Distributed Service Framework: an innovative open eco-system for ICT/Telecommunications

Rosario Alfano
Telecom Italia Lab
via Reiss Romoli 274
10148 – Torino - Italy
+39-011-2285854
rosario.alfano@telecomitalia.it

Antonio Manzalini
Telecom Italia Lab
via Reiss Romoli 274
10148 – Torino - Italy
+39-011-2285817
antonio.manzalini@telecomitalia.it

Corrado Moiso
Telecom Italia Lab
via Reiss Romoli 274
10148 – Torino - Italy
+39-011-2286780
corrado.moiso@telecomitalia.it

ABSTRACT

In order to fuel new emerging service markets, next generation service delivery framework have to be based on an highly distributed and open architectures adopting innovative technologies capable of enabling features such as robustness, security, stability and scalability.

The talk will elaborate on RT&D activities and results concerning a Distributed Service Framework (DSF) solution that has been designed as an innovative open eco-system for ICT/ Telecommunications services. Specifically, the DSF architecture is based on multiple autonomic components that are interacting each other via a self-organising overlay. Particular attention is devoted to the service execution environment of the DSF where scalability and dynamic load balancing are met by self-organization capabilities of the components. The design choices, specifically the autonomic and altruistic local behavior of components and "ServiceSpace", enable global features effortlessly, simplifying implementation and management complexities.

Keywords

Distributed Service Framework, Agents, Autonomic, Virtualization, ServiceSpace

1. INTRODUCTION

Technology is currently offering a wide set of portable digital devices (e.g. cell phones, PDAs, laptops, digital cameras, music player, Wi-Fi devices) at relatively low cost. Penetration of these and other miniaturized digital devices (e.g. sensors) is becoming deeper and deeper in modern cities. This driver is opening new opportunities to produce and access ubiquitously cross-media applications and services.

For example, the dynamics of a city can be captured in real-time by collecting and correlating data (anonymous localization, traffic, pollution, cultural sites, events, etc) provided by heterogeneous digital devices and data sources. Real cities are like ecosystems with self adaptive and self-organizing properties where social patters represent valuable (anonymous) information for providing situated services even anticipating Users' needs. Services may include pervasive communications, personalized cross-media real-time contents, infotainment, commercial and social information. Pervasiveness requires supporting technologies capable of scale over highly distributed resources with de-centralization of functionalities. These requirements might be met by dynamically configurable structure using a P2P overlay network.

As a matter of fact another contemporary driver is the wide adoption of peer-to-peer solutions to deliver services to end-users (e.g. Skype, Jost, Edonkey). It should be note that current peer-to-peer solutions are basically application-oriented and they don't meet the requirements of network-transparency for cross-media applications and services; nevertheless they has engendered a great interest for flexible, robust, secure service and control peer-to-peer distributed service platforms; further requirements that have to be met include scalability and survivability, especially in unpredictable and chaotic environments and when failures occur.

In order to meet the challenges and the opportunities offered by this service context, innovative technologies and solutions are required for exploiting highly distributed environments. In principle a distributed system is a collection of independent sub-systems (linked with distributed software) that appear to the operators/users of the system as a single entity. Distributed software enables sub-systems to coordinate their activities and to share the resources of the system - hardware, software and data. Actually this is likely to be an attracting model for the evolution of next generation Service Platforms capable of meeting the requirements of an emerging service context as described above.

The Distributed Service Framework (DSF) model, presented in this paper, is based on secure, robust, scalable, component-based environment capable of operating even in disconnected and chaotic environments (e.g. with corrupted information and capabilities).

DSF aims addressing the identified challenges, in particular by:

- enabling an open service eco-system which allows Individuals (e.g., as Prosumers), Service/Contents Providers and other Operators to produce and consume services according to innovative business models;
- providing a platform for service delivery, that reduces the costs, for both infrastructure and management/ operation aspects, of service providers

From the technical point of view DSF aims overcoming the limitations and drawbacks of current solutions for service delivery, based on centralized environments for the execution of Business Service Logic, and SOA solutions for reusing and

sharing decentralised ICT/Telco resources and capabilities, through the adoption of:

- autonomic techniques, both at the platform and at the service logic level, to achieve high degree of self-organization, self-management ("self-chop"), self-adaptation properties , etc., to the reduction of management costs and to the improvements of the reactiveness of the system to unplanned or unexpected events;
- distributed solutions for services and data, to achieve high levels of scalability, reliability, and optimization of resource usage, and seamless involvement of end-users devices.

2. TELECOMMUNICATION SERVICE PLATFORMS: CURRENT TRENDS AND LIMITATIONS

Currently telecommunication services are mainly created, executed and managed by a "Service Layer", which consists of a set of systems implementing the functions needed for the service delivery. The services are characterized by a "business logic" that may use and combine a set of Telco features which provide some basic service capabilities. Examples of Telco features are: capabilities for controlling bearers/channels; enablers for value-added capabilities and user-adaptation; identity and data management: end-user identity federation, end-users profiles, service profiles, terminal profiles, end-users account, billing/charging, etc. Services are either triggered by some events produced by Telco Features (e.g., call activation, the reception of a new message, the change of location of a terminal/end-user), or activated by some applications (e.g., deployed on an end-user terminal interacting through a specific protocol, such as HTTP, etc.). Context awareness in service activation, as this is provided through external (location, sensor readings, pushed content) or internal (device status, memory or power levels) triggering is the means to offer comprehensive service provision, with a minimum of user intervention. Currently, services interact with Telco Features and terminals through a wide set of protocols, each of which specialized to deal with specific capabilities/functions. Examples are: INAP, and CAP for session and call control, LIF for localization, SIP/ISC for session control and presence, USD for control of SMSs, MM7 for control of MMSs, etc.

Most of the current service layer deployed in the Telco Operator's infrastructures are realized as a set of "vertical" platforms or solutions, named Silos, each of them specialized to provide services involving a specific service classes (e.g., content-based services, location based-services, massaging) or a specific networks. In general, such platforms integrate in a single system the service execution environments, the relevant Telco features and some supporting functions (e.g., payment, authentication, profiles). Some Silos-based service environments are based on standards (e.g., IMS for IP-based multimedia services, IN for circuit-based services, OMA and 3GPP specifications for messaging and location-based services), others are proprietary products and/or customized solutions. Whatever the case, Silos-based service environments are typically characterized by: specific servers for service logic execution; specialized mechanisms/ protocols to activate services and to configure such activations; specialised repositories handling the data necessary for the execution and management of the services running on it; repositories for storing user preferences and service activation/deactivation rules; specific and/or proprietary protocols for the interaction of services with other services and with the functions, implemented in it.

Silos-based service environments exhibit a number of limitations with regard to the characteristics requires by innovative service scenarios. It is quite difficult to integrate/ combine functions belonging to different Silos without costly integration efforts, due to lack of adoption of open and/or standard interfaces to access their internal functions. Such a context, where the silos-based platforms deployed in the service layer are loosely integrated (and hardly integrable), introducing serious constraints and complexities in the possibility and efficiency in the process of creating and deploying new services.

In order to overcome such situation, the service layer is evolving towards a "horizontal" approach based on the integration among platforms for service delivery which are deployed in the Operator infrastructure, and the sharing of and interoperability among functions, enablers and service capabilities [1]. Although at present there is not a shared definition of a horizontal Service Layer in literature, but some common principles can be identified: logically unique functions; open interfaces and adoption of open protocols APIs detaching from actual enables; unique identification of customers and uniform access to data; uniform mechanism for the exposure of Telco features to IT applications; adoption of a common communication infrastructure (e.g., CORBA, .NET, etc.).

So-called Service Delivery Platforms [2] are examples of solutions which implement such principles but, at the moment, these are mainly proprietary proposals elaborated by vendors, based on integration of their products and of products of their partners. It is worth mentioning that several standards are being proposed providing specifications which can be used to deploy multi-vendor SDP-based solutions. Examples are: OSA/Parlay APIs and Parlay X Web Services, 3GPP GUP and Liberty Alliance Frameworks, JAIN SLEE specification provided by JSP for event-oriented service execution, and OSE, the architectural vision of OMA.

The most promising trend in this direction is the evolution of the service layer according to a horizontal approach is the adoption of Web Services and Service Oriented Architecture solutions [3]. Recent initiatives, e.g., proposed by OMA and Parlay, are aiming defining a Web Service/SOA Framework for Telecommunication Services, typically relying on existing and newly proposed standards (defined in W3C, OASIS, WS-* initiative).

A SOA-based Service Layer model typically includes sets of macro-functions for execution of loosely coupled services, service exposure, specialized functions for Telco Web services (possibly based on emerging standards such as 3GPP/ETSI Parlay X Web Services, Liberty Alliance WSF, OMA service enabler interfaces), and functions that enable the deployment and the delivery of services based on Telco features (other than basic connectivity, enables for Multimedia, WAP, location services, etc). In addition, it includes infrastructural services such as directories, SOA communication buses, and identity/access management services.

Generally, SOA frameworks are made independent of specific Telco protocols, and are (at least logically) decoupled from the servers implementing the Telco Features (e.g., JSLEE introduces several adaptors to interact with different protocols/interfaces). This makes SOA frameworks more suitable for enabling integration and convergence than vertical frameworks and other proprietary horizontal frameworks.

Still, in these frameworks, service logic is mainly "centralised" with limited distribution of business logic components: components (e.g., exposed as Web Services) are mainly used to access basic functions (e.g., for controlling Telco Features) and not to structure the application business logic. Moreover, the configuration and composition of service logic are static with limited dynamic adaptation to the service execution context. Finally, the deployment of applications/components on servers is performed according to some off-line planning without limited dynamic adaptation to traffic, performance, fault-handling conditions.

Such characteristics inherently limit scalability and the capability of handling large-scale service and data systems, limit the possibility of exploiting such frameworks for the dynamic deployment and execution of innovative, user-centered, and adaptive Telco services (also by Prosumers). Also, the need for specifically configuring application logics makes any attempt in that direction technically hard and economically unbearable.

3. DSF OBJECTIVES

The DSF model presented in this paper aims overcoming the limits of current solutions for telecommunication services by proposing a flexible distributed services framework, relying on fully distributed and self-chop solutions for services, and on a flexible peer-to-peer overlay as a substrate for their execution.

This will enable: dynamic organization/adaptation of service behaviour according to the evolution of service sessions, through the organization of the service logic in smaller independent components with autonomic behaviour; optimization of resource allocation and higher scalability properties through flexible/ dynamic allocation of services,/components to the servers; handling of immense volumes of highly-dynamic data/ information, as required by pervasive/mobile Telco-ICT services, through flexible organization of data management.

Moreover, DSF will lead to a reduction of management/ configuration costs via a uniform autonomic model, a reduction of the costs involved in hardware resources, and will open up the possibility for innovative business models, through service development, configuration, execution, assurance open to diverse set of actors (e.g., prosumers-ecosystem).

Limitations of State-of-the-Art	DSF objectives
Configuration and composition of service logic are static with limited automatic capabilities of adaptation to the service execution context	The organization of the service logic in smaller independent components, with autonomic behaviour, enables the dynamic organization/adaptation of service behaviour according to the evolution of service sessions
Limited scalability in handling data intensive applications, requiring a real-time view of the huge volume of highly dynamic information, as required by pervasive/mobile Telco-ICT services	Flexible organization of data management, by using autonomic and semantics based information, and distribution of data across computing nodes, dynamically allocated, allow handling immense volumes of highly-dynamic data/information
Static allocation services-servers, according to off-line planning, traffic estimation, etc.; "monolithic" allocation of the services to the servers; some proprietary solutions available for single vendor server farm	Resource allocation through flexible/ dynamic allocation of components to the servers, also according to autonomic decisions and exploitation of GON capabilities, optimize the usage of the resources (through an allocation on demand), and improve scalability in terms of both processing and data management
Focused on the development of services in the domain of service providers	Uniform governance of DSF across heterogeneous computing/network contexts, and autonomic support for service management, enables diverse types of actors (e.g., prosumers-ecosystem) to offer services and facilitate the coalitions of service providers

4. DSF ARCHITECTURAL OVERVIEW

The DSF is deployed on an heterogeneous set of resources which ranges on common-of-the-shelve (COTS) servers, Users' devices (e.g., PDA, mobile phone, laptop), Customer Premise Equipment (Residential Gateways, Step Top Boxes), sensors (or their aggregation proxy), etc. These nodes are interconnected through a, potentially, heterogeneous communication infrastructure (e.g., Internet): some of them could have dynamically assigned addresses, or may be allocated behind NAT/firewall nodes.

In fact, DSF has to be deployed on both end-users' terminals/devices and servers/clusters in the service providers' premises, in order to provide a platform homogenous from the point of view of services and components: this environment would contribute to achieve seamless integration of the Prosumers systems in the overall architecture.

The overall architecture may enhance the self-similarity properties of Google's Googleplex solutions (both pizza-box servers and clusters of servers have the same functional architecture) supporting a distributed replication of data and services: this would allow high levels of availability also starting from low-cost commodity H/W.

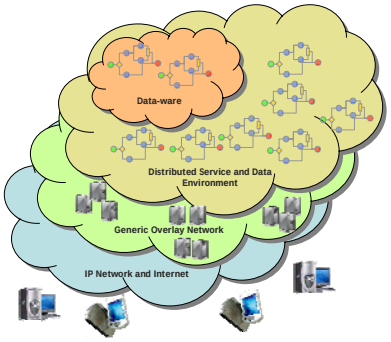

Figure 1: Distributed Service Framework: Architectural vision

The Generic Overlay Network (GON) is a layer on top of the computing resources and the Internet governing interactions between the DSF functions and services.

GON aims decoupling the service execution functions (implemented by the DSDE) from the underlying infrastructure, as a virtualization of the infrastructural resources and communication/networking among them. GON organizes the computing nodes (either end-user devices, servers, or servers clusters), interconnected through a potentially heterogeneous communication infrastructure, in a set of peers. GON provides the basic functions to build and maintain overlay networks, by handling also the dynamic and unpredictable failures of computing nodes or introduction of new ones. GON has to provide a "generic" support to distributed service execution: it is not tailored to a specific application, as the most known current overlay solutions (e.g., by Skype for VoIP services, GIA for content distribution).

The Distributed Service and Data Environment (DSDE), based on autonomic agent/component-ware technologies, will be in charge of executing, maintaining the services (e.g. functions as service composition, execution, management, assurance...). DSDE provides autonomic capabilities for self-organization, self-adaptation, and self-management (e.g., self-chop) to the service components required to implement the service business logic.

DSDE functions (i.e., execution environments for autonomic components/agents, distributed data repositories, etc.) are deployed on the GON peers, either on servers/clusters or on end-users devices/terminals (possibly belonging to different domains): moreover, DSDE relies on the functions of GON to build, optimise and maintain the virtual connectivity necessary to handle data and execute services in a distributed, reliable and scalable way.

Moreover, DSDE includes functions/components to implement Data-ware capable of gathering, caching, correlating (by also exploiting semantics knowledge) and offering huge amounts of data/information from/to different data sources/sinks (e.g. mobile handsets, distributed devices, etc) of which the availability is changing dynamically (i.e. high churn rate of resources). Data-ware functions would enable to deal (retrieve, store, access, protect, etc.) in a uniform way data/contents managed by Service/Content Providers and data/contents handled by Prosumers.

DSF has to supports the interworking with legacy systems such as service enablers from legacy service delivery environments (e.g. Location, Presence and Group List Management, Messaging Call/Session Control, Conferencing), OSS/BSS systems, Data-Centers, Contents/Media-Repositories.

These interworking should be performed in order to allow DSDE components/services interacting with the legacy systems, in a seamless way, according to the adopted component-ware paradigm. For instance, the interaction with OSS/BSS systems should be mediated in order to be integrated in the context of the autonomic self-management capabilities.

The Distributed Service and Data Environment (DSDE) has to provide the environment for the execution of services structured as distributed components: these components include autonomic capabilities for self-management, self-action and self-organization. Moreover, the DSDE has to be capable of handling and organizing huge amounts of distributed data/information.

The DSDE responds to the key evolutionary requirements with respect to the current solutions of service execution environments, typically adopted for the delivery of services over ICT/Telco infrastructures. Such current solutions, mainly based on "centralized" service logic deployed on application servers, are based on servers running execution environments either compliant to international standards (such as J2EE, JSLEE, BPEL), or proprietary solutions (in particular for services in the Telco context, such as Intelligent Network SCPs of new generation). In fact, in spite of introduction of SDP and/or SOA inspired service layers, the business logic of the services still remains mainly "centralized": at most they interact with components to control Service Enablers or to interface management systems.

In order to overcome the limitations, DSDE adopts an execution model based on interacting distributed components/agents with goal-oriented logic capable of providing a rich set of autonomic capabilities. According to view, service logic (or, better, an eco-system of service functions/components) is provided through a web of communicating autonomic components/agents, which are executed in a distributed way and (self-)managed through a set of self-* features. For instance:

- the organization of the service logic in smaller independent components, with autonomic behaviour, may offer the possibility to dynamically organize/adapt service behaviour according to the needs of service session instances;
- the allocation of services, and service components instances to the servers in a more flexible/dynamic way, possibly according to autonomic decisions taken by services/components, allow significantly optimized allocation of the resources: servers may be allocated to services/components on-demand (e.g., due to service request picks, QoS requirements, facing fault situations), and released (to be allocated to other services) when no longer needed;
- the distribution mechanisms, also based on autonomic features, for service logic and tasks increment the scalability of the solutions, for instance to monitor huge numbers of client/consumer devices to detect service triggering based on dynamic conditions (e.g., based on crossing boundaries between geographical areas); moreover, an efficient distribution of service/task execution improve the handling of sessions (determined by "long-running" Telco protocol transactions);
- the flexible organization of data management (e.g., for gathering, aggregating, inferring, and making it available dynamically where and when needed) and data distribution across on a (possibly dynamic) set of computing nodes enables the handling of data huge volume of highly dynamic data/information.

DSDE is organized as multiple/distributed instances deployed on peers. Each of the instances implements functions the execution of distributed autonomic components/agents and/or for the flexible management of huge amount of data.

In order to work on a set of dynamically available computing nodes interconnected, the DSDE will exploit the capabilities provided by the GON for handling the (dynamic) allocation of DSDE peerws on computing nodes, according to the performance and fault recovery needs. This solution would improve scalability and high-availability at a reduced cost w.r.t. to traditional solutions (based on the static allocation of execution environments on server farms).

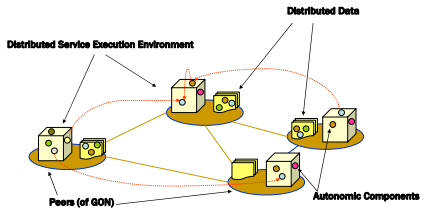

Figure 2: Distributed Service and Data Environment

Technologies for autonomic behaviour and communication of the service components provide the core capabilities for flexible management of the DSDE: solutions elaborated in Project CASCADAS [4] are a possible starting point (the component model, the related supervision infrastructure, the guidelines for self-organization, and data management functions).

As an example the following features can be implemented with goal-oriented approach:

- Self-action features that allow a component/agent to behave proactively, i.e., to act autonomously without the persistent necessity of human intervention. Such distributed decision-making may be offered by an individual component/agent, or by consensus agreement, and helps to enable the automation of selected tasks traditionally performed by human operators. This forms a necessary precursor to enabling self-management and self-organization;

- Self-management features that employ autonomic feedback loops to provide self-adaptation according to detected events and conditions (and considering also resource availability). They include self-configuration to dynamically change service configuration, composition, and distribution/ deployment according to the evolution of the execution conditions, self-optimization to optimize the behaviour and operation of service components and their allocation/configuration on the execution environment, and self-regulation to compensate for change and failures in the DSDE and thus offer dependability and resilience;

- Self-organization features that allow the dynamic and automatic assembly of the functions required to accomplish a given task or set of tasks according to component-local rules and behaviours. Sustainable system properties are supposed to emerge for self-organized behaviour of an interacting network of basic components. These features could be enriched with resource negotiation capabilities.

- Self-similarity that addresses the derivation and deployment of general management and organisation mechanisms to be applied on various levels of abstraction, thus supporting the autonomous formation of hierarchical structures in service eco-systems that are necessary for a scalable definition of the DSF.

- Self-security that addresses all functions that are concerned with the establishment of trust relations between users and systems, or between systems. It includes classical security solutions such as authentication and encryption mechanisms as well as more advanced “soft-security” concepts such as the dynamic trust and reputation management, community based security, etc. It would include a module to continuously monitor potential DoS attacks for any component of the DSDE, and to offer dynamic defense mechanisms. The optimal incorporation of this module into key DSDE components will be a topic for research and experimentation, leading to a technical solution that will offer a compromise between possible overhead during normal operation, and enhanced performance in the presence of attacks.

The autonomic features of the DSDE are improved through the introduction of goal-oriented knowledge reasoning capabilities relying on ontologically-grounded data semantics, both at the level of component lifecycle management, and at the level of component interaction and aggregation. The use of semantics as an integral part of service descriptions allows a more meaningful association between the data or knowledge rendered accessible by a service, and the programmatic expression of that data within the service description. Assigning semantics is crucial for the management of huge amount of data; DSDE will also exploit semantic information to express the underlying concepts to aggregate, combine, delete, or re-distribute data items.

DSDE includes mechanisms for the flexible organization of data management (e.g., for gathering, aggregating, inferring, and making it available dynamically where and when needed), data distribution across on a dynamic set of computing nodes, and support for rich set of data access functional requirements (e.g., currency, consistency and integrity, version, preserved distributed isolation, persistence) as well as data access non-functional requirements (e.g., latency, throughput, security, resilience and recoverability, mobility location and speed, scalability); this data management organization, data access requirements richness and data distribution would enable the handling of immense volumes of highly dynamic data/information, required to address pervasive Telco-ICT services (e.g. systems for environmental monitoring, human life support, emergency and traffic management, etc.).

As already described the target environment is characterized by high degree of dynamicity and high probability of unpredictable events, changing conditions and disconnections. This motivates one of the basic principles of the architecture: each building block has to provide the information/capabilities it is responsible for to other participants as much as possible and any time it is capable to do it. The right metaphor is the “ServiceSpace”: a shared environment each participant contributes with its knowledge and its capabilities.

This principle avoid to implement strong and heavy controls needed to ensure that the right information reach the proper

consumer when needed. If a component is available and it is capable to reach some other participant, its information is provided.

The principle described above, if applied alone, could lead to poor performance due to the huge amount of information circulated by the "altruistic" participants. In order to mitigate this effect the class of autonomic algorithms described above. The main goal for the algorithm in this case is to ensure the proper connection among participants in order to ensure that each participant is altruistic with the best group of other components.

The basic assumption underlying this mechanism is that if any participant tries to do its best job in terms of proposing its capabilities to other participant, and do this any time he can, we can achieve significant benefits at whole system level. This will lead to a "ServiceSpace" characterized by the proactive behavior of the single component: each participant, apart from its main business goal, has the main aim to contribute to the information building the "ServiceSpace". This information is statement about the proposed service.

The metaphor for the "ServiceSpace" is the *blackboard*: each component runs its service logic when needed and each time it is idle write proper information in the "ServiceSpace" in order to state its availability. The information about the services proposed is obviously augmented with further information aimed to allow its proper invocation.

The "ServiceSpace" will lead to a very useful view of the system which will state the status of the system in terms of availability of services, their locations and condition to invoke them.

5. CONCLUSIONS

This paper has provided some preliminary results of ongoing RT&D activities on the so-called Distributed Service Framework (DSF), a solution that has been conceived as an innovative open eco-system for ICT/Telecommunications services. DSF architecture is based on multiple autonomic components which interacts each other via a self-organising overlay.

The design of the DSF has devoted particular attention to the service execution environment of the DSF where scalability and dynamic load balancing are met by self-organization capabilities of the components. The design choices, specifically the autonomic and altruistic local behavior of components and "ServiceSpace", enable global features effortlessly, simplifying implementation and management complexities.

6. REFERENCES

[1] R. Minerva R., C. Moiso, "The Death of Network Intelligence?", in *Proc. International Symposium on Services and Local Access 2004 (ISSLS)*, Edinburgh, 2004.

[2] The Moriana Group, "Service Delivery Platforms and Telecom Web Services - an Industry Wide Perspective", (http://www.morianagroup.com/sw1635.asp), June 2004

[3] E. Thomas, "Service-Oriented Architecture: Concepts, Technology, and Design". Prentice Hall, 2005.

[4] A. Manzalini, F. Zambonelli, "Towards Autonomic and Situation-Aware Communication Services: the CASCADAS Vision", in *Proc. IEEE 2006 Workshop on Distributed Intelligent Systems.*