Towards In-Network Clouds in Future Internet

Alex Galis[1], Stuart Clayman[1], Laurent Lefevre[2], Andreas Fischer[3], Hermann de Meer[3], Javier Rubio-Loyola[4], Joan Serrat[5], and Steven Davy[6]

[1] University College London, United Kingdom, {a.galis,s.clayman}@ee.ucl.ac.uk
[2] INRIA, France, laurent.lefevre@ens-lyon.fr
[3] University of Passau, Germany {andreas.fischer,hermann.demeer}@uni-passau.de
[4] CINVESTAV Tamaulipas, Mexico, jrubio@tamps.cinvestav.mx
[5] Universitat Politècnica de Catalunya, Spain, serrat@nmg.upc.edu
[6] Waterford Institute of Technology, Ireland, sdavy@tssg.org

Abstract. One of the key aspect fundamentally missing from the current Internet infrastructure is an advanced service networking platform and facilities, which take advantage of flexible sharing of available connectivity, computation, and storage resources. This paper aims to explore the architectural co-existence of new and legacy services and networks, via virtualisation of connectivity and computation resources and self-management capabilities, by fully integrating networking with cloud computing in order to create In-Network Clouds. It also presents the designs and experiments with a number of In-Network Clouds platforms, which have the aim to create a flexible environment for autonomic deployment and management of virtual networks and services as experimented with and validated on large-scale testbeds.

Keywords: In-Network Clouds, Virtualisation of Resources, Self-Management, Service plane, Orchestration plane and Knowledge plane.

1 Introduction

The current Internet has been founded on a basic architectural premise, that is: a simple network service can be used as a universal means to interconnect both dumb and intelligent end systems. The simplicity of the current Internet has pushed complexity into the endpoints, and has allowed impressive scale in terms of inter-connected devices. However, while the scale has not yet reached its limits [1, 2], the growth of functionality and the growth of size have both slowed down and may soon reach both its architectural capability and capacity limits. Internet applications increasingly require a combination of capabilities from traditionally separate technology domains to deliver the flexibility and dependability demanded by users. Internet use is expected to grow massively over the next few years with an order of magnitude more Internet services, the interconnection of smart objects from the Internet of Things, and the integration of increasingly demanding enterprise and societal applications.

The Future Internet research and development trends are covering the main focus of the current Internet, which is connectivity, routing, and naming as well as defining

J. Domingue et al. (Eds.): Future Internet Assembly, LNCS 6656, pp. 19–33, 2011.

and design of all levels of interfaces for Services and for networks' and services' resources. As such, the Future Internet covers the complete management and full lifecycle of applications, services, networks and infrastructures that are primarily constructed by recombining existing elements in new and creative ways.

The aspects which are fundamentally missing from the current Internet infrastructure, include the advanced service networking platforms and facilities, which take advantage of flexible sharing of available resources (e.g. connectivity, computation, and storage resources).

This paper aims to explore the architectural co-existence of new and legacy services and networks, via virtualisation of resources and self-management capabilities, by fully integrating networking [4, 8, 10, 15] with cloud computing [6, 7, 9] in order to produce In-Network Clouds. It also presents the designs and experiments with a number of In-Network Clouds platforms [9, 10], which have the aim to create a flexible environment for autonomic deployment and management of virtual networks and services as experimented with and validated on large-scale testbeds [3].

2 Designs for In-Network Clouds

Due to the existence of multiple stakeholders with conflicting goals and policies, modifications to the existing Internet are now limited to simple incremental updates and deployment of new technology is next to impossible and very costly. In-Network clouds have been proposed to bypass this ossification as a diversifying attribute of the future inter-networking and inter-servicing paradigm. By allowing multiple heterogeneous network and service architectures to cohabit on a shared physical substrate, In-Network virtualisation provides flexibility, promotes diversity, and promises security and increased manageability.

We define In-Network clouds as an integral part of the differentiated Future Internet architecture, which supports multiple computing clouds from different service providers operating on coexisting heterogeneous virtual networks and sharing a common physical substrate of communication nodes and servers managed by multiple infrastructure providers. By decoupling service providers from infrastructure providers and by integrating computing clouds with virtual networks the In-Network clouds introduce flexibility for change.

In-Network Network and Service Clouds can be represented by a number of distributed management systems described with the help of five abstractions: Virtualisation Plane (VP), Management Plane (MP), Knowledge Plane (KP), Service Plane (SP), and Orchestration Plane (OP) as depicted in Fig. 1.

These planes are new higher-level artefacts, used to make the Future Internet of Services more intelligent, with embedded management functionality. At a logical level, the VMKSO planes gather observations, constraints and assertions, and apply rules to these in order to initiate proper reactions and responses. At the physical level, they are embedded and execute on network hosts, devices, attachments, and servers

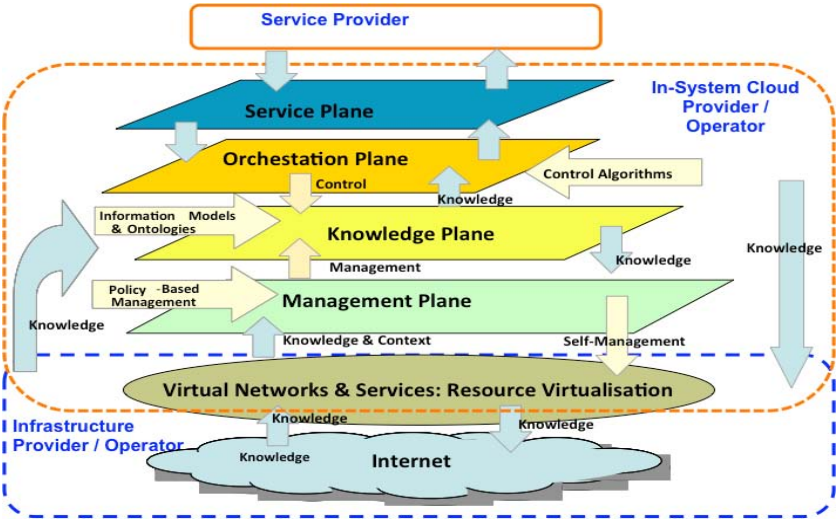

Fig. 1. In-Network Cloud Resources

within the network. Together these distributed systems form a software-driven network control infrastructure that will run on top of all current networks (i.e. fixed, wireless, and mobile networks) and service physical infrastructures in order to provide an autonomic virtual resource overlay.

2.1 Service Plane Overview

The Service Plane (SP) consists of functions for the automatic (re-) deployment of new management services, protocols, as well as resource-facing and end-user facing services. It includes the enablers that allow code to be executed on the network entities. The safe and controlled deployment of new code enables new services to be activated on-demand. This approach has the following advantages:

- Service deployment takes place automatically and allows a significant number of new services to be offered on demand;
- It offers new, flexible ways to configure network entities that are not based on strict configuration sets;
- Services that are not used can be automatically disabled. These services can be enabled again on-demand, in case they are needed;
- It eases the deployment of network-wide protocol stacks and management services;
- It enables secure but controlled execution environments;
- It allows an infrastructure that is aware of the impact on the existing services of a new deployment;
- It allows optimal resource utilization for the new services and the system.

2.2 Orchestration Plane Overview

The purpose of the Orchestration Plane is to coordinate the actions of multiple autonomic management systems in order to ensure their convergence to fulfil applicable business goals and policies. It supervises and it integrates all other planes' behaviour ensuring integrity of the Future Internet management operations. The Orchestration Plane can be thought of as a control framework into which any number of components can be plugged into, in order to achieve the required functionality. These components could have direct interworking with control algorithms, situated in the control plane of the Internet (i.e. to provide real time reaction), and interworking with other management functions (i.e. to provide near real time reaction).

The Orchestration Plane is made up of one or more Autonomic Management Systems (AMS), one or more Distributed Orchestration Components (DOC), and a dynamic knowledge base consisting of a set of information models and ontologies and appropriate mapping logic and buses. Each AMS represents an administrative and/or organisational boundary that is responsible for managing a set of devices, subnetworks, or networks using a common set of policies and knowledge. The Orchestration Plane acts as control workflow for all AMS ensuring bootstrapping, initialisation, dynamic reconfiguration, adaptation and contextualisation, optimisation, organisation, and closing down of an AMS. It also controls the sequence and conditions in which one AMS invokes other AMS in order to realize some useful function (i.e., an orchestration is the pattern of interactions between AMS). An AMS collects appropriate monitoring information from the virtual and non-virtual devices and services that it is managing, and makes appropriate decisions for the resources and services that it governs, either by itself (if its governance mode is individual) or in collaboration with other AMS (if its governance mode is distributed or collaborative), as explained in the next section. The OP is build on the concepts identified in [13], however it differs in several essential ways:

- Virtual resources and services are used.
- Service Lifecycle management is introduced.
- The traditional management plane is augmented with a narrow knowledge plane, consisting of models and ontologies, to provide increased analysis and inference capabilities.
- Federation, negotiation, distribution, and other key framework services are packaged in a distributed component that simplifies and directs the application of those framework services to the system.

The Distributed Orchestration Component (DOC) provides a set of framework network services. Framework services provide a common infrastructure that enables all components in the system under the scope of the Orchestration Plane to have plug_and_play and unplug_and_play behaviour. Applications compliant with these framework services share common security, metadata, administration, and management services. The DOC enables the following functions across the orchestration plane: federation, negotiation, distribution and governance. The federation functionality of the OP is represented by the composition/decomposition of networks & services under different domains. Since each domain may have different SLAs, security and

administrative policies, a federation function would trigger a negotiation between domains and the re-deployment of service components in the case that the new policies and high level goals of the domain are not compatible with some of the deployed services. The negotiation functionality of the OP enables separate domains to reach composition/ decomposition agreements and to form SLAs for deployable services. The distribution functionality of the OP provides communication and control services that enable management tasks to be split into parts that run on multiple AMSs within the same domain. The distribution function controls the deployment of AMSs and their components. The governance functionality of the OP monitors the consistency of the AMSs' actions, it enforces the high level policies and SLAs defined by the DOCs and it triggers for federation, negotiation and distribution tasks upon noncompliance.

The OP is also supervising the optimisation and the distribution of knowledge within the Knowledge Plane to ensure that the required knowledge is available in the proper place at the proper time. This implies that the Orchestration Plane may use very local knowledge to deserve a real time control as well as a more global knowledge to manage some long-term processes like planning.

2.3 Virtualisation Plane Overview

Virtualisation hides the physical characteristics [14, 16] of the computing and networking resources being used, from its applications and users. This paper uses system virtualisation to provide virtual services and resources. System virtualisation separates an operating system from its underlying hardware resources; resource virtualisation abstracts physical resources into manageable units of functionality. For example, a single physical resource can appear as multiple virtual resources (e.g., the concept of a virtual router, where a single physical router can support multiple independent routing processes by assigning different internal resources to each routing process); alternatively, multiple physical resources can appear as a single physical resource (e.g., when multiple switches are "stacked" so that the number of switch ports increases, but the set of stacked switches appears as a single virtual switch that is managed as a single unit). Virtualisation enables optimisation of resource utilisation. However, this optimisation is confined to inflexible configurations within a single administrative domain. This paper extends contemporary virtualisation approaches and aims at building an infrastructure in which virtual machines can be dynamically relocated to any physical node or server regardless of location, network, and storage configurations and of administrative domain.

The virtualisation plane consists of software mechanisms to abstract physical resources into appropriate sets of virtual resources that can be organised by the Orchestration Plane to form components (e.g., increased storage or memory), devices (e.g., a switch with more ports), or even networks. The organisation is done in order to realise a certain business goal or service requirement. Two dedicated interfaces are needed: the vSPI and the vCPI (Virtualisation System Programming Interface and Virtualisation Component Programming Interface, respectively). A set of control loops is formed using the vSPI and the vCPI, as shown in Figure 2.

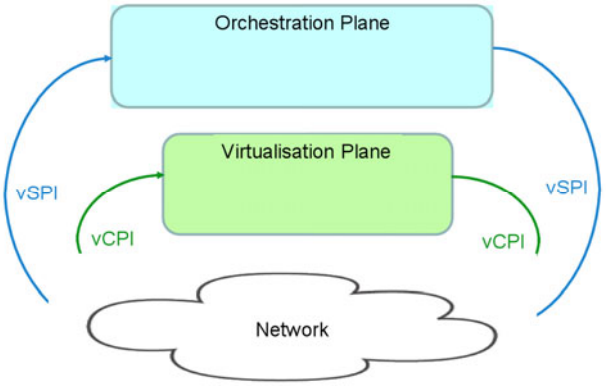

Fig. 2. Virtualisation Control Loop

Virtualisation System Programmability Interface (vSPI). The vSPI is used to enable the Orchestration Plane (and implicitly the AMS and DOC that are part of a given Orchestration Plane) to govern virtual resources, and to construct virtual services and networks that meet stated business goals having specified service requirements. The vSPI contains the "macro-view" of the virtual resources that a particular Orchestration Plane governs, and is responsible for orchestrating groups of virtual resources in response to changing user needs, business requirements, and environmental conditions. The low-level configuration (i.e., the "micro-view") of a virtual resource is provided by the vCPI. For example, the vSPI is responsible for informing the AMS that a particular virtual resource is ready for use, whereas the vCPI is responsible for informing the AMS that a particular virtual resource has been successfully reconfigured. The governance is performed by the set of AMS that are responsible for managing each component or set of components; each AMS uses the vSPI to express its needs and usage of the set of virtual resources to which it has access. The vSPI is responsible for determining what portion of a component (i.e., set of virtual resources) is allocated to a given task. This means that all or part of a virtual resource can be used for each task, providing an optimised partitioning of virtual resources according to business need, priority and other requirements. Composite virtual services can thus be constructed using all or part of the virtual resources provided by each physical resource.

Virtualisation Component Programming Interface (vCPI). Each physical resource has an associated and distinct vCPI. The vCPI is fulfilling two main functions: monitoring and management. The management functionality enables the AMS to manage the physical resource, and to request virtual resources to be constructed from that physical resource by the vCPI of the Virtualisation Plane. The AMS sends abstract (i.e., device-independent) commands via the vCPI, which are translated into device- and vendor-specific commands that reconfigure the physical resource (if necessary) and manage the virtual resources provided by that physical resource. The vCPI also provides monitoring information from the virtual resources back to the AMS that

controls that physical resource. Note that the AMS is responsible for obtaining management data describing the physical resource. The vCPI is responsible for providing dynamic management data to its governing AMS that states how many virtual resources are currently instantiated, and how many additional virtual resources of what type can be supported.

2.4 Knowledge Plane Overview

The Knowledge Plane was proposed by Clark et al. [1] as a new dimension to a network architecture, contrasting with the data and control planes; its purpose is to provide knowledge and expertise to enable the network to be self-monitoring, self-analysing, self-diagnosing and self-maintaining. A narrow functionality Knowledge Plane (KP), consisting of context data structured in information models and ontologies, which provide increased analysis and inference capabilities is the basis for this paper. The KP brings together widely distributed data collection, wide availability of that data, and sophisticated and adaptive processing or KP functions, within a unifying structure. Knowledge extracted from information/data models forms facts. Knowledge extracted from ontologies is used to augment the facts, so that they can be reasoned about. Hence, the combination of model and ontology knowledge forms a universal lexicon, which is then used to transform received data into a common form that enables it to be managed. The KP provides information and context services as follows:

- information-life cycle management, which includes storage, aggregation, transformations, updates, distribution of information;
- triggers for the purpose of contextualisation of management systems (supported by the context model of the information model);
- support for robustness enabling the KP to continue to function as best possible, even under incorrect or incomplete behaviour of the network itself;
- support of virtual networks and virtual system resources in their needs for local control, while enabling them to cooperate for mutual benefit in more effective network management.

The goal of making the control functions of Networks context-aware is therefore essential in guaranteeing both a degree of self-management and adaptation as well as supporting context-aware communications that efficiently exploit the available network resources. Furthermore, context-aware networking enables new types of applications and services in the Future Internet.

Context Information Services. The Context Information Service Platform (CISP), within the KP, has the role of managing the context information, including its distribution to context clients/consumers. Context clients are context-aware services, either user-facing services or network management services, which make use of or/and adapt themselves to context information. Network services are described as the services provided by a number of functional entities (FEs), and one of the objectives of

this description is to investigate how the different FEs can be made context-aware, i.e. act as context clients. The presence of CISP functionality helps to make the interactions between the different context sources and context clients simpler and more efficient. It acts as a mediating unit and reduces the numbers of interactions and the overhead control traffic. CISP is realised by four basic context-specific functional entities: (i) the Context Executive (CE) Module which interfaces with other entities/context clients, (ii) the Context Processing (CP) Module which implements the core internal operations related to the context processing, (iii) the Context Information Base (CIB) which acts as a context repository, and (iv) the Context Flow Controller (CFC) which performs context flow optimization activities (see Fig. 3).

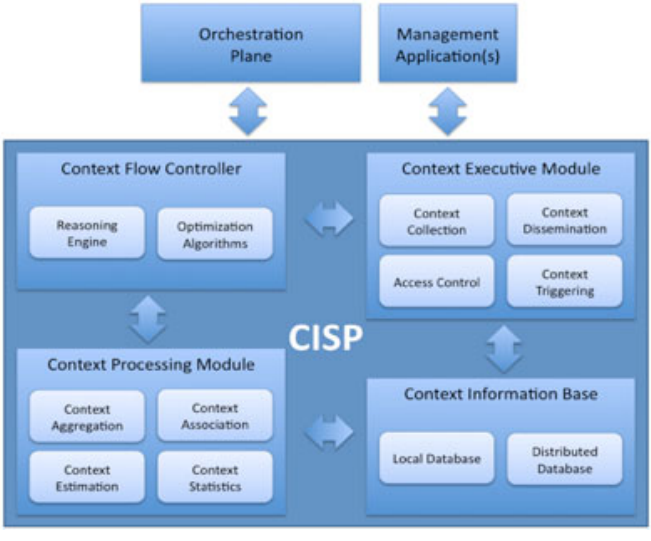

Fig. 3. Context Information Service Platform

The Context Executive Module (CE) is introduced to meet the requirements of creating a gateway into the CISP architecture and deals with indexing, registering, authorising and resolving context names into context or location addresses. Furthermore, the CE meets the requirements of context collection, context dissemination, interfaces with the Context Information Base and supports for access control. The Context Processing Module (CP) is responsible for the context management, including context aggregation, estimation and creation of appropriate context associations between clients and sources. The context association allows the CISP to decide where a specific context should be stored. Furthermore, the CP collects statistics about context usage. We note that these context statistics should be optimised in terms of memory usage for scalability purposes. In practice, the CP creates meta-context from context using mechanisms that exploit the business requirements, other forms of context and context usage statistics. The meta-context carries information that supports better the self-management functionalities of the context-aware applications. In general, the CE module is responsible for the communication of the CISP with the other management

applications/components and the CP module for the optimisation of the context information. The Context Information Base (CIB) provides flexible storage capabilities, in support of the Context Executive and Context Processor modules. Context is distributed and replicated within the domain in order to improve the performance of context lookups. The CIB stores information according to a common ontology. The Context Flow Controller configures the Context Processing and Context Executive Modules based on the requirements of the Management Application and the general guidelines from the Orchestration Plane. These configuration settings are enabling certain behaviours in terms of context flow optimization with respect to these guidelines.

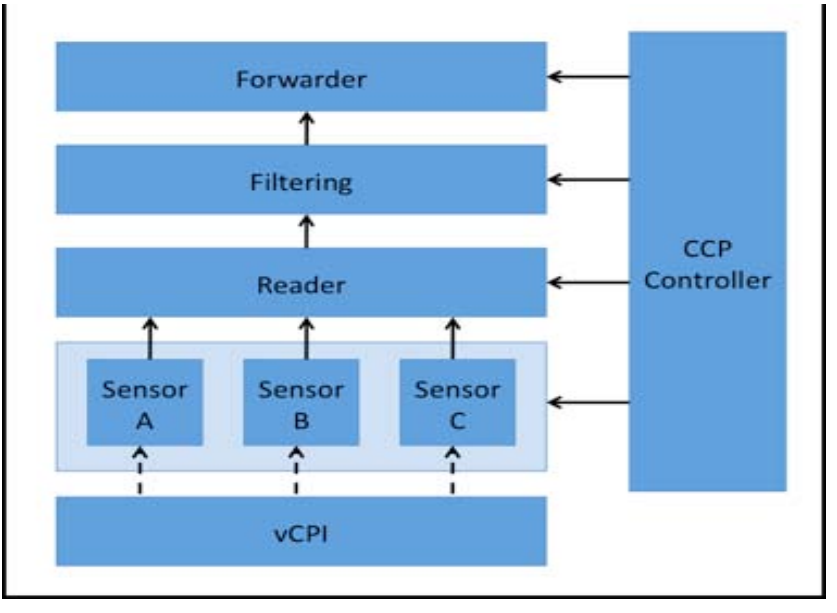

Fig. 4. The Context Collection Component

Context Collection Points. The Context Collection Points (CCP) act as sources of information: they monitor hardware and software for their state, present their capabilities, or collect configuration parameters. A monitoring mechanism and framework was developed to gather measurements from relevant physical and virtual resources and CCPs for use within the CISP. It also offers control mechanisms of the relevant probes and it also controls the context aggregation points (CAP). Such a monitoring framework has to have a minimal runtime foot–print, avoiding to be intrusive, so as not to adversely affect the performance of the network itself or the running management elements. The CISP Monitoring System supports three types of monitoring queries to an CCP: (i) 1-time queries, which collect information that can be considered static, e.g., the number of CPUs, (ii) N-time queries, which collect information periodically, and (iii) continuous queries that monitor information in an on-going manner. CCPs should be located near the corresponding sources of information in

order to reduce management overhead. Filtering rules based on accuracy objectives should be applied at the CCPs, especially for the N-time and continuous queries, for the same reason. Furthermore, the CCPs should not be many hops away from the corresponding context aggregation point (CAP). Fig. 4 shows the structure of a CCP, which we have designed and implemented, consisting of 5 main components: the sensors, a reader, a filter, a forwarder and a CCP controller. These are described below.

The *sensors* can retrieve any information required. This can include common operations such as getting the state of a server with its CPU or memory usage, getting the state of a network interface by collecting the number of packets and number of bytes coming in and out, or getting the state of disks on a system presenting the total volume, free space, and used space. In our implementation, each sensor runs in its own thread allowing each one to collect data at different rates and also having the ability to turn them on and off if they are not needed. We note that the monitoring information retrieval is handled by the Virtualisation Plane.

The *reader* collects the raw measurement data from all of the sensors of a CCP. The collection can be done at a regular interval or as an event from the sensor itself. The reader collects data from many sensors and converts the raw data into a common measurement object used in the CISP Monitoring framework. The format contains meta-data about the sensor and the time of day, and it contains the retrieved data from the sensor.

The *filter* takes measurements from the reader and can filter them out before they are sent on to the forwarder. Using this mechanism it is possible to reduce the volume of measurements from the CCP by only sending values that are significantly different from previous measurements. For example, if a 5% filter is set, then only measurements that differ from the previous measurement by more than 5% will be passed on. By using filtering in this way, the CCP reduces the load on the network. In our case, the filtering percentage matches the accuracy objective of the management application requesting the information.

The *forwarder* sends the measurements onto the network. The common measurement object is encoded into a network amenable measurement format.

The CCP Controller controls and manages the other CCP components. It controls (i) the lifecycle of the sensors, being able to turn them on and off, and to set the rate at which they collect data; (ii) the filtering process, by changing the filter or adapting an existing filter; (iii) the forwarder, by changing the attributes of the network (such as IP address and port) that the ICP is connected to.

The vCPI supports the extension with additional functions, implicitly allowing the creation of other types of sensors, and thus helping the CCP to get more information. Also various sensors, which can measure attributes from CPU, memory, and network components of a server host, were created. We can also measure the same attributes of virtualised hosts by interacting with a hypervisor to collect these values. Finally, there are sensors that can send emulated measurements. These are useful for testing and evaluation purposes, with one example being an emulated response time, which we use in our experiments.

2.5 Management Plane Overview

The Management Plane is a basic building block of the infrastructure, which governs the physical and virtual resources, is responsible for the optimal placement and continuous migration of virtual routers into hosts (i.e., physical nodes and servers) subject to constraints determined by the Orchestration Plane. The Management Plane is designed to meet the following functionality:

- Embedded (Inside) Network functions: The majority of management functionality should be embedded in the network and it is abstracted from the human activities. As such the Management Plane components will run on execution environments supported by the virtual networks and systems, which run on top of all current networks (i.e. fixed, wireless and mobile networks) and service physical infrastructures.
- Aware and Self-aware functions: It monitors the network and operational context as well as internal operational network state in order to assess if the network current behaviour serve its service purposes.
- Adaptive and Self-adaptive functions: It triggers changes in network operations (state, configurations, functions) as a result of the changes in network and service context.
- Automatic self-functions: It enables self-control (i.e. self-FCAPS, self-*) of its internal network operations, functions and state. It also bootstraps itself and it operates without manual external intervention. Only manual/external input is provided in the setting-up of the business goals.
- Extensibility functions: It adds new functions without disturbing the rest of the system (Plug-and-Play / Unplug_and_Play / Dynamic programmability of management functions & services).
- System functions: Minimise life-cycle network operations' costs and minimise energy footprint.

In addition the Management Plane, as it governs all virtual resources, is responsible for the optimal placement and continuous migration of virtual routers into hosts (i.e. physical nodes and servers) subject to constraints determined by the Orchestration Plane.

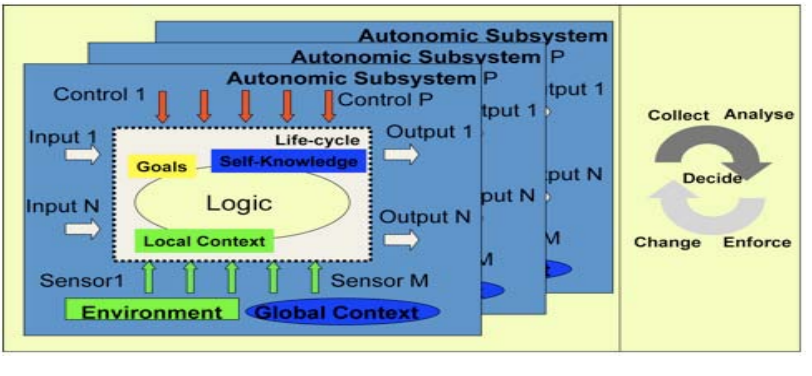

Fig. 5. Autonomic Control Loops

The Management Plane consists of Autonomic Management Systems (AMS). AMS is an infrastructure that manages a particular network domain, which may be an Autonomous System (AS). An AMS implements its own control loops, consisting of context collection, analysis, decision-making and decision enforcement. Each AMS includes interfaces to a dedicated set of models and ontologies and interfaces to one or more Distributed Orchestration Components (DOC), which manage the interoperation of two or more AMS domains. Mapping logic enables the data stored in models to be transformed into knowledge and combined with knowledge stored in ontologies to provide a context-sensitive assessment of the operation of one or more virtual resources. The AMS communicate through sets of interfaces that: (i) enable management service deployment and configuration (i.e., the ANPI and vSPI interfaces), (ii) manipulate physical and virtual resources (i.e., the vCPI interface).

The AMS are design to follow the autonomic control loops depicted in Fig. 5. The AMS is designed to be federated, enabling different AMS that are dedicated to govern different types of devices, resources, and services, to be combined. In order to support this, each AMS uses the models and ontologies to provide a standard set of capabilities that can be advertised and used by other AMS. The capabilities of an AMS can be offered for use to other AMS through intelligent negotiations (e.g., pre-defined agreements, auctioning, bargaining and other mechanisms). An AMS collects appropriate monitoring information from the virtual resources that is managing and makes appropriate decisions for the resources and management services that it governs, either by itself (if its governance mode is individual) or in collaboration with other AMS (if its governance mode is distributed or collaborative).

Since the AMS implement their own control loops, they can have their own goals. However, their goals should be harmonised to the high-level goals coming from the DOC that is responsible for each particular AMS. Each DOC is responsible for a set of AMS that form a network domain, called Orchestrated Domain (OD). An OD may belong to a single organisation that has the same high-level goals. We note that the entry point for the high-level goals is the Orchestration Plane. For example, a set of AMS may re-establish a local service in case of failure without interacting with the Orchestration Plane. However, this new establishment should follow the same guidelines that this local service used to follow. So, there is a significant difference between management and orchestration. Orchestration, actually, harmonises the different management components to one or more common goals.

3 Realisation: In-Network Cloud Functionality

A set of integrated service-centric platforms and supporting systems have been developed and issued as open source [10], which aims to create a highly open and flexible environment for In-Network Clouds in Future Internet. They are briefly described herewith. Full design and implementation of all software platforms are presented in [10].

- vCPI (Virtual Component Programming Interface is the VP's main component dealing with the heterogeneity of virtual resources and enabling programmability of network elements In each physical node there is an embedded vCPI, which is aware of the structure of the virtual resources, which are hosted in the physical node.

- CISP (Context Information Service Platform) is the KP's main component supported by a distributed monitoring platform for resources & components. CISP has the role of managing the context information, including its distribution to context clients/consumers.
- ANPI (Autonomic Network Programming Interface) is the SP's main component that enables large-scale autonomic services deployment on virtual networks.
- MBT (Model-Based Translator) platform, part of the KP, which takes configuration files compliant with an Information Model and translates them to device specific commands.
- LATTICE Monitoring Framework, also part of the KP, provides functionality to add powerful and flexible monitoring facilities to system clouds (virtualisation of networks and services). Lattice has a minimal runtime footprint and is not intrusive, so as not to adversely affect the performance of the system itself or any running applications. The monitoring functionality can be built up of various components provided by the framework, so creating a bespoke monitoring sub-system. The framework provides data sources, data consumers, and a control strategy. In a large distributed system there may be hundreds or thousands of measurement probes, which can generate data.
- APE (Autonomic Policy-based Engine), a component of the MP, supports context-aware policy-driven decisions for management and orchestration activities.
- XINA is a modular scalable platform that belong to the CISP and enables the deployment, control and management of programmable or active sessions over virtual entities, such as servers and routers.
- RNM (Reasoning and Negotiation Module), a core element of the KP, which mediates and negotiates between separate federated domains.

These In-Network Cloud platforms were integrated and validated on 2 testbeds enabling experimentation with thousands of virtual machines: V^3 – UCL's Experimental Testbed located in London consisting of 80 cores with a dedicated 10 Gbits/s infrastructure and Grid5000 - an Experimental testbed located in France consisting of 5000 cores and linked by a dedicated 10 Gbits/s infrastructure. Validation and performance analysis are fully described in [13]. Demonstrations are available at: http://clayfour.ee.ucl.ac.uk/demos/ and they are used for:

- Autonomic deployment of large-scale virtual networks (In-Network Cloud Provisioning);
- Self – management of virtual networks (In-Network Cloud Management);
- Autonomic service provisioning on In-Network Clouds (Service Computing Clouds).

4 Conclusion

This work has presented the design of an open software networked infrastructure (In-Network Cloud) that enables the composition of fast and guaranteed services in an efficient manner, and the execution of these services in an adaptive way taking into

account better shared network and service resources provided by an virtualisation environment. We have also described the management architectural and system model for our Future Internet, which were described with the help of five abstractions and distributed systems – the OSKMV planes: Virtualisation Plane (VP), Management Plane (MP), Knowledge Plane (KP), Service Plane (SP) and Orchestration Plane (OP). The resulting software-driven control network infrastructure was fully exercised and relevant analysis on network virtualisation and service deployments were carried out on a large-scale testbed.

Virtualising physical network and server resources has served two purposes: Managing the heterogeneity through introduction of homogeneous virtual resources and enabling programmability of the network elements. The flexibility gained through this approach helps to adapt the network dynamically to both unforeseen and predictable changes in the network. A vital component of such a virtualisation approach is a common management and monitoring interface of virtualised resources. Such an interface has exported management and monitoring functions that allow management components to control the virtual resources in a very fine-grained way through a single, well defined interface. By enabling such fine-grained control, this interface can then form the basis for new types of applications and services in the Future Internet.

Acknowledgments. This work was partially undertaken in the context of the FP7-EU Autonomic Internet [10] and the RESERVOIR [9] research projects, which were funded by the Commission of the European Union. We also acknowledge the support of the Spanish Ministerio de Innovación grant TEC2009-14598-C02-02.

References

1. Clark, D., et al.: NewArch: Future Generation Internet Architecture, http://www.isi.edu/newarch/
2. Galis, A., et al.: Management and Service-aware Networking Architectures (MANA) for Future Internet Position Paper: System Functions, Capabilities and Requirements. Invited paper IEEE ChinaCom09 26-28, Xi'an, China (August 2009), http://www.chinacom.org/2009/index.html
3. Rubio-Loyola, J., et al.: Platforms and Software Systems for an Autonomic Internet. IEEE Globecom 2010; 6-10 Dec., Miami, USA (2010)
4. Galis, A., et al.: Management Architecture and Systems for Future Internet Networks. In: Towards the Future Internet, IOS Press, Amsterdam (2009)
5. Chapman, C., et al.: Software Architecture Definition for On-demand Cloud Provisioning. ACM HPDC, 21-25, Chicago hpdc2010.eecs.northwestern.edu (June 2010)
6. Rochwerger, B., et al.: An Architecture for Federated Cloud Computing. In: Cloud Computing, Wiley, Chichester (2010)
7. Chapman, C., et al.: Elastic Service Management in Computational Clouds. 12th IEEE/IFIP NOMS2010/CloudMan 2010 19-23 April, Osaka (2010) http://cloudman2010.lncc.br/

8. Clayman, S., et al.: Monitoring Virtual Networks with Lattice. NOMS2010/ManFI 2010-Management of Future Internet 2010; 19-23 April, Osaka, Japan (2010), http://www.manfi.org/2010/
9. RESERVOIR project, http://www.reservoir-fp7.eu
10. AutoI project http://ist-autoi.eu
11. Clark, D., Partridge, C., Ramming, J.C.: and, J. T. Wroclawski "A Knowledge Plane for the internet". In: Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols For Computer Communications (Karlsruhe, Germany, SIGCOMM '03, Karlsruhe, Germany, August 25–29, 2003, pp. 3–10. ACM, New York (2003)
12. Jennings, B., Van Der Meer, S., Balasubramaniam, S., Botvich, D., Foghlu, M., Donnelly, W., Strassner, J.: Towards autonomic management of communications networks. IEEE Communications Magazine 45(10), 112–121 (2007)
13. Deliverable D6.3 Final Results AutoI Approach http://ist-autoi.eu/
14. Mosharaf, N.M., Chowdhury, K., Boutaba, R., Cheriton, D.R.: A Survey of Network Virtualization. Journal Computer Networks: The International Journal of Computer and Telecommunications Networking 54(5) (2010)
15. Galis, A., Denazis, S., Bassi, A., Berl, A., Fischer, A., de Meer, H., Strassner, J., Davy, S., Macedo, D., Pujolle, G., Loyola, J.R., Serrat, J., Lefevre, L., Cheniour, A.: Management Architecture and Systems for Future Internet Networks. In: Towards the Future Internet – A European Research Perspective, p. 350. IOS Press, Amsterdam (2009), http://www.iospress.nl/
16. Berl, A., Fischer, A., De Meer, H.: Using System Virtualization to Create Virtualized Networks. Electronic Communications of the EASST 17, 1–12 (2009), http://journal.ub.tu-berl.asst/article/view/218/219