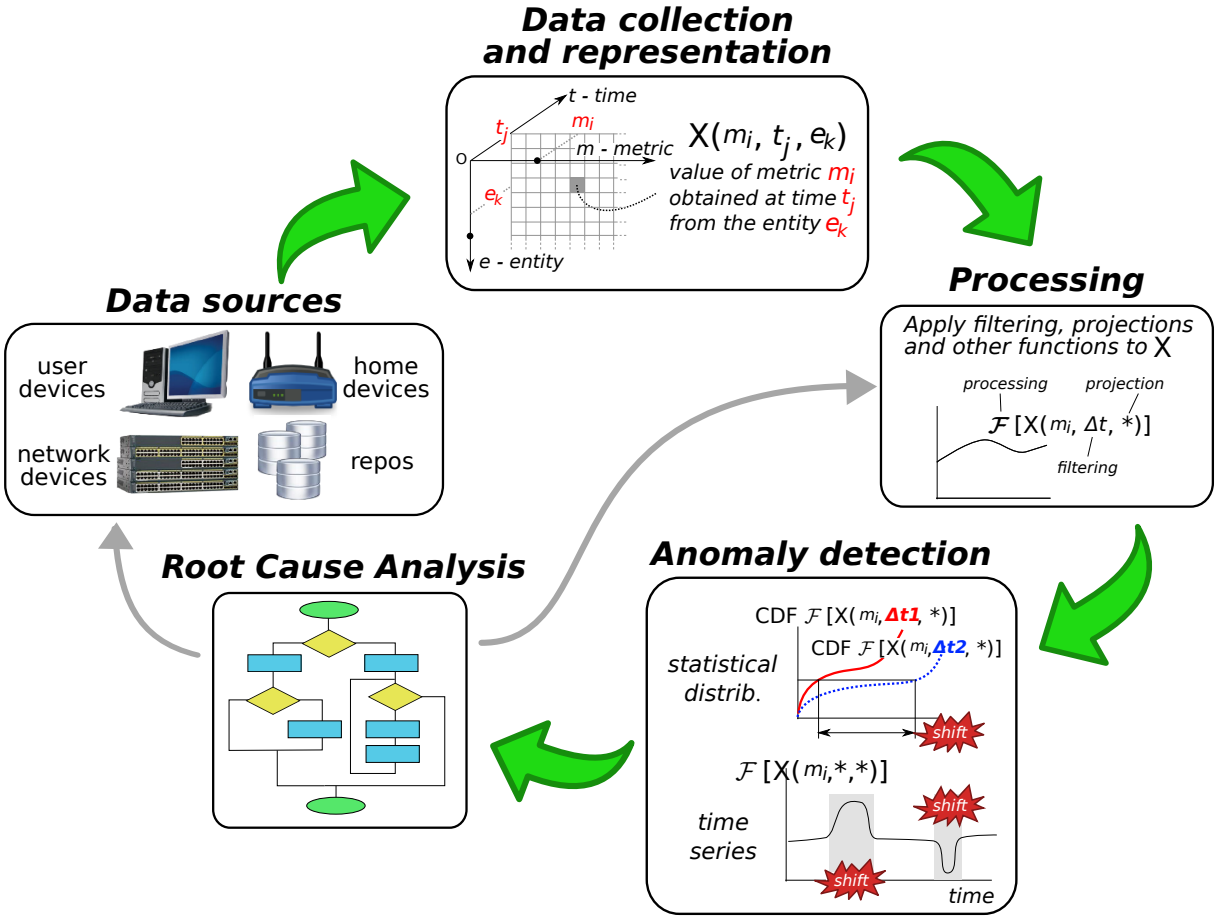


Figure 3.2: Workflow of the data processing as seen from the anomaly detection and root cause analysis module.

data sources for the analysis. These correspond for example to historical data collections already available from e.g. legacy systems.

Data at this level of analysis can be considered as ``raw data'' and need to be post processed applying filtering and other transformation to obtain data suitable for the anomaly analysis. Similarly, given the generic metric definition, they generally can correspond to per-flow logs, histograms, time series, etc.

Data collection and representation: measurements are saved in repositories to create an historical view of the network evolution, and for further processing. This is specially useful for anomaly detection and root cause analysis purposes, which normally requires to analyze the past history of the data related to some special event. Independently from their specific format, we can picture measurements as a 3 dimensional matrix X separating m - the metrics monitored, t - time and e - the monitored entity or source of measurements, which corresponds to the ``object'' on which the metric is applied (e.g., a device, a network link, a network subnet, etc.).

Processing: we can process data collected in X to ``cook'' them (aggregate, filter, transform, etc.) according to our needs. Generally speaking, both *coarse grained* and *fine grained* analyses are needed. For example, we may need to extract traffic trends picturing the evolution of

the network over the years, considering an aggregation of users. For instance, we could be also interested in the analysis of per-subnetwork or even on a per-user data basis, in smaller time windows such as hours or minutes, specially in the case of detecting short-time-scale anomalies.

Considering the type of output, anomaly detection is usually applied on time series, matrices and other statistical distributions such as PDF of CDF. As such, aggregation and filtering operations are needed also to adapt the input data to these formats.

Anomaly detection: the output sequence of the processing can be finally fed to the anomaly detection module where traffic can be analyzed over time. More in details, this analysis is based on the principle of extracting a representation of the system in ``normal'' condition, which is then used as a reference to identify the times when the system presents abrupt and unexpected deviations from it. In general terms, a network traffic anomaly corresponds to a traffic pattern that does not conform to the established normal behavior, but can also be extended to the notion of ``outlier'', which represents a pattern with statistical properties which are markedly different from those describing the majority of the patterns.

Root Cause Analysis: the analysis identified during the previous phase step are only the symptoms of possible issues. The final goal of this use case is not only to detect the symptoms, but also to diagnose them. i.e., to find the reasons for those observed anomalous behaviors. Finding the root causes of an anomaly requires to crosscheck the detected anomalous patterns, considering historical data collections or other portions of the network under monitoring. This analysis step considers not only the search for correlations among different events related to the observed anomaly (e.g., a similar anomalous behaviors observed in other parts of the network, or observed in the past, or some other indications of abnormality that may correlate with the detected event), but also the identification of ``causality correlations'', where we need to identify which events were the responsible for triggering others events involved in the ``causality chain'' of the observed anomaly. Similarly, to drill down to the root of the problem, multiple iterations on the analysis might be needed or new measurements may have to be deployed on the probes. This requires interaction between the root cause analysis module and the previous modules as indicated by the arrows in Fig. 3.2.

The input of the workflow analysis is independent from which source the data are collected. For example, assume we are considering some passive data related to the monitoring of average download bitrate of content served by a CDN and that the anomaly detection has identified a sudden drop of the performance. Besides investigating on historical passive measurements already collected, we can decide to start a measurement campaign of *active* measurements to investigate the causes of the traffic shift.

We aim at investigating traffic patterns from huge amounts of data. Therefore, wherever there is the need to process data, technologies for big data analysis have to be considered (e.g. Hadoop clusters, MapReduce paradigm, etc.). Thus, proper APIs and query languages to big data exploration technologies have to be integrated in the chain of analysis.

3.7.4 Metrics and measurements

Given the heterogeneity of services and type of devices, there is a huge set of metrics that can be considered as valid examples for the anomaly detection process. While on the one hand focusing

on some specific metrics allows to investigate how they affect the system design, on the other hand this might result in a limited vision of the system capabilities. In this description we consider the generic problem which considers any generic metric, using different possible time scales. As an example, we are interested in monitoring metrics related to the following scenarios:

- **CDN data centers evolution**: several measurements can be adopted to measure ``network distances'' between the server and the client. For example RTT, hop count, AS count, are commonly considered. By constantly monitoring them minimum RTT we can track network changes. Also, we can study architectural and load balancing properties of a CDN, e.g., identifying time variant load balancing policies and/or the deployment of a new data center on the CDN. Other metrics such as the *download bitrate* are instead related to performance and could also be useful to identify load balancing policies. For example, a drop in performance at peak hours can be due to policies redirecting the clients requests to ``far'' data centers, or to congestion events in the network.

- **Traffic share evolutions**: network probes should have a traffic classification module capable of computing metrics on a per-application base. Alternatively, the probes should collect data which allow to later classify the traffic (e.g., collecting a log reporting the HTTP requests with the headers). Using such capabilities we can derive trend on the evolution of Internet services such as YouTube and Facebook, considering their share of volume or their popularity across the users. It is important also to investigate variations of the portion of traffic which is not classified. For example, sharp changes in the amount of generic HTTP traffic can hide the presence of a new application that is becoming popular.

In general, it is not realistic to suppose that the network probes can provide filtered data ready to be used for anomaly detection. For instance, data transformations such as Principal Component Analysis (PCA), clustering techniques, statistical-based detection models, or in general any kind of data mining technique can be used to extract important traffic characteristics that can be monitored over time. However, these techniques generally require complex computations which cannot be performed on the probes due to resources requirements.

3.7.5 Repository schema and features

Repositories are fundamental for the general functioning of the work flow previously described. First of all, pattern analysis and trend evolutions require to observe recent periods of time (e.g., minutes and hours), and historical collections (e.g., months or years). Moreover, in presence of anomalies observed from a probe, crosschecking have to be performed using data collections obtained from other vantage points or from other measurements in general. This means that the analysis of the data collected is not local to a single probe but require a wider view of the network. Finally, the output of the preprocessing and time series generated by the probes can be saved in the repositories to speed up comparisons and further post-processing.

Therefore, the majority of the data collected correspond to i) data logs ii) histograms iii) output of previous processing. Each collection has to be associated to a schema detailing both the format (e.g., the column names and types) and the description (e.g., the data set has been collected from the probe X, between *date1* and *date2*, etc.).

3.7.6 Application of iterative measurement

In case the anomaly detection step identifies deviation from normal condition, the root cause analysis module is responsible for the investigation of the causes of the variation. In the context of the overall system architecture, this module is a component of the *supervisor* that interacts with it to perform further analysis in an automated fashion. For example, the module should try to answer questions like the variation is local to a single vantage point, subnet, user, server, etc. or it can be observed from other portion of the network?; is it already happened in the past?; is this reflected in other measurement as well?; etc.

3.7.7 Results

With the help of this framework which aim at improving anomaly detection and trouble-shooting in large-scale networks:

- **Operators**: will be able to fast detect unexpected events (e.g., apparatuses failures, flash-crowds) out of regular traffic patterns, and to possibly apply proper countermeasures (e.g., changing routes, blocking specific traffic patterns, etc.). In other words operators will increase their management efficiency thanks to a continuous and accurate control of the status of their networks.
- **CDN provider**: will be able to monitor the efficiency of their resources, thus selecting the best policies to optimize content caching.
- **End-users**: thanks to the continuous network monitoring, they will experience a better and more secure service delivery and in a total privacy-safe fashion.

3.7.8 Applications for inter-ISP scenario

Anomaly detection and root cause analysis can involve multiple ISPs both considering measurements (e.g., metrics collected from probes running in different ISPs), and anomalies (e.g., detection of anomaly in an ISP and verification if the same anomaly is present in another ISP). When finding the causes of an anomaly, combining measurements from multiple ISPs can enhance the diagnosis of the causes (e.g., in the case of anomalies related to a path traversing multiple ISPs).

3.7.9 Architectural Requirements

The design of the system has to offer high flexibility considering how data are collected, internally represented and processed. The system has to be flexible enough to monitor different metrics and also offer post processing capabilities as to adapt the data collected to the different needs of anomaly detection and root cause analysis. Overall, the monitoring system has to decouple the data collection process from the data analysis. The probes must be capable of advertising the type of measurements they provide, including time scale, units of measurements, etc. Data must be stored in repositories capable of keeping information for week and possible month duration. Algorithms must run periodically and on batches of data coming from the probes or available at the repositories. Iterative analysis can be triggered after an alarm is detected.

3.8 Verification and certification of service-level agreements

The considerations reported in this section are partially derived from experience gained by mPlane partner FUB in the framework of the Italian Resolution (AGCOM DEL 244/08/CSP) to implement the Italian QoS monitoring network "MisuraInternet" with the aim of verifying customer SLA in wireline access networks. Additional information is given in the Appendix, section A.3.

The ubiquity of Internet access, and the wide variety of Internet-enabled devices, have made the Internet a principal pillar of the Information Society. As the importance of the Internet to everyday life grows, reliability of the characteristics of Internet service (availability, throughput, delay, etc.) grows important as well. Service Level Agreements (SLAs) between providers and customers of Internet services regulate the minimum level of service provided in terms of one or more measurable parameters.

The verification of an SLA is technically equivalent to the verification of the implicit guarantees of service made by a provider offering Internet service. Currently SLAs are tested in terms of some network performance parameters as "bandwidth'' (generally expressed in terms of raw throughput). However, with the evolution of the applications SLAs will regard aspects more and more related to user perception.

Therefore we first need to define new metrics that take into account Quality of Experience (QoE), and investigate on the introduction of SLAs based thereon. In this scenario, each user may have SLAs with different providers (e.g. ISP, VoIP, IPTV); we therefore need to consider correlation among different SLAs. This will permit both the verification of SLAs between providers and customers from either the provider or customer end, as well as the customer-end verification of advertised throughput for comparison and regulatory purposes.

In this scenario, we look at both the verification of SLAs between providers and customers from either the provider or customer end, as well as the customer-end verification of advertised throughput for comparison and regulatory purposes.

3.8.1 Scenario Description

An SLA is a negotiated agreement between two parties, where one is the customer and the other is the service provider.

The SLA records a common understanding about services, priorities, responsibilities, guarantees, and warranties. Each area of service scope should have the ``level of service'' defined. The SLA may specify the levels of availability, serviceability, performance, operation, or other attributes of the service, such as billing. The ``level of service'' can also be specified as ``target'' and ``minimum'', which allows customers to be informed what to expect (the minimum), while providing a measurable (average) target value that shows the level of organization performance. In some contracts, penalties may be agreed upon in the case of non-compliance of the SLA. It is important to note that the ``agreement'' relates to the services received by the customer, and not how the service provider delivers that service.

SLAs commonly include segments to address: a definition of services, performance measurement, problem management, customer duties, warranties, disaster recovery and termination of agreement. Any SLA management strategy considers two well-differentiated phases: the negotiation of the contract and the monitoring of its fulfilment in real-time. Thus, SLA Management encompasses

the SLA contract definition: basic schema with the QoS and/or QoE parameters; SLA negotiation; SLA monitoring; and SLA enforcement—according to defined policies.

The SLA specifies the metrics in which the level of service is expressed: availability, throughput, latency, according to the application; availability is the most common of these for simple provider/customer relationships; throughput is most commonly used in verification of advertised capacity; finally, latency and latency variation are useful indicators for interactive applications.

3.8.2 Scenario Narrative

SLA verification is generally a continuous process. Therefore, the mPlane infrastructure of the service provider, or of an enterprise customer, is set up to monitor a given SLA at the time the SLA is committed; the supervisor then instructs the probes to periodically perform measurements, either passively or actively, and compares measured parameters to stored SLA parameters, until such time as the SLA is discontinued.

Periodically generated results or reports are then either forwarded by the supervisor to the appropriate administrator or regulatory authority, or stored in a repository for later retrieval. SLA violations can also be alerted by the supervisor; the exact reporting and alerting behavior is subject to the terms of the SLA.

3.8.3 Metrics and measurements

The metrics required for SLA analysis are those which are used to specify the level of service required; for the scenarios envisioned:

1. simple availability of a service (i.e., a time-series of yes/no states),
2. throughput between two endpoints given application-unlimited demand,
3. throughput between two endpoints given constant-rate demand,
4. latency between two endpoints given a load profile, and/or
5. application-specific metrics for QoE measurement.

These measurements should be taken in the presence of other normal background load at both endpoints.

With the service evolution, the trend is to shift the verification towards applications. This implies that metrics and measurements have to be applied at the application layer; suitable SLA parameters must be defined for specific Web services such as YouTube, specified in terms of application-level measurable parameters (e.g. measured video stall rate). These Application SLAs (ASLAs) will necessarily depend on lower-layer SLAs: clearly, for high-quality HD video it is necessary to have a very good wide broadband connection that has to be guaranteed by a suitable SLA between ISP and user, even though that is not sufficient to guarantee very high quality services, e.g. HD video services.

In measuring service levels in terms of throughput on a broadband connection, the SLA is often defined in terms of the physical characteristics of the connection from the user gateway to the network (by means of either twisted pair, or fiber, or 3G/4G radio access). This implies measurement

method that is able to exploit all the capacity of the line; the most suitable method here is an active probe. Since dedicated hardware devices for measurement are prohibitively expensive for home gateway applications, this active probe should be based on a software tool that a user can download and run on their home PC, laptop, or tablet. This has the disadvantage of running in a relatively uncontrolled environment; specifically, it may be difficult to isolate last-mile from in-home or on-device causes for measured phenomena.

3.8.4 Repository schemas and features

Repositories for storing SLA verification information must be able to store the metrics listed above as well acceptance criteria for SLA comparison.

3.8.5 Data analysis tasks

The primary data analysis task in SLA verification is the comparison of metrics to acceptance criteria.

3.8.6 Applicability of iterative measurement

SLA verification is not inherently iterative: generally, measurements are either passive or periodic. The supervisor, in this case, is required only to control the measurements, to generate periodic reports in the format required by a regulator, to alert the administrator in case an SLA is not met.

3.8.7 Results

This scenario produces reports on the SLA compliance of a given link or link aggregate in time-aggregate or time-series.

3.8.8 Inter-domain considerations

SLA verification is an inherently cross-domain operation, as there are always at least two parties to such an agreement. However, prior work on SLA verification has focused on verification from the point of view of a single party. This is largely a matter of trust: an entity doesn't necessarily trust the measurements taken by its counterpart in the SLA.

In the case of QoE-based SLA, however, measurements must necessarily be taken at or close to the client; in this case, an mPlane infrastructure at a customer network could take measurements of controlled video transmission (e.g., with the ISP actively downloading videos emulating user behavior), and report the results back to the provider network to inform it of SLA-relevant parameters; the provider could then compare customer-visible measurements to provider-network measurements in verifying its own compliance to an SLA.

4 Existing systems and standards

The problem of distributed measurement of observable performance at network endpoints is as old as the Internet; as such, there has been much prior work in this area. Many of these efforts focus on creating measurement infrastructures and communities, as opposed to platforms. For example, CAIDA's Archipelago [6] produces topology measurements using active measurements in the data plane, while Routeviews [25] passively observes control-plane messages. RIPE produces RTT and topology information in its Atlas [19] project, while TTM[20] provides delay, loss, and bandwidth measurements among about a hundred observation points within IPS networks; both rely on active measurements. PerfSonar [12], on the other hand, provides a measurement infrastructure construction kit, including a set of tools for measuring various network parameters, and a service-oriented architecture for accessing these tools across multiple observation points.

These projects are focused on network-core level measurements, and delivering information likely to be of use for enterprise or provider network management. Recent work has focused on expanding measurement for residential and small-business broadband users beyond simple speed test applications, as in QoS and SLA verification scenarios. ICSI's Netalyzr [14] is an active measurement tool that runs in a browser and provides comprehensive reachability, latency, and bandwidth analysis, and diagnoses common problems along with an explanation of each; the data is used in aggregate for research purposes. BISmark [23] measures the broadband gateway instead, supplying firmware for Netgear routers, and focusing on aggregate throughput and latency, allowing users to compare their performance to others.

While PerfSonar does provide a Web Services-based measurement control plane, it is not aimed at the inclusion of existing measurement tools. None of these systems incorporate iterative measurement or a supervisor role with a potential for semi-automated or automated root cause analysis process with the support of an intelligent reasoner.

Regardless of the nature of the interface(s) chosen to mPlane components, they will require a common *vocabulary* of metrics to be measured by the probes and features to be calculated on those metrics. The definition of a registry of these metrics is essential to the interoperability of mPlane components, but defining a registry from scratch would be a daunting task for a research project of the scope of mPlane. We therefore must leverage existing standards as an initial basis for this vocabulary.

The IP Performance Metric (IPPM) Working Group (WG) of the Internet Engineering Task Force (IETF) is pursuing standardization to this end [5], and it is expected that mPlane will be able to reuse much of this work. Many of the metrics in question are already in used and well understood. This will certainly help to make mPlane components to interoperate with existing measurement infrastructures and leverage already collected data. Well-defined metrics and measurement methodologies to arrive at these metrics will be the underlying basis for all mPlane components.

Another aspect regarding standards is the data exchange format. mPlane set out to tap into existing repositories in order to not reinvent the measurement wheel. Regarding these existing repositories, some work has already been done in terms of standards or specifications. For example, PerfSonar uses an XML schema defined by the Open Grid Forums Network Measurement Working Group. These may be used by mPlane for data import.

The IP Flow Information Export (IPFIX) WG in the IETF has defined an information model of simple data types along with a typed data export protocol [7] well suited to asynchronous data export of network-relevant data. mPlane will consider the use of IPFIX as an asynchronous data export, but

also consider the IPFIX Information Model and Information Elements Registry [18] as a basis for simple types.

There is little standardization work to be leveraged for mPlane control interfaces at this point. There is work ongoing at the Broadband Forum (potentially based on TR-69) and emerging work at the IETF (an effort called ``Large Scale Measurement of Access Network Performance'' or LMAP in short). It is not expected that this work will closely map to the interfaces work required by the mPlane components, however, certain aspects might be leveraged as this work progresses, especially to allow future LMAP-compliant probes to be integrated into mPlane infrastructures with minimal control adaptation.

5 Requirements

This section enumerates the requirements for the mPlane *platform*: those requirements which must be met by the components and the interfaces between them in order for them to inter-operate.

These requirements are drawn from an analysis of the scenarios elaborated in section 3, and are split into **component**, **interface**, and **regulatory and legal**. Component requirements describe what the platform and its components will do, interface requirements describe the constraints on these interfaces, and regulatory and legal requirements describe how it will interact with the regulatory environment in which it operates, especially with respect to privacy and data protection regulations.

5.1 On data types

A measurement platform is necessarily concerned with the movement of typed data from the measurement edge, where the data is generally less refined, through decision and analysis points, where the data is refined, to generate information and knowledge about the measured network.

A type in the mPlane platform is defined by a set of values and the operations that can be performed on those values. Types can be composed of other types into tuples. The most simple type describes a single (scalar) data value with a name and an underlying primitive abstract data type: for example ``byte count'', which is an integer; or ``source IP address'', which is an IP address. This is roughly analogous to a column name and type in a relational database, to an information element in IPFIX, or to an element name in an XML schema. Complex types are composed of simple types, and are analogous to a table definition in a relational database, a template in IPFIX, and an element or document schema in XML. Two complex types are said to be *compatible* over the intersection of simple types they contain: one can be converted into the other, omitting values only found in the source type and containing null values for those only found in the destination.

Many measurement operations are concerned with the production of values of a single complex type of information, or the aggregation/analysis of a set of source values of complex types into a result value of a complex type. As typed data becomes more and more refined through the platform, the type itself generally becomes more and more complex.

In mPlane, both types are named; two types are considered to be the same across the platform if their names are equal. Federation of namespaces (e.g., to incorporate external type registries) is supported by this arrangement but not necessarily required within the platform.

This concept implies that the choice of a set of types is crucial to the inter-operation of components in an mPlane platform; the definition of a type registry is therefore the next significant task in this work package. Work will draw from existing standards and will be elaborated in a future deliverable.

5.2 Component Requirements

The mPlane architecture is designed to allow the maximum possible flexibility of component functionality and deployment. Any software or hardware component which implements the mPlane interfaces is part of the mPlane platform. The mPlane interfaces include a capabilities specification

facility to allow components to specify what they can do and to discover the capabilities of their component peers. The capabilities so advertised determine the role each component has within the platform. This capability specification is bidirectional; that is, it can either be initiated by the advertising component (push-model), or initiated by the requesting component (pull-model); see section 5.3.1.1.

That said, components can be split into three broad categories: **probes,** which measure and analyze traffic; **repositories,** which store and analyze data produced by probes and/or provide interfaces to external data sources; and **supervisors,** which coordinate the actions of probes and repositories.

The ``component'' that gives orders to a supervisor is termed an mPlane **client**; this represents a user interface behind which a human analyst sits, a backend application using mPlane to generate periodic reports, and so on. Reasoners are a special type of mPlane client. Reasoners are a special class of client within the platform; they use mPlane components through a supervisor and provide intelligent reasoning capabilities for iterative measurement automation.

A group of components accepting commands from one or more coordinated supervisors is termed an mPlane **infrastructure**.

5.2.1 Base component requirements

As stated, any software or hardware component which implements the mPlane interfaces is part of the mPlane platform. At minimum, an mPlane component must:

1. specify its capabilities to other mPlane platform components and/or clients,
2. accept commands via a *control interface,* and
3. accept and/or produce data via a *data interface,* as appropriate to its function and capabilities, and
4. identify itself using an identifier unique to an mPlane infrastructure.

Capability specification may be requested from a supervisor or sent to a supervisor independently, depending on the deployment.

Components may also support a *management interface*, via which the component is managed. Management actions include software updates, access control configuration, and other administrative actions. These are separated from the control interface since i) they are often component- and application-specific, and ii) they have other access control requirements than measurement control. In general, the control interface is used to *use* the capabilities of a component, while the management interface is used to *modify* the capabilities of a component.

5.2.2 Probe requirements

A probe is an mPlane component which measures activity (network traffic, host activity, etc), may perform analysis thereon, and exports information to repositories or supervisors continuously or on demand. Probes may keep some local state but generally do not provide large-scale data storage.

In addition to base component requirements (section 5.2.1), a probe must be able to:

1. perform measurements and/or analyses of data from an observation point; the set of measurements derived from the chosen scenarios is given in section 5.2.2.2.

Probes always produce data. The actual actions performed by a probe are dependent on that probe's capabilities, and the application it supports.

5.2.2.1 Capabilities

Probe capabilities include but are not limited to:

1. the measurement(s) the probe can perform,
2. the types of data the probe can return about those measurements,
3. whether the probe supports synchronous data export (section 5.3.2.1) and/or delayed data export (section 5.3.2.2),
4. the asynchronous data export protocol(s) (see section 5.3.2.3) the probe supports,
5. the capacity of the probe, expressed in terms of maximum observable data rate and/or maximum rate at which measurements can be produced and exported,
6. additional operations the probe can apply to the measurements, if any,
7. the probe's programmability (see section 5.2.2.4), if any,
8. the probe's timestamp generation capability, timing precision, and accuracy (see section 5.2.2.5), and
9. a specification of the probe's deployment location: the link(s) the probe can observe, in the case of passive measurements; or the address(es) from which the probe can perform measurements, in the case of active measurements.

5.2.2.2 Measurements

While the architecture must support a fully flexible and extensible set of measurements at the probes (see section 5.2.2.1), the following measurements, derived from the candidate scenarios, are those which will be prioritized during the project.

1. One-way delay and connectivity metrics (see RFC 2679 [2])
2. One-way packet loss metrics (see RFC 2680 [3])
3. Round-trip delay and connectivity metrics (see RFC 2681 [4])
4. Delay variance ("jitter") metrics (see RFC 3393 [8])
5. One-way throughput metrics
6. Path hop and AS count

7. Nominal path bandwidth and nominal path bandwidth-delay product
8. Inferred path congestion
9. Unidirectional network flow data (see RFC 5101 [7])
10. Bidirectional network flow data (see RFC 5103 [24])
11. Enhanced network flow data, including application-layer goodput and loss metrics
12. Labeled traffic share (in effect, flow aggregates classified by application)
13. Content item statistics (flow aggregates classified by pseudonymously-identified content-item)
14. Link utilization and available link bandwidth
15. Infrastructure unit (e.g. router or mobile base station) utilization
16. Mobile base station channel utilization and radio characteristics
17. Application request/response time (e.g., actively probed HTTP, passive DNS/HTTP etc.) and connectivity
18. Application session request/response time (e.g., full web-page load)
19. Streaming buffering delay, buffer status, stall count and duration
20. Nominal and observed video frame rate
21. End-device status: CPU, memory, energy, radio power, radio SNR
22. Path and service change events -- i.e., an assertion that a path has changed its characteristics (e.g., link failure), or will change its characteristics in the future (e.g., planned outage)

The platform must support probes which are capable of more complex measurements, including analysis of data in place at the probe. A common in-probe analysis task is time-series or key-space aggregation; therefore, probes may also generate aggregates of any of the measurements above.

Not all measurements will necessarily be developed or demonstrated during the project; additional measurements not in this list may be provided by probes developed in the course of the project.

5.2.2.3 Synchronous and asynchronous measurement

The platform must support probes that can return instantaneous measurements upon demand (synchronous measurement), whether instantaneous or delayed, as well as probes that send data periodically or continuously based upon events observable at the probe or a schedule of measurements taken at the probe (asynchronous measurement). Not every probe must support both modes of operation.

5.2.2.4 Programmability

mPlane probes need not be programmable; however, for probes that provide a programming language or other facility for dynamic definition of measurements, this programmability should be represented as a capability. The mPlane platform need not provide a translation layer for probe programs; each probe's programming language is specific to that probe. Programs are managed via the management interface. Newly loaded programs are subsequently represented as new capabilities.

5.2.2.5 Timing precision, accuracy, and synchronization

Timing precision is key to any time-based metric, while timing accuracy is key to synchronization of measurements taken at multiple probes. Probes must know the precision and accuracy with which they represent timing information, and represent these as capabilities. Clients and supervisors may use this information to determine whether a given probe meets their timing requirements; precision and accuracy information may also appear in a probe's results, to allow later estimation of timing error during data analysis.

Clock synchronization is to be done using existing protocols, e.g. the Network Time Protocol (NTP) [16] or Precision Time Protocol (PTP) [13]. Probe applications requiring very high timing accuracy should use external clock sources (e.g., GPS). Clock synchronization parameters may be configured using the management interface, and are component-specific.

5.2.3 Repository requirements

A repository is any source of measurement data which does not directly measure that data itself. Repositories may provide large-scale data analysis and storage services, or may provide access to some large external source of data.

In addition to base component requirements (section 5.2.1), a repository must be able to:

1. allow storage and retrieval of measurement data received from probes or other repositories (see section 5.2.3.2 for a list of data types to be supported) and/or
2. provide retrieval of externally sourced data, and/or
3. perform analyses on stored or collected measurement data, and/or
4. merge/fuse data from multiple sources.

The actual actions performed by a repository are dependent on that repository's capabilities, and the application(s) it supports.

5.2.3.1 Capabilities

Repositories can specify *capabilities*, which must include but are not limited to:

1. the types of data the repository can accept,

2. whether the repository supports synchronous data export (section 5.3.2.1) and/or delayed data export (section 5.3.2.2),
3. the asynchronous data export protocol(s) (see section 5.3.2.3) the repository supports,
4. the types of data the repository can produce,
5. the transformations/analyses the repository can apply to the data,
6. in case data is aggregated the level of aggregation should be provided,
7. if data from multiple sources is merged, control information exposed by the repository should summarize the control information received from the probes (e.g. location of the probe), and
8. the capacity of the repository, expressed in terms of record acceptance rate and production rate, as well as maximum available storage.

5.2.3.2 Data types

The data types supported by a repository are application-specific, and the architecture must support a fully flexible and extensible set of complex types and analyses at the repositories (see section 5.2.3.1). The complex types to be prioritized during the project are derived from the types of measurements to be supported by the probes (see section 5.2.2.2); the repositories will support:

1. single values of each type, annotated with timestamp, observation point (i.e. probe) identification, and appropriate parameters (e.g., a delay report contains the source and destination address of the measurement)
2. aggregate values of each type over time, annotated with time interval and appropriate parameters; these can be combined for trending

Note that some of the measurements in section 5.2.2.2 can be derived from others; these derivations may take place at a repository as well as being directly performed at a probe.

The platform must support the use of complex types that support additional basic types other than those required by section 5.2.2.2. Not all complex types will necessarily be developed or demonstrated during the project. However, a *given* repository itself does not need to support any type other than those required for the application(s) they support.

5.2.3.3 Specialized external repositories

Some envisioned scenarios require specialized repositories for providing access to external data sources to mPlane components; these may be implemented as needed within the scope of the project.

Some measurement operations must compare multiple data points for a given user; however, the platform must not support allow for personal user tracking. Therefore, the database mapping IP address(es) within an operator's network to a user identity must be made available through a *user pseudonymizer*: instead of returning a user identity for an address, it would return a non-reversible pseudonymous identifier which maps to a user identity for some limited time period.

Scenarios using path characteristics can benefit from routing topology information derived from a control plane looking-glass; a repository built around an internal looking-glass or a well-known service such as Route Views [25] would provide this information.

Scenarios based around content popularity and content migration in CDNs can benefit from location-based aggregation. A repository built around a IP geography database (e.g. MaxMind GeoIP[1]) would provide this mapping.

5.2.4 Supervisor requirements

A supervisor is an mPlane component that coordinates the measurements performed by the probes and analysis and storage of data at the repository. The supervisor iteratively drills down into the cause of a phenomenon, determining the conditions leading to given issues, and supporting the understanding of problem origins and/or general phenomena.

The supervisor has a set of analysis modules capable of automatically processing the historical and/or the real-time data coming from both the repositories and the probes. These analysis modules provide different analysis services to the reasoner, which correlates and synthesizes results to retrieve high-level answers.

In addition to basic component requirements in section 5.2.1 supervisor must be able to:

1. specify its capabilities to another mPlane supervisor or mPlane client,
2. maintain a registry of the capabilities currently provided by the mPlane components it supervises, and allow registration of these via a registration interface,
3. issue control messages to other components (probes, repositories, supervisors) on behalf of clients, to perform iterative measurement and analysis,
4. accept results from these components, and make them available to mPlane clients,
5. perform analysis on received measurement data to provide results to clients or as input to further iterative analysis.

5.2.5 Reasoner requirements

Measurement for root cause analysis is an inherently iterative process. This iterative analysis is supported and automated by an *intelligent reasoner* which provides the intelligence and adaptability required by the applications supported by mPlane.

The intelligent reasoner has access to a set of domain-knowledge-based rules that helps it to correlate intermediate analysis results with new monitoring and analysis actions in the iterative analysis process. These rules are conceived as a knowledge structure where mPlane users can include domain-knowledge information related to the different monitoring applications that enrich the reasoner analysis. The set of rules is not necessarily static or only adapted by domain-knowledge information, but can may be augmented over time as the reasoner learns from successes and failures in the iterative measurement process.

The reasoner may be an mPlane client (that is, external to the supervisor, and interacting with the mPlane platform via the supervisor's control and data interfaces), or it may be tightly integrated into the supervisor.

The reasoner must:

1. analyze and correlate data from an mPlane infrastructure, using the capabilities of the components therein,

2. access and maintain a set of domain-knowledge-based rules that guides the analysis process,

3. perform data analysis and apply these rules to the analysis process to decide the next measurement step(s) to take, and

4. control components of an mPlane infrastructure through a supervisor to perform subsequent iterations.

While the goal of a reasoner is to automate the iteration of the root cause analysis process, it may also support semi-automated iteration: assisting a human analyst by performing ``easy'' iterative steps and leaving more involved decisions to a skilled analyst. Automation or semi-automation is an application-specific characteristic.

5.2.6 Interdomain Cooperation

Interdomain cooperation in mPlane is primarily supported by supervisor-supervisor delegation. In a *peer* arrangement, two supervisors in cooperating domains are clients of each other: one supervisor delegates measurement requests about the other domain to the other, and vice versa. In a *federation* arrangement, all the domain supervisors within a measurement federation are supervised by a federation supervisor; all the mPlane clients within the federation then delegate measurement to this higher supervisor. The selection of arrangement is application- and policy-dependent.

As supervisor-supervisor interactions may introduce delay that could invalidate a measurement, these may be set up in advance in order to avoid synchronization issues. Interdomain applications requiring synchronous data access to measurements in other domains must tolerate proxy or authorization delay at the domain edge.

5.3 Interface Requirements

This section enumerates requirements of the interfaces provided by an mPlane component; these are divided into *control*, *data*, and *management* interfaces.

Components newly implemented during the course of the project provide these interfaces as their primary configuration and data retrieval interface. Existing probes and repositories may be modified within the project to provide these interfaces, as well, or may have some or all of their functionality exposed via mPlane interface proxies to be developed during the project.

5.3.1 Control Interfaces

The control interfaces are used to determine component capabilities, to instruct components to perform measurements either instantaneously, periodically, or continuously, and to query probes and repositories for results or stored measurement data.

5.3.1.1 Capabilities specification and registration

Each component specifies its capabilities, and registers those capabilities with one or more supervisors, which in turn store those capabilities in order to track which components and capabilities are available. An mPlane infrastructure is then the set of components which are registered to a given supervisor.

This registration process can be initiated either by the supervisor (configured to query a component at a known address/port or scan for components in a network on a known port and retrieving its capabilities), or by the component (by sending its capabilities to a supervisor on a known port). Component(s) and supervisor(s) may support either method of registration or both, depending on application and policy.

The registration interface can be used to change capabilities (when a component is modified via its management interface), or to note that a given component is available or unavailable; availability in this case is simply another capability. Registrations may also be re-queried or re-sent periodically to ensure the supervisor has the most recent information on capabilities and availability.

5.3.1.2 Access control

The control interfaces provide access control: every control interaction must be specifically allowed based upon the identity of the requesting entity (i.e., supervisor, repository, or external client) and the operation requested. Access control configuration is supported over a management interface. Access control must use standard protocols and practices.

5.3.1.3 Measurement and analysis control

The control interfaces allow a supervisor or mPlane client to use the capabilities declared via the registration interface, whether as an instantaneous query (to be answered via synchronous data access, section 5.3.2.1, or delayed data access, section 5.3.2.2) or as an ongoing measurement (to be exported as in 5.3.2.3).

5.3.2 Data interfaces

The data interfaces are used to move measurement data from probes to repositories, probes to supervisors, and repositories to supervisors; and to return the results of queries of probes and repositories.

5.3.2.1 Synchronous data access

mPlane components must support *synchronous data access*: a supervisor or mPlane client must be able to ask for the result of a measurement of analysis on demand via the control interface, whether current or stored, and receive a result on the same channel. The representations used by this synchronous data access interface should be unified across all mPlane components, and must be based upon standard representations and protocols.

5.3.2.2 Delayed data access

mPlane components may support *delayed data access*: a supervisor or mPlane client must be able to ask for the result of an analysis which will take some time to complete, as with the synchronous data access, then return later to receive the result. The representations used by this delayed data access interface must be identical to those in the synchronous data access interface.

5.3.2.3 Asynchronous data export

mPlane components must support *brokered asynchronous data export*: it must be possible to set up a probe or repository to export data periodically to another component, whether driven by some timer or by asynchronous events (e.g. packet arrival or alert condition on a probe). The representations and protocols used by this interface are heterogeneous: probes which natively speak some export protocol should use this protocol. Note that this implies that the control protocol must provide a registry of asynchronous data representations and protocols, and that these representations and protocols are part of the capabilities of each component.

Control interactions must allow for the starting and stopping of asynchronous data export, as well as for the indirection of asynchronous data export to any repository or other destination.

The project will specify a small set of asynchronous data interfaces which are preferred for new development; these will be based on existing standards where available.

5.3.2.4 Private query access

Some mPlane components, especially those built around legacy probes or repositories, may provide built-in data retrieval interfaces which provide more functionality than the synchronous data access interfaces. These are represented as capabilities, but should not be used for inter-component communication.

5.3.3 Management interfaces

Management interfaces are used to configure mPlane components, including the programming of programmable probes, the addition of new analyses to repositories, the configuration and revocation of access control rights, as well as the low-level administration of the systems on which mPlane components run.

All components supporting dynamic access control -- that is, access control whose credentials and/or

authorizations can be changed at runtime -- must support a common *access control configuration interface* via which credentials for control interface access control may be added or removed. The root credential for this interface must be configurable out of band. This interface will be specified in a future deliverable.

Other management interfaces are in general component-specific; each management interface implemented by a component is represented as a capability of that component.

5.3.4 Interface principles

To the extent possible, the mPlane interfaces are:

Stateless: while state may be kept (especially about running measurements) at each component, that state is not necessarily transferred from component to component, and one component need not know about the internal state of another in order to properly interact with it. Specifically, this implies that, access control aside, all the information required to answer a request appears in the request. Registration is a necessary exception to this principle: supervisors must know about the capabilities of components they supervise in order to properly control them.

Idempotent: multiple requests for the same information will lead to a single action on the part of a component. Of multiple registrations, the last one wins. This allows a collection of components to recover from disruptions or react to reconfiguration simply by restarting registration or control.

Protocol-agnostic: Control and data interactions, while carried by a defined representation, may occur over multiple protocols (e.g. HTTP, SSH, BitTorrent).

Initiator-independent: The ``sender'' in a given interaction is not necessarily the connection initiator in that interaction.

Statelessness and idempotence together help in error recovery of widely distributed systems where some components (especially probes on end-systems or edge devices) may have intermittent activity and/or connectivity; here, the explicit tradeoff is simplicity of recovery for efficiency.

Protocol agnosticism and initiator independence similarly support multi-scale measurement in a wide variety of deployment scenarios.

5.4 Regulatory and Legal Requirements

Since the aim of mPlane is to collect various types of data from users and services, particular attention must be paid in order to protect the privacy of said data according to the European legislation. For this reason, the two main data privacy regulatory acts issued by the EU, the Data Protection Directive and the Data Retention Directive, are described in the following section.

5.4.1 Directive 95/46/EC - Data Protection Directive

The Data Protection Directive is a fundamental provision that lists some general principles that regulate the collection and processing of personal data. These principles represent the basis for the European data protection legislation.

5.4.1.1 Personal Data

Personal Data is defined in the Directive as ``any information relating to an identified or identifiable natural person,'' where an identifiable person is ``one who can be identified, directly or indirectly, in particular by reference to an identification number or to one or more factors specific to his physical, physiological, mental, economic, cultural, or social identity.'' According to the Art.29 Working Party (the Data Protection Authority of the EU), also IP addresses may be considered as personal data, since they are an information concerned with an identifiable person.

5.4.1.2 Aggregation and Pseudonymization

Aggregation of Personal Data (using an irreversible process) may be considered as a form of anonymization, since it is impossible to revert back to the original subject of the data. Pseudonymization techniques consist of replacing the original (nominative) identities of subjects by pseudo-identities that cannot be linked directly to their corresponding nominative identities. This transformation can be implemented differently according to the project requirements, e.g. guaranteeing the unambiguous mapping of a given identifier with the same pseudonym; or the invariance in time, location, and content. Anyway it's preferable to assign to the same data subject more than one pseudonym if the data relating to the same data subject are used for different purposes or are to be used in different circumstances. Also in this case, similarly to the aggregation process, if the pseudonymization process is irreversible, the data may be considered anonymous. However, even when dealing with anonymized data, is essential to avoid the possibility of indirect identification by means of information cross-processing.

5.4.1.3 Obligations of Data Controllers

When dealing with Personal Data (that is, data that can be linked to a data subject, not anonymized nor pseudonymized), the data controllers must observe the following principles:

- Data must be processed fairly and lawfully
- Data must be collected for specified, explicit and legitimate purposes and not further processed in a way incompatible with those purposes
- Data processed must be adequate, relevant and not excessive in relation to the purposes for which they are collected
- Data must be kept in a form which permits identification of data subjects for no longer than necessary
- The data subject must be notified about data collection and processing, providing information about the Controller, the Purpose, and the Categories of data involved

- The data subject has the right to be informed on the purposes and conditions of the processing activities, and can modify or delete data regarding himself in any moment, even interfering in the processing.

5.4.1.4 Security Measures

Security measures are intended to serve a three-tier purpose: preventing risks and damages, reacting when threats occur in order to limit damages, and analyzing the incident in order to assess what went wrong and who might be accountable for the incident.
They have to be implemented to protect personal data against accidental or unlawful destruction or accidental loss, alteration, unauthorized access, and against all other unlawful forms of processing.

5.4.2 Directive 2006/24/EC - Data Retention Directive

The Data Retention Directive applies to traffic and location data on both legal entities and natural persons, and to the related data necessary to identify the subscriber or registered user. It aims to harmonize the Community member states' legislations in relation to the retention of traffic, location and other data generated or processed.
This regulation aims to ensure that the data gathered and processed are made available for purposes of investigation, detection, or prosecution of crimes.

5.4.2.1 Storage Requirements

The data retained under this Directive have to undergo appropriate technical and organizational measures in order to protect said data against destruction, loss or alteration; and to guarantee that said data may be accessed only by expressly authorized personnel.
In case of national member states' authority's request, the data must be transmitted to the same without any delay.

5.4.2.2 Statistics

On an yearly basis, the member states must provide to the European Commission statistics containing information on the retention of the data generated or processed in connection with provision of publicly available electronic communications services or provision of a public communications network.

5.4.3 Lawful Intercept and User Tracking

Support of lawful intercept and user tracking by elements of the mPlane platform is *specifically and explicitly* out of scope: lawful intercept infrastructure must be deployed alongside, not within, an mPlane infrastructure. Eavesdropping (i.e., partial or full packet payload and capture) and user identification and tracking (by IP address or username) are not possible within the platform by design, and components supporting the capture of packets or the personal tracking of user activity will not be developed by the mPlane consortium within the scope of the project.

6 Initial Architecture Guidance

Drawing from the stated requirements in section 5 and the first principles identified in section 2, this section sketches the interfaces to be implemented by mPlane components. This description is a work in progress, and may vary significantly from the future interface descriptions and implementations.

6.1 Arrangement of components

At the implementation level, there is no distinction among probes and repositories; mPlane components advertise their capabilities, and those capabilities, not a ``component type'', define what they can do. This allows flexibility in data processing -- e.g., pushing processing to the ``probe'' edge to increase scalability and decrease threats to end-user privacy due to aggregation, or building ``repositories'' that can verify service with low-impact active measurements.

Supervisors are also components, but they have more specific responsibilities: as they serve clients directly, they must track the state of other components and running measurements; handle finer-grained access control, proxy client requests to components, and handle requests for and route requests to supervisors in remote domains.

The general scheme of data and control flows, following from the requirements in section 5.3.2, is shown in Figure 6.1.

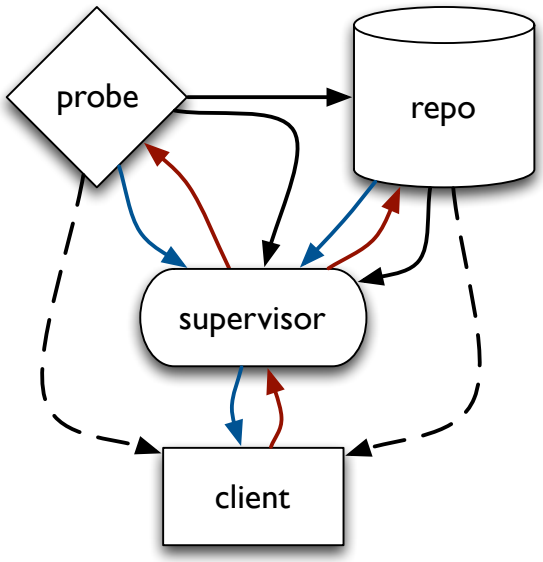
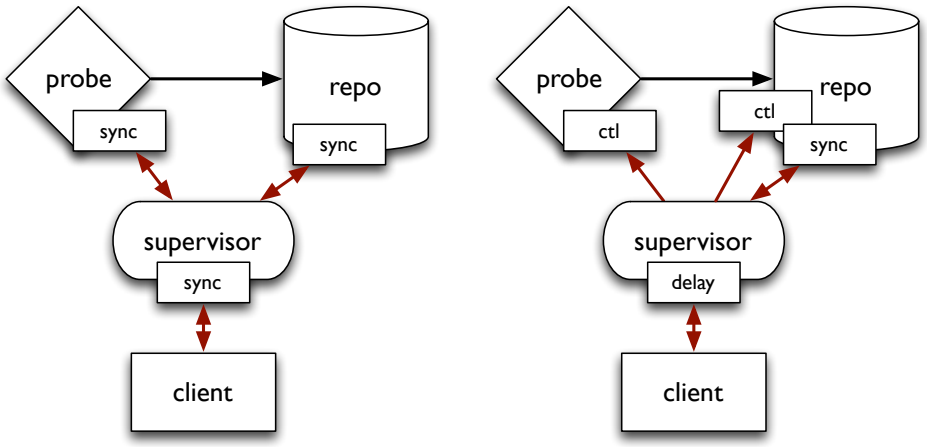


Figure 6.1: General scheme of data and control flows in mPlane (data in black, control in red, capability in blue; uncommon data flows dashed)

This figure shows each control/data flow from or to each component type; each of these flows is not, of course, active in every interaction. Considering the examples in Figure 6.2 shows a slightly different picture.

In the first, a client sends a request to determine whether the current delay between two points is



(a) Synchronous query

(b) Delayed query with synchronous retrieval and asynchronous collection

Figure 6.2: Example control/data flows

normal to the supervisor. The supervisor, in turn, asks a probe to actively measure current delay, a repository for historical delay figures, and compares the two to give an answer synchronously to the client

In the second, a client sends a request to retrieve time-series average packet loss numbers from several probes. Since the request involves the correlation of data from several probes, it is delayed. The supervisor sets up asynchronous data export between the probes and a repository to send intermediate results to the repository. The client then returns to retrieve the results, which causes the supervisor to retrieve them synchronously from the repository.

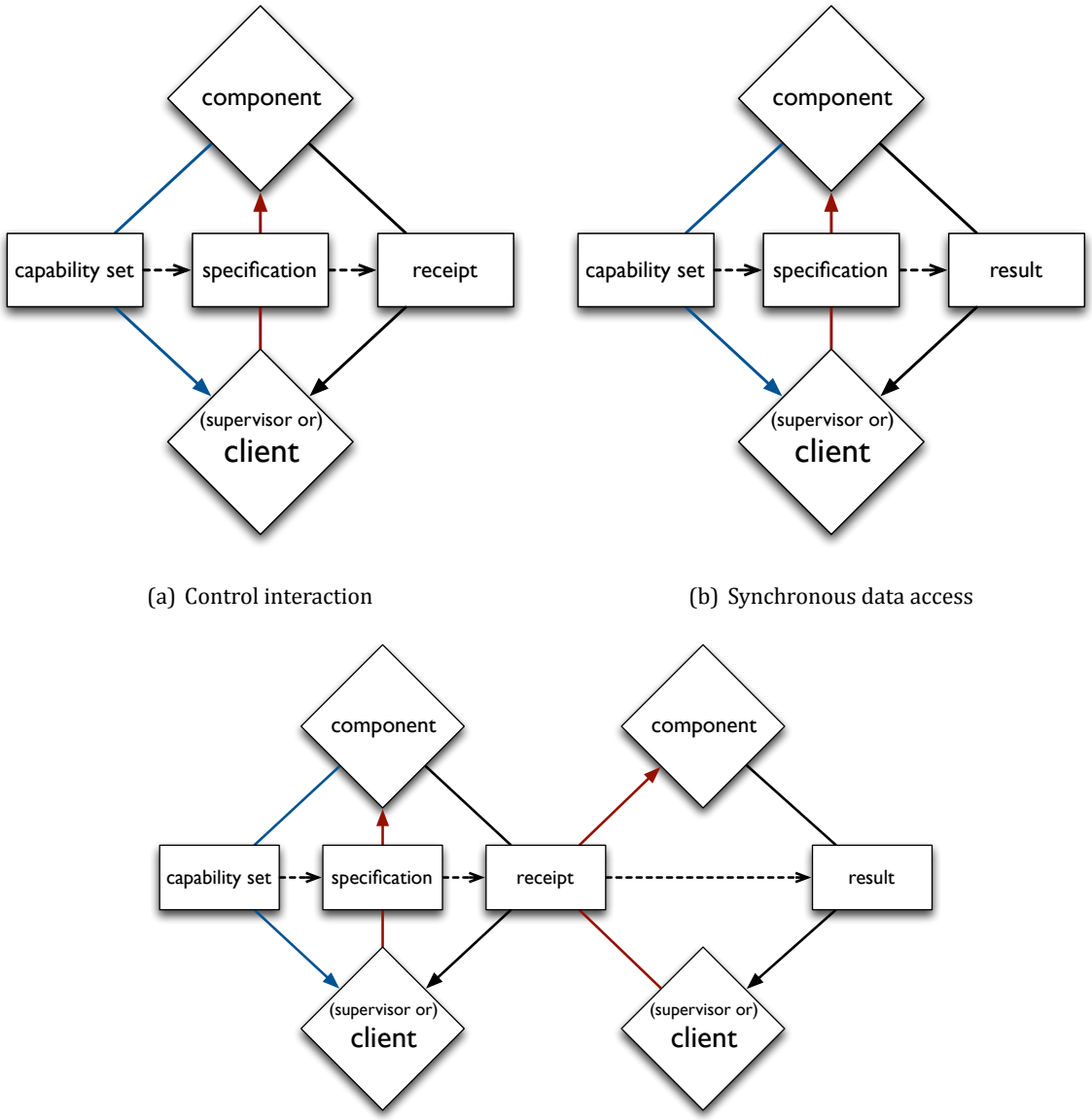
6.2 mPlane Control Dynamics

The mPlane interfaces in essence allow a client to take *capabilities* (a list of things a component can do) and build a measurement *specification* from that set of capabilities (such that each thing the component is told to do is in the set of things it can do). In the synchronous case, the component then turns that specification into a *result*; for delayed and control interactions, it returns a *receipt* instead. Each receipt has an identifier, such that it can be presented back to the component at a later time to retrieve a result if necessary.

The operation of interfaces in synchronous data access is shown in Figure 6.3(b); in delayed data access in Figure 6.3(c), and in other control operations in Figure 6.3(a). As in Figure 6.1, data flows are shown in black, control flows in red, and capabilities in blue.

In these diagrams, capabilities are ``filled in'' to become measurement specifications -- i.e., for a given set of the capabilities a component provides that the client wants to access, values are filled in for the declared parameters, and the rest of the capabilities are left out.

Measurement specifications are likewise filled in to become results: the specification itself was returned as well, along with values for the requested parameters, so that the result stands on its own as a fully specified data record. In other words, the answer to the question (written in En-



(a) Control interaction

(b) Synchronous data access

(c) Delayed data access

Figure 6.3: Interactions among components and clients using capabilities, specifications, results, and receipts

glish)``what is the one-way delay from 1.2.3.4 to 1.4.5.6 using UDP port 789 right now using 20 samples?'' would be ``the one-way delay from 1.2.3.4 to 1.4.5.6 using UDP port 789 at 14:31:12 on January 23 using 20 samples is 130.2ms with a standard deviation of 2.4ms'' as opposed to the simple answer ``130.2ms''.

To support stateless, protocol-agnostic, initiator-independent operation, the capabilities, specifications, receipts and results should share a common representation. To support easy development and debugging, this representation should be text-based, preferably using a structured representation for which parsers and generators of reasonably high quality exist in multiple languages. This would point to a structured markup representation such as XML, JSON, or YAML. The initial speci-

fication of interface definitions will be presented as a set of classes and relationships among them, separate from the serialization format; this is left as future work, and will be reported in a forthcoming deliverable.

6.3 Capabilities

The set of capabilities supported by an mPlane component is completely flexible; capabilities are identified by their names, so to support a new capability a component must simply state it is willing to provide that capability. The project will operate a registry of capabilities to enforce rules for capability names to reduce duplication in capabilities, and to provide a common measurement vocabulary for all components. Metrics, measurements and data types in capabilities will be defined by reference to external standards whenever possible.

The capabilities scheme supports the special status of data types within the system with special capabilities: ``produce'' and ``consume'' for simple or complex data types, and ``transform'' for the simple transformation of one data type into another. Data types will be registered along with capabilities.

A capability specifies a set of parameters which must be filled in, and a set of parameters which may be filled in, when requesting that capability. Parameters used to scope a capability (e.g., the time intervals supported, the observed network for passive probes or addressable endpoints for active probes, export protocols and destinations supported, and so on) share their names with the parameters used to specify the scope in a specification. Parameters will be registered along with capabilities, and basic parameters shared among capabilities to ensure a common vocabulary.

Special capabilities will be defined for verification of the suitability of a measurement: ``provides'' and ``requires'' to ensure e.g. that timing accuracy and precision constraints are met, as well as ``status request'' and ``status report'' to query the state of a component.

Further definitions of capabilities, parameters, and data types to be supported will be drawn from the requirements in section 5; the policies for the management of these registries are left as future work, and will be reported in a forthcoming deliverable.

A Service Level Agreement Verification

A.1 Background on Service Level Agreements

SLAs have been used since late 1980s by fixed line telecom operators as part of their contracts with their corporate customers. This practice has spread such that now it is common for a customer to engage a service provider by including a service level agreement in a wide range of service contracts in practically all industries and markets.

SLA can be defined at different levels both from the network infrastructure point of view and from service and customer point of view. In such a way SLA can be defined either at backbone or access level. On the other hand SLA can be introduced for web services (Web Service Level Agreement, WSLA) and can reach sophisticated monitoring technique when we refer to high level service as in the case of Cloud and Grid Computing.

Any SLA management strategy considers two well-differentiated phases: the negotiation of the contract and the monitoring of its fulfilment in real-time. Thus, SLA Management encompasses the SLA contract definition: basic schema with the QoS (quality of service) parameters; SLA negotiation; SLA monitoring; and SLA enforcement—according to defined policies.

The inability to meet service level agreements can often result in monetary damages, the provider of services, regardless of the specific type of service being offered, attempts to meet the service level agreements to the best of their ability.

A service provider may sign SLAs with difference performance objectives with different customers. The network operator needs to identify the type of packets coming into the network, so that they can be dealt with appropriate urgency. Once the SLA has been agreed upon, the network operator needs to monitor the performance of the network. The SLA would determine which network performance metrics ought to be monitored, as well as the operating ranges of the performance metrics.

The creation and filing of periodic reports is an important step in the process of supporting SLAs. The reports on monitored performance must be available for examination by the customer. A side-benefit of storing reports would be that the historic information can be used to extrapolate trends in network traffic, and thus be used as input to the service provisioning process.

If monitoring indicates that all the SLAs are being satisfied, there is no need for any further action. A worse case would be when the SLA objectives are not being met. In this case, the network configuration must be changed, through the service provisioning process, so that the objectives can be successfully met.

As a last step of the customization process, one must examine if the agreed SLAs can be satisfied. If experience shows that the SLAs cannot be met, one may want to revise the performance objectives to those that are feasible to meet. One can also revise SLA objectives to become more stringent, if that is likely to attract new customers and new streams of revenues.

SLAs for enterprise customers differ than those for end users; In the latter case the SLA generally regards a few network parameters related to the access network as bandwidth (or throughput), jitter, latency and packet loss. Conversely in the former case SLA can regard several network aspect regarding the particular network connection under consideration (for an instance at access, backhaul, metro and core networks; ATM or GBE or WDM); in such a case we can refer for an example to bitstream services including provisioning and monitoring.

A.2 Background note on QoS vs. QoE for multimedia content

From the user point of view the most important metric is the Quality of Experience (QoE) that is just related to the user perception and depends on several human factors. QoE measures the fact if the user likes a service, or not. Therefore QoE should consider all degrading effects such as as blur, jerkiness, blocking, freezes and so on, all effects that depend on the network performance but the relationship with network parameters is usually rather complex. QoE clearly depends on the service and on the performance required to the networks. For instance for YouTube 360p a bandwidth of 1 Mb/s could be sufficient, conversely for YouTube 1080p at least 4 Mb/s is necessary. QoE is often assumed as a subjective network measurement. QoS is used to be measured in terms of Mean Opinion Score (MOS) [22] and it can be evaluated with subjective tests and objective tests. MOS based on subjective tests is generally evaluated by a pool of reviewers that look at the video services and manifest a score either following the quality evolution in time or at the end of the service [15]. Conversely QoE based on objective tests uses software and algorithms tools to quantify the service degradation, for an instance also by using a reference service [22]. Currently some works have showed some relationships between QoE and QoS, for an instance related to the network parameters as throughput, jitter, Round Trip Time, etc [10], and some interesting results can be found in [17], [21], [11], [9]. In particular in [9] an experimental investigation is reported for HD video tests in a wide bandwidth access (20 Mb/s) using both TCP and UDP forwarding. The results show that, also in the presence of wide bandwidth, due to high network delay (RTT), strong throughput reduction can be manifested. In these conditions, for an instance by using Windows XP operating systems for RTT=60 ms a throughput equal to 9 Mb/s was measured that induces a strong degradation on video perception (minimum video bit rate equal to 10 Mb/s). Therefore the curve QoE vs QoS (in terms of either RTT or throughput) shows a step behavior passing from 5 (high QoS) to 1 (poor QoS) with a threshold around (60 ms-9 Mb/s). Conversely it has to be pointed out that QoE does not show any degradation (QoE=5) either in the case of TCP with operating systems as Windows 7 and Linux or by using UDP. It is clear that the relationship between QoE and QoS depends on too many considerations related to all the OSI Layers and in particular to the characteristics of the application. Therefore further investigations are necessary to take into account all the user aspects.

A.3 The MisuraInternet project

In this paragraph we describe the architecture of "MisuraInternet"that is the Italian method of monitoring the QoS (Quality of Service) provided by Internet Service Providers (ISPs) to check the user SLA. This project is based on AGCOM n.244/08/CSP Resolution and the technical aspects are complied with pertinent ETSI standard.

Measurements estimate the throughput of a client-server connection, the latency between the same client-server couple and the packet loss. The analyses of the measurements are made according to the methods that are proposed by ETSI in [4]. The measurement systems are:

- ISP measurements of throughput and QoS, carried out by means of a monitoring network consisting of 20 test points distributed along Italy and specifically placed in major Italian cities (i.e., one test point for each region). This monitoring network works all day every day.
- End user measurements, concerning throughput and QoS, that are carried out directly at the user home, using a open source software Ne.Me.Sys (Network Measurement System) that is developed specifically for this project by FUB.

The aim of the MisuraInternet is the creation of a generic and efficient instrument for the consumer. Every aspect of the project is public and every activity is well detailed on the dedicated web site (www.misurainternet.it).

The whole project is based on three pillars that are benchmarking approach, personal approach, and statistical approach. The aim of the statistical approach is the creation of a network monitoring system that is able to collect sample of the two most sold lines of each operator.

Benchmark values are referred to under test lines, which are located in government buildings and the whole environment is managed by FUB. The electrical characteristics of the lines are predefined and the same for each operator. Such set up guarantees that measures are carried out in the most common conditions, according to mean Italian users local loop length and mean cross-talk value. In this way, all obtained results could represents the benchmark value of each operator, indeed measures are not influenced by the single user access network characteristics.

When a consumer, by means of the software Ne.Me.Sys., measures its own fixed line connection performances, a personal approach is defined. Obtained results have a legal value and allow the consumer to recede the contract without penalties, in case of performances do not match the values that are declared by operator. Furthermore, the consumer could compare the obtained results with benchmark and statistic values.

The third pillar is statistic results. The aim of this part of the project is the development of a instant speed national map, based on the real experience of consumers. All data, incoming from results of certificated software Ne.Me.Sys. (and next release "Ne.Me.Sys Speed Test''), are collected and statistical elaborated. In so doing it is possible to compare the result of consumer measurements with the mean value of each operator profile, considering a specific geographic area.

Such three pillars constitute the main aims of the project. Summarizing there are a monitoring network along the whole country, a software to allow every consumer to measure its own fixed line performance, and a statistic session, within MisuraInternet web site, that concerns data which are collected from real end user cases.

All these information constitute what we call Italian QoS Monitoring Network, indeed data incoming from probes and end-users all suitable integrated and general statistic are computed.

To estimate the QoS the considered ETSI standard suggests to calculate both throughput and latency between a client-server couple by means of the FTP method 3.8.3. The server of the couple has been located into a IXP (Internet eXchange Point), also called NAPs (Neutral Access Points), and the client could be the end user personal computer or a probe located within one of the government building. In order to estimate throughput and latency, an FTP connection is established between IXP and client. More specifically the client has to (down/up)load a pseudorandom file to evaluate throughput. The file size is a very important parameter since it affects the TCP (Transmission Control Protocol) slow start mechanism. The file should be enough long to allow the increase of congestion window, optimizing the TCP file transfer mechanism. To make a measure, each pseudorandom file is downloaded (uploaded) 20 times consecutively.

Involving ISP access networks over all Italian territory, the QoS monitoring infrastructure has been set up. The monitoring system architecture, implemented in trusted environments (Government territorial branch offices, i.e. “Ispettorati Territoriali”), is shown at `https://www.misurainternet.it/architettura.php`.

To realize such infrastructure, each ISP brings into a probe room two ADSL/ADSL2+ connections (access networks based on ADSL2+ is the European most adopted technology in Telco Operators network) according to its own customer base. For each ISP, the most two out sold headline speeds

has to be measured. The probe room is chosen by following two criteria:

1. the distance between the client and the Central Office has to be 1,2 km (such value is the mean Italian distance between consumers and the Central Office);
2. the line has to belong to one of the busier DSLAM (Digital Subscriber Line Multiplexing) in terms of traffic.

The constraint on DSLAM is always respected, because government buildings involved in the project are located in the center of the major Cities, many ADSL/ADSL2+ connections are activated at the same time and DSLAM has always to manage a lot of traffic.

As far as distance between probe room and Central Office is concerned, in many circumstances it could be less than 1,2 km; in these cases, it is possible to artfully extend the line length and the constraint is respected if the upload attenuation equal to 11 dB.

The monitoring system structure is designed to implement FTP based tests between two hosts located in relevant network sections.

Active probes are distributed over Italian territory adopting access networks that can be considered representative of the average conditions in terms of congestion (number of customers belonging to the same DSLAM) and physical parameters (such attenuation or distance from the central office).

FTP servers are located in IXPs, which are the Big Internet gateway for ISP networks. They are physical infrastructures through which ISPs exchange Internet traffic between their networks (autonomous systems). By means of a measurement server located in IXP, it is possible to ensure that each client will measure only the performance related to its own ISP network. This approach has been chosen since allows comparable and reproducible results.

In these two year many results has been obtained in terms of data collected. Two types of results have been reached; one type belongs to category of end-user measures and the other one to category of probe measures.

As far as first type of results are concerned, more than 11000 people have characterized its own network by means of Ne.Me.Sys. and in some cases several users completed measures more than one time, achieving more than one certificate. Indeed when a consumer completes the measures a certificate is released by Ne.Me.Sys.

Now the more than 13000 certificates have been realized. All these data allow to establish the actual speed provided by ISP.

Moreover more than 85000 user have made the registration. In general, if a consumer thinks that the speed of his fixed Internet line is much lower than the speed advertised by his operator, he approaches to the project.

For an instance by considering the 8 Mbps profile, 57% of certificates have a very low RTT (0-20 ms), and only 8% have an high RTT (40-60 ms); furthermore 35% of certificates have a RTT included between 20 and 40 ms. Consequently it is possible to say that for the most diffused bandwidth profiles the RTT is not high and so only the network quality is measured. TCP implementation stack does not influence almost measurement results.

Comparative results are available on line and refreshed every six months. Statistic are calculated aggregating all national data per profile or aggregating data per Region. The user can view all profile on www.misurainternet.it.

As far as statistical values are concerned, it was possible to observe a wide difference between ULL ISPs and Bitstream ISPs, the first operators has its own device (DSLAM) within the Central Office, and their network is directly connected to the DSLAM. The second one ISPs aggregate traffic in a few interconnection points within the network, DSLAM belong to “incumbent” operator, and observed throughput is quite sensitive to traffic congestion.

A.4 Ne.me.sys

Here we report some outcomes carried out in the framework of the field trials of Italian QoS measurements based on the DEL 214 08 recommendation of the Italian Regolamentation Authority for telecommunication (AGCOM), based on a software agent (Nemesys) and the check the quality of ISP by means of active proves located in some particular points of Italy.

To have a reliable description of the user performance to be compared with SLA several measurements are necessary along the time. For such an aim several measurements would be necessary along the day to take into account all the possible variations due to environmental conditions and traffic distribution, but only one day could be not enough since the analysis should require also the week behavior, but also the month and the year ones. It is clear that we cannot propose this boring approach to the user and therefore the measurements have to be limited to a short period. To give some idea we can say that according to AGCOM DEL 244-08 a period on an whole day was chosen for tests and to complete a full day test the period under observation was for a maximum of three days. To made a measure, each pseudorandom file is downloaded (uploaded) 20 times consecutively. More specifically one single test is composed by a single file download/upload, 20 tests constitute a measure. At least one measure per hour during the day is needed. As far as latency is concerned, 10 PING test constitute a measure, also in this case at least one latency measure per hour is need. The ICMP packet is composed of 32 Bytes (Windows OS default value). All measures are statistically elaborated, the 5Th (called minimum bandwidth) and the 95th (called maximum bandwidth) percentile, as well as the average value and the standard deviation are calculated for each test. Starting from tests, also Packet Loss Ratio and Unsuccessful Data Transmission rate are calculated. Packet Loss Ratio represents the ratio between the number of not replied PING commands and the total number of sent PING commands. Unsuccessful Data Transmission rate is the ratio between unsuccessful data transmissions and the total number of data transmission attempt in a specified time period.

The method adopted by Ne.me.sys allows the user to monitor his wireline broadband access by means of its PC. Therefore it is necessary a method to avoid that other devices (PCs, tablet, smart TV) connected to the broadband access disturb the measurement. Therefore to guarantee the reliability of the measurement Ne.Me.Sys adopted the following conditions:

1. only the user PC, that is involved to measure, has to be connected to the LAN, so the other hosts have to be disconnected during the measure;
2. the traffic, that does not belong to the measure traffic (namely alien traffic), is denied;
3. the traffic over wi-fi connection is not allowed (personal computer has to be connected using a cable);
4. the CPU usage has to be under a predetermined threshold (< 85%);

5. the RAM usage has to be under a predetermined threshold (< 95%) and the RAM of used PC has to be at least 12 MBytes.

In the second release of the software, many difficulties occurred during user measures have been overcome, and some constraints have been relaxed. One of the most stringent constraint concerning the alien traffic; indeed, in the first release, the user web-surfing was forbidden, but sometimes some applications automatically connected themselves to Internet, without the user permission. In these case Ne.Me.Sys. did not allow the measure. In the second release a small amount of alien traffic is permitted; in this way the end user can complete more easily the measure cycle and the measures are always valid. In particular a threshold has been introduced: the alien traffic has to be less than 10% of the measure traffic.

References

[1] `http://dev.maxmind.com/geoip/`.

[2] G. Almes, S. Kalidindi, and M. Zekauskas. A One-way Delay Metric for IPPM. RFC 2679 (Proposed Standard), Sept. 1999.

[3] G. Almes, S. Kalidindi, and M. Zekauskas. A One-way Packet Loss Metric for IPPM. RFC 2680 (Proposed Standard), Sept. 1999.

[4] G. Almes, S. Kalidindi, and M. Zekauskas. A Round-trip Delay Metric for IPPM. RFC 2681 (Proposed Standard), Sept. 1999.

[5] M. Bagnulo, T. Burbridge, S. Crawford, P. Eardley, and A. Morton. A Registry for Commonly Used Metrics. IETF Internet-Draft draft-bagnulo-ippm-new-registry-00 (work in progress), Jan. 2013.

[6] K. Claffy, Y. Hyun, K. Keys, M. Fomenkov, and D. Krioukov. Internet Mapping: From Art to Science. In *Proceedings of the 2009 Cybersecurity Applications & Technology Conference for Homeland Security*, CATCH '09, pages 205--211, Washington, DC, USA, 2009. IEEE Computer Society.

[7] B. Claise. Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information. RFC 5101 (Proposed Standard), Jan. 2008.

[8] C. Demichelis and P. Chimento. IP Packet Delay Variation Metric for IP Performance Metrics (IPPM). RFC 3393 (Proposed Standard), Nov. 2002.

[9] A. V. et al. QoE and QoS Comparison in an Anycast Digital Television Platform Operating on Passive Optical Network. In *Proceedings of the 2012 International Telecommunications Network Strategy and Planning Symposium*, NETWORKS '11, pages 1--6. IEEE, 2012.

[10] P. B. et al. Monitoring of the user quality of service: Network architecture for measurements, role of the user operating system with consequences for optical accesses. In *Proceedings of the 2011 Optical Network Design and Modeling Conference*, ONDM '11, pages 1--5. IEEE, 2011.

[11] M. Fiendler, T. Hassfeld, and P. Tran-Gia. A generic quantitative relationship between quality of experience and quality of service. *IEEE Networks*, 24(2):36--41, Apr. 2010.

[12] A. Hanemann, J. W. Boote, E. L. Boyd, J. Durand, L. Kudarimoti, R. Lapacz, D. M. Swany, S. Trocha, and J. Zurawski. Perfsonar: a service oriented architecture for multi-domain network monitoring. In *Proceedings of the Third international conference on Service-Oriented Computing*, ICSOC'05, pages 241--254, Berlin, Heidelberg, 2005. Springer-Verlag.

[13] IEEE. IEEE 1588-2008, July 2008.

[14] C. Kreibich, N. Weaver, B. Nechaev, and V. Paxson. Netalyzr: illuminating the edge network. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, IMC '10, pages 246--259, New York, NY, USA, 2010. ACM.

[15] F. Matera, L. Rea, V. Baroncini, and G. Gaudino. Comparison between objective and subjective measurements of quality of service over an optical wide area network. *European Transactions on Telecommunications*, 19(3):233--245, Apr. 2008.

[16] D. Mills, J. Martin, J. Burbank, and W. Kasch. Network Time Protocol Version 4: Protocol and Algorithms Specification. RFC 5905 (Proposed Standard), June 2010.

[17] R. K. P. Mok, E. W. W. Chan, and R. K. Chang. Measuring the Quality of Experience of HTTP Video Streaming. In *Proceedings of the 2011 Integrated Network Management International Symposium*, IM '11, pages 485--492. IEEE, 2011.

[18] I. A. numbers Authority. IP Flow Information Export (IPFIX) Entities. `http://www.iana.org/assignments/ipfix/ipfix.xml`, Jan. 2013.

[19] RIPE. Atlas Project. `http://atlas.ripe.net/`, January 2013.

[20] RIPE. Test Traffic Measurement Service. `http://www.ripe.net/ttm/`, January 2013.

[21] K. Seshadrinathan, R. Seshadrinathan, A. Bovik, and L. K. Cormack. Study of Subjective and Objective Quality Assessment of Video. *IEEE Transaction on Image Processing*, 19(6):1427--1441, June 2010.

[22] R. Stankiewicz, P. Cholda, and A. Jajszczyk. Qox: What is really? *IEEE Communication Magazine*, 49(4):148--158, Apr. 2011.

[23] S. Sundaresan, W. de Donato, N. Feamster, R. Teixeira, S. Crawford, and A. Pescapè. Broadband internet performance: a view from the gateway. In *Proceedings of the ACM SIGCOMM 2011 conference*, SIGCOMM '11, pages 134--145, New York, NY, USA, 2011. ACM.

[24] B. Trammell and E. Boschi. Bidirectional Flow Export Using IP Flow Information Export (IPFIX). RFC 5103 (Proposed Standard), Jan. 2008.

[25] University of Oregon. Route Views. `http://www.routeviews.org/`, January 2005.