Deploying telecom-grade products in the cloud

Gone are the days when the cloud was simply considered to be extra – and unreliable – computing capacity. Today it has developed to become the center of an established business model in which global applications are hosted, resources are efficiently managed and economies-of-scale created. The cloud as a platform is now ready for the next level, being remodeled to host critical services.

Cloud computing has come a long way from simply being virtual data warehousing – the shapeless blob used to illustrate a network owned by another operator or organization, or to represent unknown architecture. Today’s cloud solutions manage anything as a service (XaaS). This includes anything that an enterprise, government, organization or individual might need to go about their day. Cloud solutions house applications, databases, services, software, test environments, financial platforms and back-office systems. The cloud approach brings high accessibility through thin clients and apps, resource sharing, scalability and recovery – all while providing economies of scale and pay-as-you-go pricing. However, this maturity and readiness to challenge new requirements results in a need for more stringent security demands.

As the cloud matures, the one-size-fits-all model of telecom products is also in the process of evolving, moving toward product diversification to match individual subscriber requirements and tastes. To achieve individualized product offerings, a much higher degree of automation is needed to define products in such a way that they do not conflict with each other, can ensure a short time to market, and can be assigned with appropriate Service Level Agreements (SLAs) that can be upheld without requiring manual intervention from an operational center.

Cloud management

Cloud solutions have matured to the point of becoming an attractive option for hosting complex and mission-critical systems. The OSS and BSS functions required to define, allocate, supervise and monetize cloud resources, as well as implement some of the key aspects of the cloud (automation, self-service, and on-demand resource management) are collectively referred to as cloud management. This is one of the essential components of the maturing cloud model. However, the level of automation in mainstream cloud-management solutions tends to be limited, supporting just the basic building blocks of a deployed service: virtual machine (VM) management, software installation and configuration of some aspects of storage and networking. Tasks such as securing optimal placement of workloads, fulfillment of SLAs and resource reallocation are, for the most part, still performed manually, with limited support from the underlying management system.

The lack of automation is a significant barrier to developing and delivering complex services – ones that require multiple resource types and need to meet stringent QoS requirements. To overcome this barrier, cloud management needs to be enhanced with innovative technologies that automate tasks such as reallocation of resources and load balancing. Ericsson’s concept of model-based service definition and automatic resolution of SLA constraints is a step toward increased automation.

Functional architecture

The functional architecture of a cloud-management solution is illustrated in Figure 1. The typical OSS/BSS components can be reused in the cloud model as follows:

**Planning**

This component defines cloud services, including dimensions for price, campaigns, and SLA characteristics.

**Fulfillment**

This component creates, sets up and deploys applications and services hosted by the cloud. It uses a PaaS/IaaS approach to employing platform
resources (such as life-cycle management and embedded cloud functionality) and infrastructure resources, such as virtualized computing, networking and storage.

Assurance
This component supervises and, if possible, automatically adjusts allocated resources. Typical aspects of service assurance are monitoring, SLA enforcement and fault management.

Security
Due in part to the multi-tenant nature of cloud solutions, security functions in the cloud context are more important than they are in traditional data warehousing. They play a vital support role in SLA management, and mandatory security aspects of any cloud-management solution include identity control, access management, logging and monitoring, auditing and compliance.

Charging
This component implements the pay-as-you-go model of cloud computing, by generating the corresponding charging records for external billing systems.

Deploying complex services
Over the past decade cloud-based offerings have evolved constantly, becoming more complex and highly integrated. Today, the dependency hierarchy of most cloud applications and services is non-trivial. A cloud service may, for example, use another mash-up, rely on a networking feature or be dependent on a web service provisioned by another vendor. Failures or delays that occur in one service can impact multiple dependent services, creating situations that can become difficult to manage or recover from manually.

To reduce the knock-on effect of failures in highly dependent systems, current research and development is moving towards building platforms that manage this complexity through automated creation, composition, publishing, marketing, bidding, contracting and revenue-sharing functions. The next step in the evolution of the cloud model will support portable self-described cloud applications and services that use model-driven configuration, interaction and management.

Telecom-grade and other regulated services place strong requirements on end-to-end performance and non-functional parameters, such as latency, throughput, localization, security, availability and reliability. For SaaS deployments, such requirements need to be translated into cloud-platform SLAs that guarantee sustained and verifiable end-to-end performance.

Models
Service Delivery Platforms (SDPs) have been used to great advantage in communication services for life-cycle management. These platforms generally include a framework for creating services, an order/fulfillment subsystem, and an execution platform to host communication services. An SDP enables products to be created from groups of resources, value-added services, multiple configurations and billing plans. By picking and mixing in this way, communication service providers can use an SDP to define and offer services with a short time to market.

High availability, elastic scalability and provision for disaster recovery are among the key benefits of a cloud deployment. To maximize these benefits, complex and QoS-sensitive applications need to be specified within the service creation environment and SLAs need to be handled automatically at the fulfillment stage. In other words, by extending the space where services are created, complex applications can also be made scalable, available and suitable for cloud deployment.

The Ericsson model
To create value-added services, Ericsson’s concept of an extended creation environment for cloud services is based on modeling the characteristics of a virtual data center (VDC) and virtual zones. The concept models applications and services in terms of logical components that are available in a palette, where embedded integrity and validation rules are used to guide the composition of cloud services and avoid any invalid combinations.
Automating the cloud

One of the challenges of this approach is to identify the right level of abstraction in the service-creation environment - somewhere between a business abstraction and a technical specification - so that the service can be monetized and meaningful SLAs can be defined from a user perspective. For example, if a service includes disaster recovery, this may be implemented through geographic replication. However, the service users only need to know that several levels of backup are available with sliding payment packages; they need not be concerned with how recovery from disaster is achieved.

The typical service model of a triple-play product enhanced with a VDC cloud offering is shown in Figure 2. The basic building blocks are specification components for resources, services and products, which allow capabilities to be described in a modular way:

- **resources** - network, storage and computational resources are the basic functions of the platform, they support the execution of services and need to be allocated quantitatively;
- **services** - represent an identified piece of basic functionality (such as SMS) or value-added services such as internet access or voice; and
- **products** - provide a complex end-to-end service with defined business and billing characteristics, such as a package including free voice calls in the home network, free SMS, and mobile data provided at a flat rate.

As shown in Figure 3, an SLA can be applied as an additional constraint at any level. When a service component is deployed in the cloud, a description for it is created, which includes:

- **basic properties** - an informal description;
- **offering** - the functions that are included, used to construct the dependency tree;
- **characteristics** - fixed properties, such as memory or processing consumption needs, and variable, deployment-dependent properties such as price or location;
- **configuration** - information needed for service instantiation, such as installation scripts and download URLs;
- **dependencies** - other service components that the service being described depends on; and
- **constraints** - that describe non-functional requirements - the SLA core.

In the Ericsson concept, SLAs can specify both traditional networking aspects, such as bandwidth and delay, as well as non-functional aspects, such as processor speed, available memory, placement of VMs at specific locations in the network topology, and the cost associated with a certain component.

For traditional networking aspects, SLAs are defined in terms of specific dependencies, such as the geographic location of a given service component or the throughput that a particular network component should provide. The Ericsson concept extends the SLA definition mechanism to include aggregated constraints. For example, a service provider is more likely to be interested in the total price of a service including its dependencies, or the total (aggregated) reliability of a service from an end-to-end perspective, than the cost or reliability of its individual components. The freedom to mix and match potential dependencies without restraint, as long as the aggregated SLA is fulfilled, optimizes the service-selection process. The aggregation mathematical function also needs to be specified, which in the

---

**FIGURE 2** Model-based service definition

[Diagram showing service model components and process flow]

**FIGURE 3** SLA management

[Diagram showing SLA management and dependencies]
case of delay and pricing is addition, and for availability is multiplication.

Each service component has an associated process component, which governs its life cycle from activation through assurance and decommissioning. For example, the Extended Messaging Service (EMS) resource is associated with the procedure for activating internet access over Wi-Fi. At order time, all process components are collected to form a service-composition plan – macro flow – which includes a dependency structure that specifies: when various process components should be invoked; what data is defined as input to the component; and the possible set of logical operations that can occur between components.

The SLA resolver ensures the SLAs are upheld. At order time, the creation environment sends the order description – the top-level service dependencies and SLA aspects chosen by the consumer – to the SLA resolver. Based on the available service components and their characteristics, the resolver automatically creates the entire service tree and deployment plan for use by the fulfillment component.

**SLA management**

To fulfill end-to-end SLAs, they must be translated into resource-related requirements on the cloud infrastructure. This is more challenging in a cloud context than in traditional data centers as service component characteristics vary. They may be allocated in different locations, accessible over different network links or share infrastructure with other applications dynamically.

Manually constructing a cloud-deployment graph that defines which services to use and how and where to create the necessary VMs and network links is a complex and maintenance-intensive process. An automatic process that builds the dependency tree based on the service model and resolves SLAs into resource parameters suitable for instantiation would be very beneficial. Such an automated process – the SLA resolver in the concept – uses a resolution engine to settle constraints and build the dependency tree. Starting with the top-level service description, and including the SLAs agreed with the consumers of the service, the engine chooses the services that match the dependencies and fulfill the constraints. The resolver algorithm is applied iteratively to each of the dependencies and employs backtracking when a constraint is not fulfilled.

With a fully built resolution tree, it is possible to identify the components needed to support a service and then group them into a virtual node. The resources needed by the virtual node to host the allocated software stack can also be computed automatically. To prepare for infrastructure allocation, the resolved service-description tree is packed into an abstract service map (ASM) consisting of nodes, each with its own software stack and network links. The ASM is handed off to the fulfillment function, which determines the available physical infrastructure, based on information stored in the configuration management database (CMDB). Once the necessary VMs are instantiated, the provisioning mechanism uses the VM’s service description sub-tree to deploy the software stack in the right order and according to the configuration scripts specified in the service description.

![FIGURE 4 Example of service model with SLAs](image-url)
requires a three-second delay and the sum of the delays of the dependencies is less \((1.8 + 0.5 + 0.2 + 0.2 = 2.7)\). The corresponding ASM and mapping to the infrastructure is shown in Figure 5.

**Conclusion**
Innovative mechanisms for automatically managing complex cloud services and associated QoS and other SLA requirements are needed to take the cloud model to the next level, and bring increased automation into the underlying architecture. Two Ericsson-developed technologies, model-based service definition and automatic resolution of SLAs, will help bring about automation in the cloud by extending the service-creation environment and supporting service providers to create non-conflicting, differentiated offerings with short times to market and automatic SLA fulfillment.

**András Vajda**
is an expert in cloud computing and cloud management within Group Function Technology and Portfolio Management. He joined Ericsson in 2001 working within Ericsson Research and Business Unit Networks. At Ericsson Research he was one of the initiators of Ericsson’s cloud-research efforts, driving the strategy and architecture work. He holds a B.Sc. in mathematics and computer science and a master’s in distributed computing from Babes-Bolyai University, Cluj-Napoca, Romania.

**Jan Sölvhammars**
is product manager for OSS products at Ericsson’s Business Unit Support Solutions, where he has been working with various software products for multimedia applications and OSS. He joined Ericsson in 1990 and has over 20 years of product-management experience in different areas. He has worked with radio-network products for many years and holds an M.Sc. in industrial engineering and management from Linköping University, Sweden.

**Francesco Caruso**
is a principal cloud architect within the Cloud Infrastructure Management group at Ericsson’s Business Unit Networks. He joined Ericsson in 2012 from Telcordia Technologies, where he was director of the Enterprise Integration Group. He championed the internal cloud program to transition OSS to the cloud environment and to extend the OSS into the cloud-management domain. He holds a B.Sc. in computer science from the University of Pisa, Italy, and has more than 15 years of expertise in the telecom OSS domain.

**Calin Curescu**
is a senior researcher with the Services and Software department of Ericsson Research. He initiated the research on SLA-driven management of cloud services, and has worked with cloud computing, service composition and network exposure. He holds a Ph.D. in computer science from Linköping University, Sweden.