Generalized Cognitive Networks of Virtual Resources

A. Manzalini[1], C. Moiso[1], R. Minerva[1],X. Hesselbach[2], J. S. Pareta[2], J. F. Botero[2]
[1] Telecom Italia, Strategy and Innovation, Torino, Italy
Name.Surname@telecomitalia.it
[2] Universitat Politècnica de Catalunya, Barcelona, Spain
xavierh@entel.upc.edu ; pareta@ac.upc.edu; jfbotero@entel.upc.edu;

Abstract.

The pervasive distribution of digital devices (e.g., laptops, mobile phone, digital camera, music players, RFID, sensors, smart cards) will create a large distributed computational and networking environment that will foster the evolution towards ecosystems of resource, services and data. Dynamicity and ubiquity of said future environments pose new challenges and requirements that current infrastructures cannot meet. Several limitations have to be overcomed, for instance: low flexibility and scalability, missing optimized cross-layer/cognitive (and cross-domain) resources allocation, limited potentialities of fostering new business models and above all a brittle integration of IT and network solutions/technologies.
This paper proposes a systemic design to develop a future network architecture overcoming said limitations. Architecture is based on three main decentralised planes: the Knowledge Plane (KP), the Network Control Plane (NCP) and the Resource Control Plane (RCP). The three planes are implemented a distributed way and interact with each other (through a signalling overlay) for the allocation and management of virtual (communication, storage and processing) resources in overlay networks: novelty stands in coupling some enabling technologies with the creation and use of a cross-layer network knowledge. Paper also presents, as an example, some simulation results showing the optimal allocation of resources for the proposed architecture. Eventually, recommendations and future work conclude the paper.

1. Introduction and problem formulation

Future networks will be increasingly complex and they will include communications, storage and processing resources heterogeneous and pervasively distributed (e.g., laptops, mobile phone, digital camera, music players, RFID, sensors, smart cards). Due to this complexity and the growing dynamism of services and applications, the tasks of ensuring end-to-end performances will be more and more challenging.

Current infrastructures cannot cope with such evolution. Traditionally, infrastructure has been developed and implemented to provide a designed, and relatively limited, set of services, usually centrally managed and controlled. Main limitations encompass: low flexibility and scalability, missing optimized cross-layer/cognitive (and cross-domain) resources allocation, limited potentialities of fostering new business models and above all a brittle integration of IT and communication solutions/technologies.

This paper proposes a systemic approach to develop a functional architecture overcoming two mail limitations. The first problem solved is getting a **unified approach in virtualizing and allocating processing and storage resources** and, above all, a practical way **to hook together above resources with communication**

services (offered by the network lower layers) to execute applications. The second problem solved is having a **formalism (e.g., both mathematical and semantic) to represent (and reason about) a cross-layer knowledge, necessary to exploit autonomic/cognitive mechanisms of self-management and self-optimization**.

In particular, architecture is capable of hooking together and managing different types of virtual resources, such as computing, storage, network resource (e.g., routers, nodes), and "smart things. It is based on three decentralised planes: the Knowledge Plane (KP), the Network Control Plane (NCP) and the Resource Control Plane (RCP).

It is argued that proposed architecture goes beyond the current state-of-art, where (to our understanding) a systemic and cognitive approach (based on a cross-layer knowledge) is missing (e.g., in projects such as VIOLIN [1], IRMOS [2]). The novelty of proposed architecture stands in enabling a deep integration of a virtualised IT resources (e.g., storage, processing) with network services. This approach opens several levels of abstraction to support an almost unlimited number of services without the restrictions of dependencies inherent to specific domains to heterogeneity of resources.

Eventually, the proposed architecture raises several issues that require further investigations in order to assess usability and scalability of service ecosystems also for deployment of future Internet.

2. Architecture and enabling technologies

As previously mentioned the proposed functional architecture is capable of hooking together and managing different types of virtual resources (up to the Users), such as computing, storage, network resource (e.g., servers, routers, nodes, laptops, mobile phones), "smart things", etc. organised in overlays networks.

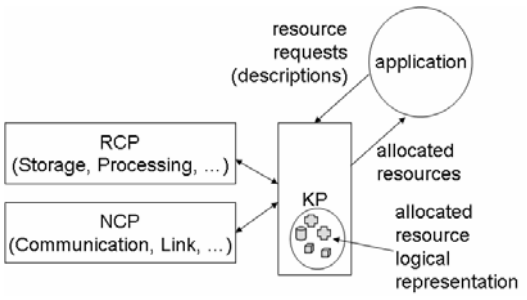

Fig. 1. Architecture for generalized cognitive networks

Architecture is based on three main decentralised planes, which strongly cooperate in the allocation of overlays of virtualised resources and in their management (Fig. 1):

- Network Control Plane (NCP) performs connection management for the data plane (where the actual forwarding logic stands) for all network interfaces (both packet-switched and non-packet-switched);
- Resource Control Plane (RCP) performs the allocation of IT-based virtual entities according to the requirements put forward by applications; allocations are decided by using distributed algorithms empowered with autonomic features; the virtual entities

represent either infrastructural resources (e.g., computing and storage) or service elements (e.g., to access application functions or data);

- Knowledge Plane (KP) receives from the applications the requests of resources and interacts with the RCP (e.g., for allocating storage and computing resources) and the NCP (for communication resources); moreover, it maintains a logical representation of all the resources allocated to applications, e.g., used for supervision purposes.

Let's consider a video-streaming application. Being this application highly resource-consuming (e.g., CPU, memory, and bandwidth), cross-layer resource allocation and management are advisable to ensure the appropriate end-to-end QoS, and to regulate resource fair resource sharing. In this case, QoS specifications encompass quantitative parameters (e.g., jitter, delay, bandwidth) and qualitative parameters (e.g., CPU scheduling policy, error recovery mechanism), as well as adaptation rules; for example, given certain streaming application requirements, the network resources must ensure certain bit rate, latency, jitter, packet error rate, etc. in principle this knowledge can be represented by an ontology (described using an expressive language that allows encoding rich semantics).

Specifically the video-streaming application interacts with KP in order to get the resources needed for its execution across a network (in general characterized by a description and a set of parameters, representing the functional and non-functional requirements). It can require and release the resources in an incremental and dynamic way, according the execution requirements.

KP interacts with RCP to get a set of (virtual) resources candidate for the allocation. RCP returns a set of resources fulfilling the application requirements, each of them labelled with some additional information useful for optimizing the selection according to the adopted criteria (e.g., performance, cost reduction, or power savings).

KP, by using the updated information on the availability of network resources (e.g., obtained though NCP) and the information previously allocated to the application, selects the "best" subset of virtual resources returned by RCP, and requests NCP the provisioning of the virtual links interconnecting in overlays the selected virtual resources. NCP, by means of a distributed network signalling (e.g., with RSVP-TE) allocates the network resource, fulfilling the application requirements (e.g., QoS parameters), and serving to set-up such overlays, and returns to the KP the data on the allocated virtual links. In case of failure, KP can either select a "sub-optimal" subset of virtual resources, or negotiate with the application a refinement of the requirements.

2.1 Enabling technologies in a nutshell

This architecture relies on some key technologies, such as virtualization, distributed overlay, autonomic and knowledge representation and analysis.

It's true that some recent proposals in the context of virtual networking provide effective mechanism to flexibly reconfigure and maintain a network/distributed system. For instance VIOLIN [1] presents an architecture to relocate virtual machines in an autonomic way so as to optimize network usage transparently from the application point of view. Nevertheless, this is purely IT based approach, while the proposed architecture is enabling a deep integration of virtualised IT resources with network services provided through a cognitive extension of GMPLS technologies.

The three planes (KP, NCP, RCP) are implemented in a decentralised way, as a set of instances running on different nodes, interacting though distributed processing mechanisms. Instances can be deployed either on some dedicated servers, in charge of implementing cognitive capabilities, or on the computing resources in the system. For instance, RCP instances are implemented by specific functions in the virtualized resources (e.g., the negotiation and allocation function), while KP instances can be distributed on specialized servers or on dedicated partitions of virtualized resources.

All the local instances of a plan are interconnected though an overlay; its links are in charge of transporting signalling-like messages, for enabling the interworking and the cooperation of the local instances. These overlays are created and maintained through self-organization logic, e.g., able to cope with recovery from failures and to optimize its performance. The signalling links of these overlays are set-up as virtual links, by using the NCP facilities. Overlays are suitable means to abstract the heterogeneity of the underlying computing and communication infrastructure, cope with the dynamic evolution of the nodes/entities involved in the system, and to achieve scalable solutions.

One important aspect for the plan implementation is the absence of the requirement to adopt a single unifying component model: this is in line with the middleware-deconstruction approach [3]. Therefore, the design of the instances of the plans should not necessarily include the definition of how they are made internally (e.g., which component model, or toolkit must be adopted). Each developer should be able to create components without any constraints. The only constraint is on the alignment to the protocols used for the interactions among the instances of a plan and those for the interactions among instances of different plans. This also includes the requirement to adopt the same formalism for coding the information transported through the protocols.

In order to deal with different kinds of data and information sources, it is fundamental identifying general-purpose data and meta-data models and languages able to capture the main characteristics of present and future resources. An important and vast area of research with regard to knowledge representation comes from the Semantic Web community. In this context different forms of data representation (e.g., RDF and OWL) have been proposed to effectively browse, search and share information among services and applications. More recently, the need to take into account the proper trade-off between the adoption of common ontologies to describe data, and the necessity to pragmatically account for emerging (user-generated) ontologies has been recognized. Other than data organization and representation, it is – of course – fundamental to have mechanism to process and analyze (to reason about) these data to transform them in knowledge for taking decisions and autonomic actions.

2.2 Resource virtualization models

These plans work on virtual resources which are provided by “physical” resources, according to a virtualization model. The virtualization model enables to create, starting from a shared ensemble of physical resources, a uniform substrate of virtual resources, dynamically allocable to applications. The model is generalized as it can be applied a different types of resources, such as computing, storage, network components (e.g., bridges, NAT, host-only, virtual switches, virtual DHCP server, and virtual network adapters), “thinks”, service components, etc. A virtualization model includes:

- an abstract view of the features/functions provided by the resources, in order to (1) simply the access to the capabilities, and (2) hide the heterogeneity on interfaces/ protocols actually provided by physical resources to control/access their features;
- a mechanism to group the capabilities of the resource (e.g., in terms of processing, storage, bandwidth) in isolated partitions; the partitions are to be used as allocation units of virtual resources; configuration mechanisms allow to associated, in a flexible way, the capabilities of the physical resources (e.g., storage dimension, execution cycles, queue lengths, optional functions) to the partitions, and to configure the QoS parameters, to fit the negotiated SLA; each partition is an isolated context to protect the physical resource and the other partitions to an incorrect/malicious behaviour of the application and create an environment for the enforcement of negotiated SLA, by avoiding (functional/non-functional) interferences among the partitions;
- a formalism for the requests and negotiation of virtual resources, e.g., in terms of dimension/quantity of resources to be allocated and description of non-functional features; an associated protocol is adopted to request/negotiate virtual resources.

Through a virtualization model, a physical resource can be "virtualized", so as to create and manage virtual resources to be assigned to applications. The following sections introduce the models adopted to virtualize IT and communication resources.

2.2.1 Virtualization model for communication resources

Network virtualization [4] is a powerful technique as it provides flexibility, promotes diversity, and promises security and increased manageability: by allowing multiple heterogeneous network architectures to cohabit on a shared physical substrate, network virtualization is a diversifying attribute of the future inter-networking paradigms.

Prior-art is already providing interesting proposals of network virtualization at different layers: examples are L1-UCL, L2-VNET, L3-AGAVE, VIOLIN, etc.

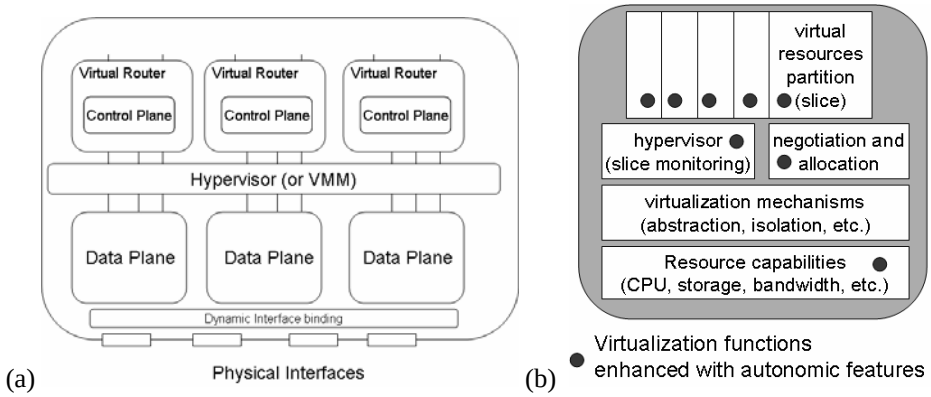

Fig. 2. Architecture for virtualized communication resources (a) and IT-based resources (b)

Existing technologies provide already coarse-grained link virtualization. Virtual Private Network (VPN) connecting multiple distributed sites through tunnels over shared or public networks (e.g. L3 VPN based on IP and MPLS, L2 VPN based on Ethernet, L1 VPN based on circuit-switching and VPN at the application layer). An Overlay Network is again another form of network virtualization which is typically implemented in the application layer, even if various implementations at lower layers of

the network stack do exist. PlanetLab represent an interesting example: Users (mainly researchers as PlanetLab is a scientific test-bed) can easily create and manage customized environments to design and evaluate new architectures, protocols, and algorithms. Also peer-to-peer networks (e.g. BitTorrent) can also be seen as examples of virtual networks. However, they suffer from a lack of node programmability. Virtual Machine Monitor (VMM) features the ability of a physical router to host several virtual machines on its hardware (Fig. 2 - a). Nevertheless, all these examples suffer from a lack of node performances and programmability, sufficient isolation, cross-layer and cross-domain interoperability, optimization of physical resources, etc.

Future scenarios imply many challenges for Network Resource Virtualization. Ability to control and move these virtual nodes creates a need for an intelligent architecture that can adapt to changing demands and conditions by autonomically tuning performance, isolation, migration, and sliver resizing. Among others, there is also a need to find innovative solutions for developing cross-domain virtual control planes, high performance virtual forwarding planes, and to make the forwarding planes re-programmable: with this capability, one can think about self-organizing forwarding paths where they adapt dynamically to the incoming traffic or applications requests (even for new forwarding path experiments without affecting the operation of the existing network, for example see OpenFlow). Virtual router should be easily created, stopped and moves from one physical resource to another. Migration of control plane is an interesting topic (VROOM Router leveraging virtual machine migration techniques).

2.2.2 Virtualization model for IT-based resources

A “virtual-machine” like model, similar to the one proposed by XEN [6], was adopted for virtualizing IT-based resources. The following functions are identified (Fig. 2 - b):

- the capabilities (in terms of hardware, Network, Operating System Capabilities, services at applicative level, etc.) offered by the physical resource (e.g., CPU, RAM);
- the virtualisation mechanisms for partitioning the resource capabilities in “isolated” slices; they provide interfaces to access the allocated resources, according to the adopted abstract view; examples of slices are the Virtual Machines;
- hypervisor monitors the state of the resource capabilities and of the slices; it implements supervision functions in order to detect critical situations, and trigger a decision process to select and activate corrective actions; for instance, hypervisor must be able to enforce the policies to fulfil negotiated SLAs, to detect and recover failures, to optimize performances and the use of resources; in order to accomplish these tasks, the hypervisor maintain a knowledge on the state of the resource;
- virtual resource allocator in charge to allocate, de-allocate and, possibly, migrate virtual resources, e.g., partitioned in slices, according to the requests; this function is in charge of implementing policies governing the negotiation and the allocation of virtual resources to fulfil the received requests; it has to interwork with hypervisor to get information on available resources, with the virtualization mechanisms to request the creation of slices, their configuration (in terms of allocated resources, SLA parameters, etc.), and their termination; virtual resource allocator must be able to cope with formalism and protocol for requests and negotiation of virtual resources;
- a set of slices, each of which represents the partition of virtual resources allocated to an application; each slice is an “isolated” context that perform application activities.

2.3 Overlays of virtual resources

The plans in the generalized cognitive networks create, maintain and, possibly, optimize the allocation of virtual resources, such as computing, storage, network resource (e.g., servers, routers, nodes), "smart things", to distributed applications deployed across a network (e.g., the Big Internet).

The virtual resources allocated to a specific application are organized in **monitoring overlays**. KP, which is in charge to keep a view of all the resources allocated to an application, is in charge to create and maintain these overlays. Monitoring overlays can be used by an application in order to have a view of all the virtual resources allocated to it: by means of such a monitoring overlay, the application can, for instance, perform some supervision/management activities, e.g., in order to optimize its performance or the use of the allocated resources (e.g., in order to reduce the costs), or to detect/recover some failures. Monitoring overlays can also be used by KP in order to supervise allocated virtual resources. Monitoring overlays interconnect software entities in charge to monitor/supervise the virtual (IT-based or network-based) entities. For instance, they could interconnect, the "slices" of a virtual machine, or the software representation of (GMPLS) virtual connections. The links interconnecting distributed elements of monitoring overlays are used to transport signalling messages. These signalling links are set-up as virtual links, by using the NCP facilities.

While the monitoring overlay is introduced for enabling the activities, performed by KP and the applications, for the supervision of the virtual resources allocated to an application, the **application overlays** are created by the applications to form the distributed computing and communication dedicated to execution of their components. For instance, the application uses the allocated computing virtual resources to deploy its software modules/instances, and the storage resources to store its data; these IT-based resources interact though the allocated virtual communication links.

2.4 Autonomic features

Architecture entities (e.g., local instances of the plans, virtualized resources) must be enriched with autonomic features in order to:

- promptly react to critical/unplanned situations, by identifying contingency plans;
- optimize resource configurations and allocations;
- tame complexity of huge amount of entities, by ruling aggregations and cooperation.

Autonomic features are introduced by means of autonomic control loops:

- internal control loops: they are in charge of achieving self-awareness, to control the internal state and events, to detect (or forecast) critical and/or unplanned situations, and to trigger a decision process to select and actuate the corrective actions;
- external control loops: they are in charge of achieving self-organization and self-adaptation of the autonomic entity with respect to. its "environment", by processing events and messages sent by other entities (e.g., resources, local instances of the plans, and applications) interconnected though overlays.

External control loops can be used to implement a cooperative behaviour among peer entities to perform distributed decision algorithms or to create a "non-local" vision of the state of the system. The messages consumed and produced by external control

loops are exchanged among peer entities through (self-organized) overlays, by using gossiping-like protocols, to reduce the communications.

Autonomic entities in the architecture can be implemented as ensembles of interacting autonomic components, such as the ones provided by CASCADAS ACE Toolkit [7]. Autonomic control loops need to work on "knowledge", e.g., on:

- internal state of a resource (e.g., managed by the hypervisors);
- the local environment, achieved through gossiping protocols among peer entities;
- the state of the whole system, retrieved from the KP, which has to create and maintain it, by collecting and processing data from all the entities in the system.

Autonomic control loops must implement decision processes, for selecting the (best) actions to be performed to react to some critical/unplanned situation and for self-adapt its behaviour. Decision processes can exploit different technologies, e.g., game theory, artificial intelligence, auction models.

In the architecture for generalized cognitive networks, autonomic features are included in both the local instances of the plans, and the virtualized resources. In particular, functions in virtualized resources may provide the following autonomic features [8], which must cooperate by exchanging information, and events:

- Hypervisor: (1) self-CHOP features to supervise the behaviour of the slices; (2) self-adaptation of policies to guarantee the enforcement of the negotiated parameters associated to the slices; (3) algorithms to monitor/forecast the load;
- Allocator: (1) algorithms for the negotiation and allocation of virtual resources, in cooperation with peer entities for achieving "optimal" policies; (2) algorithms for deciding the migration of slices to/from other resources;
- Slices: Self-CHOP features to monitor/tune behaviour to fulfil negotiated parameters;
- Resources capabilities: Self-CHOP features to monitor/supervise their state.

3. Network Control Plane

The Network Control Plane can be seen as a decentralised control plane that supports the ability to establish a network service through network signalling. This is in contrast to the traditional method, where a management system establishes the network service by configuring each individual network element.

NCP performs connection management for the data plane (where the actual forwarding logic stands) for all interfaces (both packet-switched and non-packet-switched). NCP encompassed the following features: Discovery, Routing, Path Computation, Signalling. The NCP will evaluate the strategy to solve all of these problems. This layer takes into account the knowledge and the state of the devices, according to some data model, such as the one defined in the previous section.

These strategies must be defined. Typically, this will be based on an optimization formula with a set of restrictions. Taking into account the graph of the network, in general the problems will become NP-complete. Since the complexity, in general, a heuristic should be formulated and the implementation will consider approximate algorithms. This is for further study. The implementation of a decentralised control plane must be analysed in order to be as close as possible to the global optimum for every feature managed by the NCP.

NCP design can leverage the GMPLS (*Generalised Multiprotocol Label Switching*) control plane. GMPLS was proposed to extend MPLS (*Multiprotocol Label Switching*)

to cover not only the control of the packet switching but also the time division (for example, SDH), wavelength (for example, optical networks) and spatial switching (for example, for incoming port or fiber to outgoing port or fiber).

MPLS creates a sort of overlay network (based on packet-switched IP) to facilitate traffic engineering and allow resources to be reserved and routes predefined. It provides virtual links (or tunnels) through to connect network nodes. For packets injected into the ingress of an established MPLS tunnel, normal IP routing procedures are suspended; instead the packets are label-switched so that they automatically follow the tunnel to its egress. GMPLS is based on the idea of a label can be generalized to be anything that is sufficient to identify a traffic flow, both connection-less and connection-oriented. A GMPLS piece of equipment, generally called Label Switch Routers (LSRs), has a control-plane (in analogy with IP routers) that bases its functions on a table that maintains relations between incoming label or port and outgoing label or port.

As an example, the GMPLS control plane essentially performs the following basic functions. It supports switching in time (SDH), wavelength (WDM), and space domains (ports) along with packet switching. So, it generalizes and integrates all kind of traffics.

NCP extend the GMPLS control plane with the novel idea of introducing autonomic/cognitive capabilities for deploying autonomic features, such as self-reconfiguration, self-adaptation, self-optimization of network infrastructure resources (Layer 3, 2 and 1). Moreover, interworking of NCP with RCP and KP will prove a novel and promising way for an effective integration of service and network resources. From this perspective there is the need to develop standard resources representation schemas for any virtual resources and virtual resource sets to exchange services. The inter-domain exchange of information and synchronization has also to work smoothly and scale gracefully.

4. Resource Control Plane

The RCP performs the allocation of IT-based virtual resource (i.e., slices of Storage, Processing, etc.), according to the requirements put forward by the applications. RCP identifies the nodes where the needed resources are best allocated (according to adopted allocation policies), and returns to KP the data used for an optimized global allocation.

This section describes a solution for implementing an RCP in a decentralized way, able to fulfill requirements on scalability, and reliability, in particular in pervasive and multi-domain context. According to the internal function architecture of virtualized resources, the “negotiation and allocation” function has to process the requests of “virtual resources”, by matching them with the capabilities presently available in resource (e.g., in cooperation with the hypervisor), and, in case of successful allocation, request to configure a slice, fulfilling the request parameters.

In order to achieve a distributed solution for virtual resource allocation, all the “negotiation and allocation” functions related to the same type of virtualized resources must interact to cooperate, by possibly optimizing the allocations, according some given policy/criteria. They are interconnected through an overlay, dedicated for the exchange of information relevant for resource allocation, de-allocation, and re-allocation (e.g., to implement load balancing or fault recovery policies). This cooperation infrastructure can, for instance, enable an auction-based approach [9]: when a KP instance receives a request of a virtual resource from an application, KP forwards it to one or more

allocation functions of the virtualized resources through the overlay (for given number of hops). Each returns a bid (e.g., the amount of currently available resources or its estimation in the near future). The KP instance selects the "best" bid and sends back a confirmation. When the selected resource receives the confirmation, it creates a slice and allocates it to the requesting application.

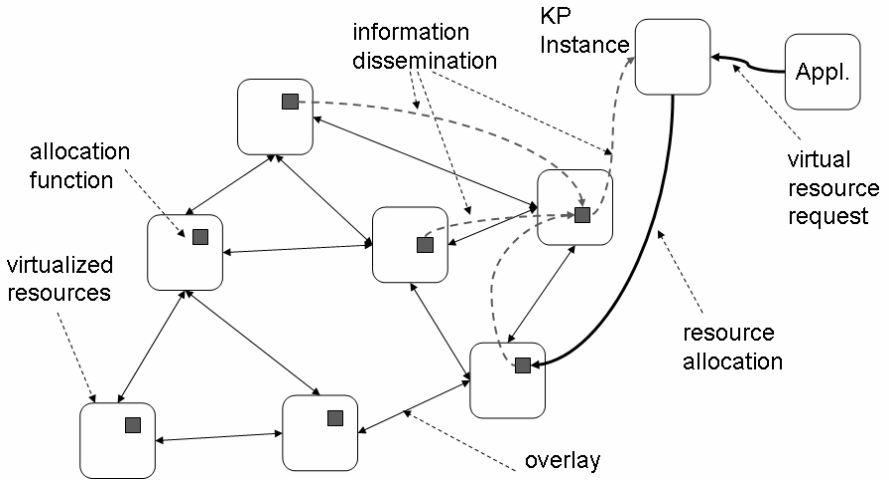

Fig. 3. Dissemination of resource availability information for cooperative distributed allocation

An alternative approach is to use the overlay to disseminate information on the resource availability on each resource (Fig. 3). The dissemination could be performed though gossiping protocol. When a KP instance receives a request, it selects the "best" virtualized resource able to serve it, according to the disseminated information, and forwards the request to its allocation function. In this way, KP instances create knowledge on resource state, share and combine it among all the other KP instances.

The overlays in the RCP can also be used to exchange information, through gossiping protocols, to balance resource allocation, e.g., due to picks of traffic, several cycles of allocation/de-allocation, faults [10].

5. Knowledge Plane

KP receives from applications (or even Users) the requests of resources and interacts with the RCP (for allocating IT-based resources) and the NCP (for allocating Communication resources) to assure the most appropriate allocation; moreover, it maintains a logical representation of all allocated resources allocated, for optimization purposes (e.g., engineering re-allocations) control and supervision purposes.

KP learns how to improve the network behaviour and may instruct the other planes, or even directly the resources, on how to change their configurations in order either to optimize the network performance or to adapt it to the requirements of new services. The configuration task of a network and its resources become a continuous process, adapting them to the evolution of their conditions, priorities, and of application needs.

We introduce the concept of knowledge field (the word field has to be intended according to the Physics meaning), spatially distributed across all resources, created and used by the KP. It looks like a sort of gravitational field emitted by each component itself. By following the local shape of the fields components perform decision-making and actuation processes. In other words KP components sense the knowledge field and act accordingly to specified behavioural patterns or plans. Also knowledge field models can potentially be implemented, as an overlay network, on any middleware providing basic support for data storing, communication and event-notification. What is required is to provide simple storage mechanisms (to store field values), communication protocols and primitives (to propagate field values to peers), and pub-sub mechanisms.

5.1 Knowledge representation

The cognitive features of the Knowledge plans require adopting a language for the representation of the "knowledge" on the system created, maintained, updated and processed (e.g., by reasoning, learning or decision algorithms) by KP.

It must be able to represent different types of information, both numerical/quantitative and symbolic/qualitative [12], including: the description of the resources in the system, their states (e.g., the amount of virtual resources allocated, information on failures, load, performance, etc.), the description of the active and pending requests of virtual resources performed by the applications, the description of virtual resources allocated to applications, etc.

This language also complements the definition of the "signalling" protocols defined for the interaction among the plans' instances, in order to code the transported pieces of information. For instance, it can be used to describe the virtual resources requested by an application, to be included in the messages forwarded to the negotiation and allocation functions [13].

The representation of the knowledge must enable and ease to execution of algorithms for its creation, maintenance and processing. In particular, it must enable the adoption of knowledge network solutions, for the aggregation and organization of knowledge, and semantics reasoning. Moreover, due to the organization of the KP as an ensemble of instances, the representation must allow the distribution of the knowledge descriptions across multiple instances. In principle, KP requires a collection of knowledge representation formalisms. Examples would range from purely statistical descriptions (e.g. co-occurrences and correlations based models) to highly semantic ones (logical representation of networks, descriptions of devices, descriptions of activities etc.). It is proposed to define a scalable network knowledge representation for KP, based on a combination of mathematical models (e.g., for reasoning on quantitative parameters) and semantic/logical formalisms (e.g., for reasoning on relationships among pieces of knowledge, and qualitative characteristics and constraints), such as RDF.

In general the knowledge representation formalisms must be selected to make cognitive processing for a network feasible: the performance of KP must be carefully planned (e.g., by introducing dynamic filters on data to be collected, and processed), to reduce the resources required by KP to create/maintain/process the knowledge. [14].

5.1.1 Example of semantic representation model for resources

The network knowledge development and use require a language to describe both network and IT resources (and the related services) in an abstract way (at the application level). A possible approach might be adopting some reference language developed by W3C. The idea of using RDF to describe network concepts has been already considered: for instance Network Description Language (NDL) [15] provides ontology for computer networks that can be easily extended, even if this approach does not fully cope with handling network resource requests. An interesting proposal is the Network Resource Description Language (NRDL) described in [16]: through NRDL each communication can be enriched with detailed information about QoS and network requirements (i.e. bandwidth, jitter, delay, etc.).

5.1.2 Example of representation model for network resources

The resources offered by the network (the physical devices) are different depending on the nature of the technology supported: for instance from wavelengths to circuits, from label switched path to ports, or even other abstractions for OPS (Optical Packet Switching) or OBS (Optical Burst Switching) technologies.Therefore, we will consider a classification of resources based on its technology. This classification will be done in the nodes but also in the links [17]. So, we will call NR (node resources) the resources available in the nodes (processing capacity, maximum number of instances, number of interface types, etc), and LR (link resources) the resources in links (ports, number of lambdas, bandwidth for OBS, types of Virtual Containers for SDH, etc). Both NR and LR are offered by the substrate network.

Substrate Network (SN). SN is represented by a directed graph $G(V;A)$, where V is a set of elements called vertices, representing the set of nodes of the SN, and A is a set of elements called arcs, representing the set of links of the SN. Considering an ordered set of vertices $V_1; V_2;\ldots; V_n; V_{n+1}$; a *free of cycles directed path* (just path from now on) is any sequence of edges $\in E$ of the following type: $\{(V_1; V_2); (V_2; V_3); \ldots ; (V_n; V_{n+1}))\}$.

Each SN has a set of associated nodes and links capacities. We define a set of n resource parameters associated with a node and m resource parameters associated with links. Node resources will be represented, for each node $i \in V$, as $NR(i) = \{NR_1(i);NR_2(i);\ldots ;NR_n(i)\}$. For instance, $NR_1(i)$ can describe the number of available instances of a node $i \in V$, $NR_2(i)$ the number of ethernet interfaces, etc.

Likewise, link resources will be represented, for each link $(i; j) \in A$, as $LR(i; j) = \{LR_1(i; j);LR_2(i; j); \ldots ;LR_m(i; j)\}$. For instance, the number of available wavelengths in the link $(i; j) \in A$ is represented by $LR_1(i; j)$.

Virtual Resources Request (VRR). Let VRR be a set of virtual resources requests, each $VRR_k \in VRR$ is represented by the directed graph $G^k(V^k;A^k)$ and has associated virtual nodes and link demands. Virtual node demands (ND) will be represented, in each $VRR_k \in VRR$ for each node $i^k \in V^k$, as: $ND(i^k) = \{ND_1(i^k);ND_2(i^k); \ldots ;ND_n(i^k)\}$. Also, virtual link demands will be represented, in each $VRR_k \in VRR$ for each link $(i^k; j^k) \in A^k$, as: $LD(i^k; j^k) = \{LD_1(i^k; j^k);LD_2(i^k; j^k); \ldots ;LD_m(i^k; j^k)\}$. For example, wavelengths demand for the link $(i^k; j^k) \in A^k$ can be represented by $LD_1(i^k; j^k)$.

6. Simulation results

This section provides some preliminary evaluation results on the performances of the distributed allocation algorithms introduced for RCP, achieved through simulations. The experiments considered a simplified version of the algorithms, in which a single resource is selected by the KP instance for each request: i.e., the experiments do not consider the possibility to perform a second choice in case the selected resource is not able to create the slice of virtual resources. Moreover, they assumed that the KP instance logic is executed by the resources themselves. The results pointed out that:

- Auction-based algorithms depend on how to estimate the resources reserved to serve the active bids; in fact, if the estimation should balance between the requirements to increment the level of efficiency in using the resources, and reduce the failures in serving a confirmed allocation;
- Dissemination-based algorithms depend on the rate of update of the data on resource availability, and on the estimation of information "reliability";
- Dissemination-based algorithms react more quickly to allocation requests, as the information on availability are proactively sent by the resources; they can perform selections as soon as they receive a request, while auction-based algorithms require at least *1+k* steps, where k is the number of hops.

To create the overlays considered in the simulations, each node randomly selects *N* neighbours (and can be selected by the other nodes); thus, each node has at least *N* neighbours, and at the average, *2N* neighbours.

Table 1. Simulation results of distributed allocation algorithms for RCP

		Auction-based		Dissemination-based	
	Hops	1	2	1	2
N	1	8.29%	-------	7.15%	-------
	2	**4.20%**	4.42%	**3.60%**	4.38%
	3	5.62%	6.71%	7.02%	18.65%

Table 1 summarizes the simulation results; it reports the rates of failures in allocation requests. The case *Hops = 1* involves a single node in the selection. The reported data are related to simulations with 100 nodes, but the simulations showed that the algorithms scale for *N=225*, and *N=400*. In both solutions, the best results is achieved for overlays created with *N = 1*, and with information propagated through 2 hops. The dissemination based solution has a better performance, as: (1) it can perform decisions on more updated information, and (2) it has a more effective usage of resources, as it does not require resources reservations. On the other hand, the performance of dissemination-based solution strongly depends on reliability of disseminated information: the "freshness" of the information decreases when more hops are considered; moreover, it becomes unreliable, with the increment of the amount of nodes that receive information on a node, and, thus, can select it for allocating resources. This motivates the high rate of failures of the case *Hops = 3* and *N = 2*.

These simulations provide only a preliminary estimation of the performances of the algorithms. Several parameters must be considered for a complete evaluation of the approaches (e.g., the rate of updates in information dissemination, methods to improve the estimation of bids, the impacts of reserved resources).

7. Conclusions and future steps

The pervasive distribution of digital devices (e.g., laptops, mobile phone, digital camera, music players, RFID, sensors, smart cards) will create a large distributed computational and networking environment that will foster the evolution towards ecosystems of resource, services and data. Traditionally, infrastructure has been developed and implemented to provide a designed, and relatively limited, set of services, usually centrally managed and controlled: this is creating several limitations to cope with this evolution.

This paper proposes a systemic approach to develop a functional architecture overcoming said limitations. In particular, architecture is based on three main decentralised planes: the Knowledge Plane (KP), the Network Control Plane (NCP) and the Resource Control Plane (RCP). The three planes interact with each other for the allocation and management of virtual resources in overlay networks: specifically NCP performs connection management for the data plane going beyond the state of art (for instance it encompasses autonomic functionalities, and it is extended to deal with access network resources); RCP performs, by means of distributed algorithms empowered with autonomic features, the allocation of IT virtual resources; KP holds the dialogue with applications (or even Users) to handle the dynamic requests of resources, and it interacts with (and bridge) RCP and NCP, thus exploiting a cross-layer network knowledge. The three planes are implemented a distributed way, e.g., by interconnecting, through a signalling overlay, instances hooked together and running on different nodes.

Preliminary simulations demonstrate the allocation of IT resources achieved through a distributed algorithms implemented in RCP is feasible. Allocation based on the information disseminated, through gossiping, offers better performances both in terms of response time and in reduction of allocation failures.

Through this architecture, value creation can migrate within the value-chain according the context in which the devices/nodes/networks are operating in, and the time-varying performance objectives. It is reasonable to admit that the overall value is potentially much greater (both for Operators and Users) than the one of the traditional architecture; interestingly, KP will enable the symbiosis of privately owned and community networks (thus allowing a potential split of local and global connectivity costs) and new business opportunities in synergy with Internet of Things.

7. References

[1] X. Jiang, D. Xu, VIOLIN: Virtual Internetworking on Overlay Infrastructure, Proc. Int. Symp. on Parallel and Distributed Processing and Applications (2004), 937-946.

[2] K. Oberle, M. Kessler, M. Stein, T. Voith, D. Lamp, S. Berger, Network Virtualization: The missing piece, Proc. ICIN2009 (2009).

[3] M. Mamei, F. Zambonelli, Programming Pervasive and Mobile Computing Applications: the TOTA Approach, ACM Trans. on Software Engineering. and Method. 18(4), (2009), 1-56.

[4] N.M.M.K. Chowdhury, Network virtualization: State of the art and research challenges, IEEE Communications Magazine 47, (2009), 20–26.

[5] N.M. Chowdhury, R. Boutaba, A survey of network virtualization, Computer Networks, 54(5), (2010), 862-876.

[6] P. Barham, B. Dragovic, K.raser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, A. Warfield. Xen and the art of virtualization, Proc. 19th ACM Symposium on Operating Systems Principles (2003), 164-177.

[7] CASCADAS, ACE Toolkit open source, http://sourceforge.net/projects/acetoolkit/.

[8] D. Marinescu, R. Kröger, State of the art in autonomic computing and virtualization, Distributed Systems Lab, Wiesbaden (2007).

[9] S. Magrath, F. Chiang, S. Markovits, R. Braun, F. Cuervo, F., Autonomics in tele-communications service activation, Autonomous Decentralized Systems (2005), 731-737.

[10] P. H. Deussen, L. Ferrari, A. Manzalini, C. Moiso, Highly Distributed Supervision for Autonomic Networks and Services, Proc. AICT2009 (2009), 111-116.

[11] D. Clark, C. Partridge, J. Ramming, J. Wroclawski, A knowledge plane for the internet. Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols For Computer Communications (2003), 3-10.

[12] D. Lewis, D. O'Sullivan, K. Feeney, J. Keeney, R. Power, Ontology-based engineering for self-managing communications, Proc. Modeling Autonomic Comm. Environment. (2006).

[13] F. Callegati, W. Cerroni, B. Martini, M. Gharbaoui, A. Campi, P. Castoldi, Configuration of Network Resources for Future Internet Application Services, Proc. Future Network and Mobile Summit 2010 (2010).

[14] A. Manzalini, P. H. Deussen, S. Nechifor, M. Mamei, R. Minerva, C. Moiso, A. Salden, T. Wauters, F. Zambonelli, Self-optimized Cognitive Network of Networks, The Computer Journal (2010), doi: 10.1093/comjnl/bxq032.

[15] J. van der Ham, P. Grosso, R. van der Pol, A. Toonk, C. de Laat, Using the network description language in optical networks, Proc. Integrated Network Management, (2007).

[16] A. Campi, F. Callegati, Network resource description language, Proc. IEEE GLOBECOM Workshops, (2009), 1–6.

[17] J.F. Botero, X. Hesselbach, A. Fischer, H. De Meer, Optimal mapping of virtual networks with hidden hops. Telecommunication Systems Journal (2010).