TECHNOLOGY WHITE PAPER

Please carefully read the instructions in this document and comply to them, in order to avoid us returning it to you for completion. Thank you.

Article id Name Version

Contact table

Contact table	Name	Internet - e-mail address
Lead author	LARVET Philippe	Philippe.Larvet@alcatel.fr
Contributor(s)	BONNIN Bruno	Bruno.Bonnin@alcatel.fr
Editorial team	Sandra Cornet-Vernet/Lehongre Myriam BLOUIN	Sandra.cornet vernet-lehongre@alcatel.fr Myriam.Blouin@alcatel.fr

Title

Please make your title attractive and not too long (8 words is already a lot), avoid obvious advertising words such as "leading-edge", "top notch", "innovative", "break-through", etc. : they have worn out and create an adverse effect. Create something simple, but rich in terms of content.

Your title : **Instant Delivery of Blended Composed Semantic Services**

Headline

One punchy sentence of about 15 to 25 words to further arouse the curiosity of readers.
This will used on internet and intranet pages as a teaser for readers to learn more about the article.

Your headline : A user-centric solution for instant building and delivery of new services composed with existing Web services discovered and assembled on-the-fly.

Abstract (100-120 words)

The abstract serves as a tool to **sell your point-of-view** covered by the article to a large audience, and not simply to write a summary. Get to the point right away! Use a captivating style.
Be aware (and accept) that **the majority of readers will only read the abstract**. Even if interested, decision makers have little time to spend on this, so if they think they got the point through reading the abstract, they'll move one to other matters. So it's important to make sure that the main messages get across in the abstract. Do not use it as a teaser for the reader to go through the entire article in order to only have him find out your point of view at the end of the article!

In order to facilitate the on-demand delivery of new services for mobiles as well as for fixed phones, we propose a solution based on Service-Oriented Architecture (SOA) within the context of Alcatel's Open Service Delivery Environment (OSDE). This solution is made of three main mechanisms: a ***service discoverer***, transparent for the user, allows to find the pertinent Web services matching with the user's original request, expressed vocally or by a SMS or a simple text ; a ***service composer***, based on semantic descriptions of the Web services, allows to combine and orchestrate the discovered services in order to build a new service fully matching the user's request, and a ***service deliverer*** makes the new service immediately accessible by the user.

1. Introduction (250-300 words)

The introduction sets the stage for the technology topic in a "market" context. It recaps the state of the art of the technology and/or the status of results obtained so far.
Emphasize the importance of what you are going to demonstrate or develop:

- Definition of the emerging technology
- The audience and needs the technology addresses, especially in the context of Alcatel's vision of a User-Centric Broadband world
- A brief analysis of the technology issues that will be presented.

Web services, as they are easily accessible from any point of Internet through an Application server, are suitable to build rapidly on-demand applications.
From the point of view of their internal complexity, Web services (WS) can be divided into two families : *elementary* WS and *composite* WS.
Elementary WS provide a basic service, comparable to mathematical libraries, and contain a low level of data transformation, embedded in few algorithms ; for example, translation services are elementary WS.
In the contrary, composite WS are able to provide a high level service and contain many levels of data accesses and transformations, given by the cooperation of several elementary services. Such a composite service, resulting of the composition of several *processes* logically assembled, can be called an *orchestrated* service ; for example, reservation services or secured-payment services are composite orchestrated WS.
If the development of elementary WS is obviously the domain of specialized software engineers, the emerging technology of *semantics*, associated to web services, allows a end user, through an appropriate application accessible from his preferred phone (mobile or

fixed), to build *transparently* and to use some new services made by an assembly of existing elementary WS, discovered and orchestrated on-the-fly. "Transparently", because the process of discovery, composition and delivery of the new service has to be totally transparent for the user whose the only demand is to get a new easy-to-use service immediately responding to his original easy-expressed request.
A second advantage of the semantic technology attached to WS is to allow keeping only the semantic descriptions of the requested services, instead of the orchestrated resulting service itself, because the semantic descriptions can be processed and re-processed at any time, permitting a *dynamic* re-discover, re-orchestrate and re-deliver of a "new" new service, matching better with the original request.
Alcatel's Open Service Delivery Environment, as it is an open environment for development, repository and delivery of services, is the privileged architecture for building and holding the three main modules that implement our discover-orchestrate-deliver concept.

2. Body (1900-2000 words)

The outline of the body section follows what was briefly covered in the introduction.

- Describe shortly the current state of a technology and its inherent limitations, especially in achieving a User-Centric Broadband world.
- Define a new technology
- Describe in detail, but in understandable, tutorial style what this new technology/solution is all about
- Demonstrate how the new technology/solution is superior to the existing ones (use simulation results, benchmark data, or any other evidence which helps underpin the value of the new approach.

Keep a neutral, objective writing style.

Some basic editorial rules:

- No more than 3 heading levels
- Each figure must have a caption

Please refer to the guidelines for more details.

Easy-expressing new services: concrete use cases and benefits

Let us develop a concrete use case to explain the functioning of our solution. For example, a user wants to get a french translation of the latest CNN news.

The user simply activates the "instant delivery" function of his terminal, and expresses vocally his demand : "I want to get a french translation of the latest CNN news". He could also send an SMS containing this short text to the address of the Instant Delivery Service. Indeed, there is no easier way to express a requirement !

The Fig. 1 allows you to follow the building and delivery of the final required service from the original request.

The vocal message (flow 1) is taken into account by the Discover-Orchestrate-Deliver Mechanism of a given Application Server. Let us be for the moment a little generic about this Application Server; we will see in the last part of this article how the Alcatel's Open Service Delivery Platform (OSDE) could take care of the capabilities of this server.

A fisrt block, Speech-to-Text (see Fig. 2), "understands" the message and transforms it into a text, that is processed by a Semantic Analyzer. If the user has sent a text (flow 2) instead of speaking a vocal message, this text directly enters the Analyzer that extracts from the text content the needs for the intermediate elementary required services. In the case of our example, the needs are:

- french translation
- latest CNN news

These needs are kept in the Application Server under the form of "semantic descriptions" of the intermediate required services and stored in a specific Semantic Repository (flow 4).

The stored semantic descriptions are processed by the Discover module (see fig. 2) that searches in a public Semantic Services Repository (flow 6) some service descriptions that match with the required stored semantic descriptions.

The corresponding elementary web services (for example a web service Translator and a web service CNN News), fitting with the required semantic descriptions, are passed to the Orchestrate module (flow 7), which makes the composition of the elementary web services, using the orchestration handlers contained in their semantic descriptions.

The resulting composed service (flow 9) is taken into account by the Deliver module, that stores the new service in the Service Repository of the Application Server (flow 10) and registers in the public Intranet/Internet Semantic Services Repository the semantic description of the new service (flow 11).

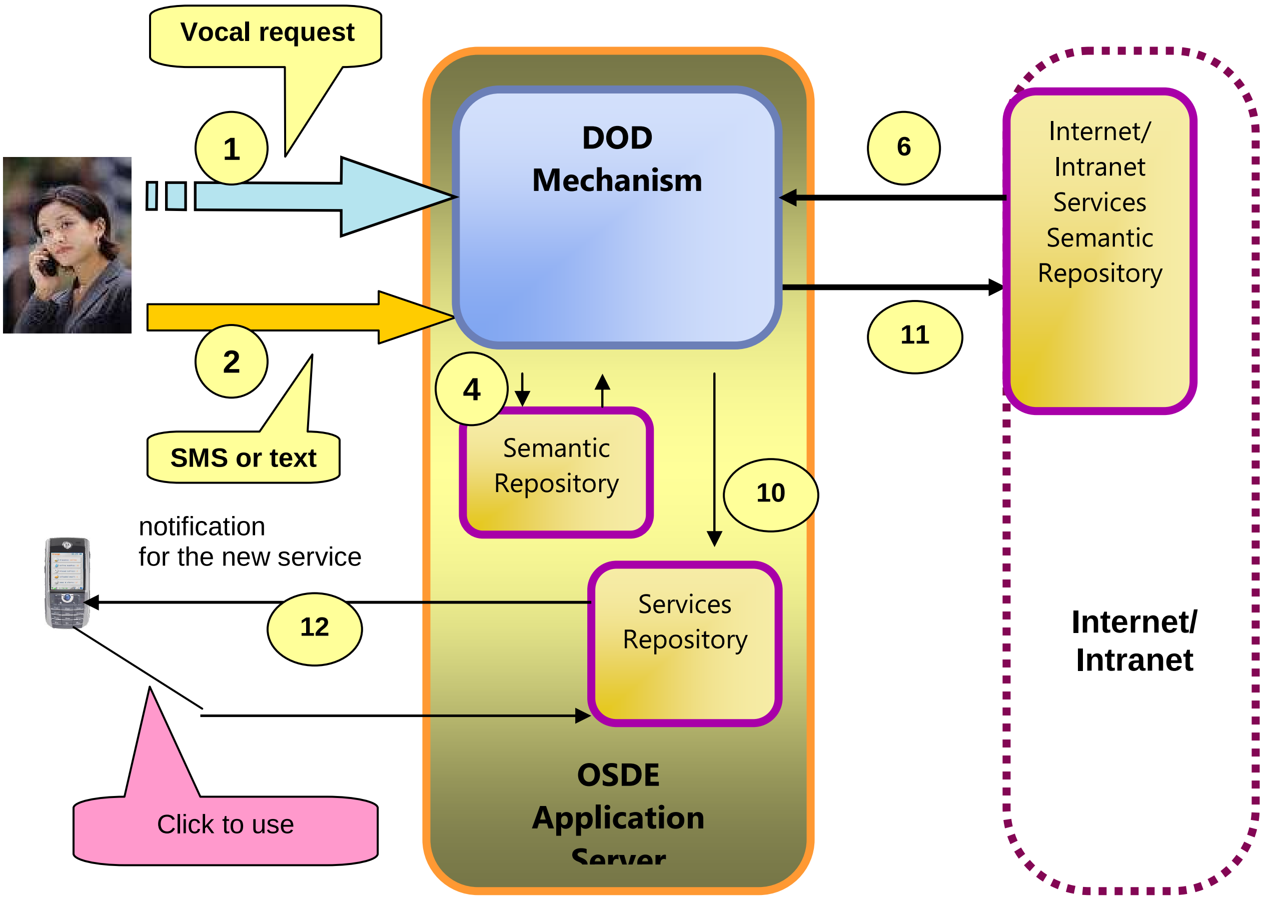

Figure 1 – Processing an Instant Service Delivery use case (global view)

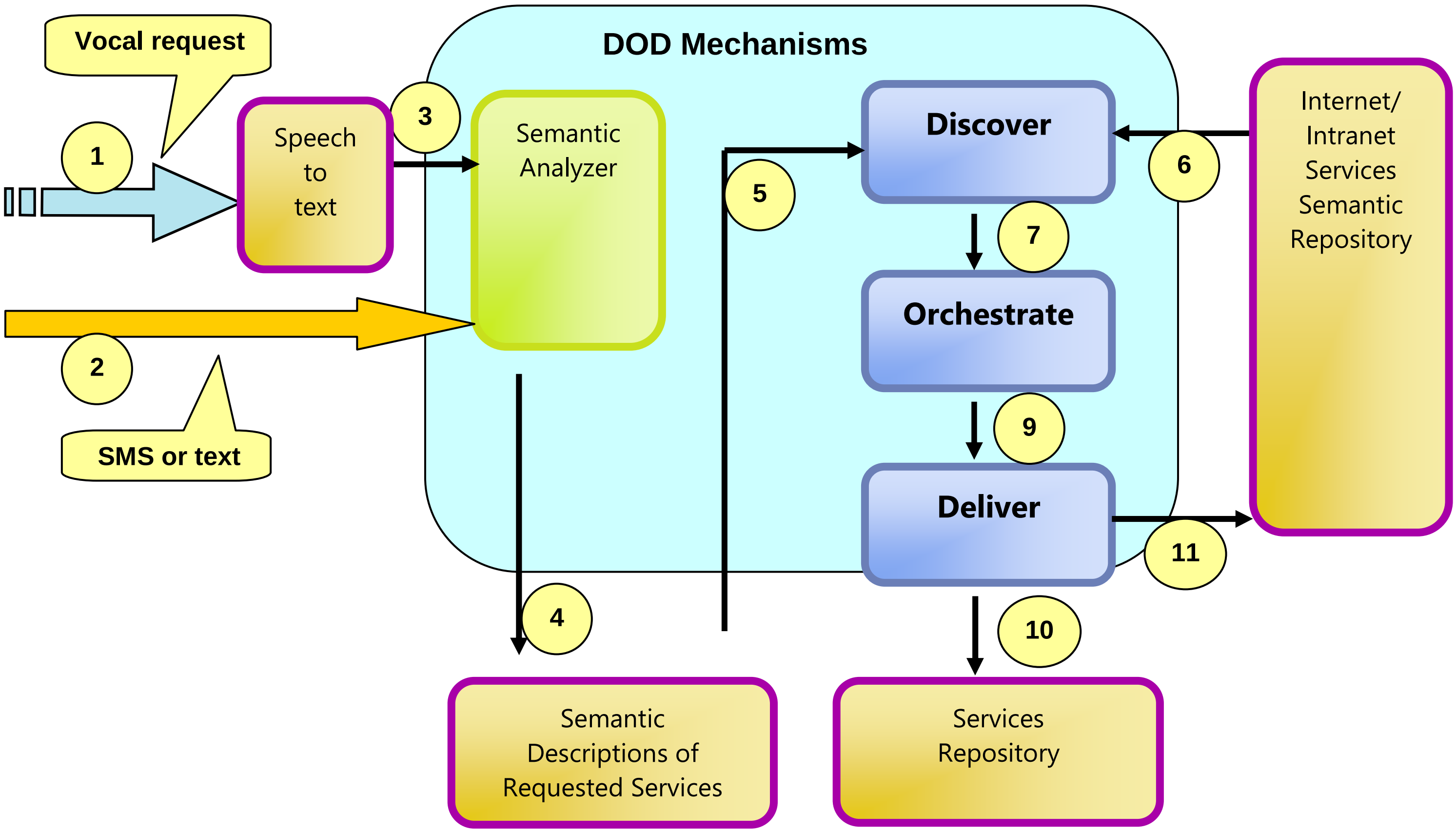

Figure 2 - Processing an Instant Service Discover-Orchestrate-Deliver use case (detailed view)

The Service Repository has a local intelligence allowing to send to the user (and, occasionally, to every user who previously subscribed to this service) a notification (flow 12) informing him that the new asked service is now available on his preferred server.
Then, and this is the end of our use case story, the user can immediately activate and use the new service – and to get a french version of the latest CNN news.
Beyond the obvious benefit of this process for the user – getting a real-time and on-the-fly response to his original demand, under the form of a new service fully matching his request and immediately activable and usable – another interest of this solution is set in the notification mechanism of the Service Repository: every user who previously subscribed to the "Instant Delivery" service is informed that a new service is available.

The following paragraphs explain in detail the functioning of the three main blocks of the DOD Mechanism: discover, orchestrate and deliver.

The discovery mechanism: from the user's original request to the pertinent elementary web services

Before entering the Discover module, a pre-treatment is made onto the text of the original request, in order to transform it into semantic clauses, each of them representing a precise need for an elementary web service.

This need has to be expressed in a formal way, for example under the form of a XML-based language, to allow its easier post-processing.

In the context of our solution, several choices were possible for this semantic description language:

- creating a new semantic language – but, considering the number of already existing ones, we thought this way was not the best one ;
- extending an existing semantic language, like OWL for example – but, in spite of a standardization process driven by W3C, OWL is not yet systematically used everywhere inside Internet to describe the semantics of web services…
- imagining a simple solution based onto existing and already-present structures, like WSDL for example (Web Service Description Language, a W3C's standard).

We made this last choice ours, and we developed a semantic description mechanism based upon WSDL– taking into account the fact that a WSDL description is always present for any web service. The process of making semantic descriptions from WSDL is protected by one of our Alcatel patents.

A "semantic description" for a given service contains:

- input data description (data category, types and semantic tags)
- output data description (same structure than input)
- semantics of operations
- "orchestration handlers", i.e. connection points expressing how this very web service could be automatically connected with another WS. This automatic connection process is protected by another of our Alcatel patents.

The semantic descriptions for the required elementary services, built by a Semantic Analysis module from the text of the original request, are stored in a specific Semantic Repository, local to the Application Server (see Fig.2, flow 4).

These descriptions are used by the Discover block (flow 5) to make a search into the Internet/Intranet Semantic Service Repository (flow 6). The Discover block contains a matching module that explores the external Semantic Repository, looking for a match between the external semantic descriptions (flow 6) and the required semantic descriptions of the flow 5 that come from the user's request. This match is based upon a correlation search between the main elements of the semantic description: inputs, outputs and semantics of operations.

When a match is found for each required semantic description, the corresponding WSDL descriptions of the elementary services are sent to the Orchestration module.

The composition mechanism: from the elementary bricks to the final new service

Orchestration is the term used to describe the creation of a "business process" (or a workflow) using Web services. A business process (BP) is an aggregation of services whose the operations (i.e. the processes) are logically linked together in order to reach a given objective.

Aggregating services to build an added-value service have many solutions depending on the chosen environment. For Web services, the orchestration is usually expressed with a specific language like BPEL for example (Business Process Execution Language), that describes the interactions between the services.

A business process:

- has a goal,
- has specific inputs and specific outputs,
- uses resources,
- has a number of activities (an activity is a simple task, such as invoking a service, testing a value, …) that are performed in a certain order.
- is deployed itself as a service, so it can be used by other processes

A business process language describes the behaviour of business processes based on Web services, i.e.:

- Control flow (sequences, loops, conditions, parallelism, …)
- Variables, exceptions, timeout management.

By providing semantic descriptions of services, we produce in fact specifications for all the layers of the Web services stack (see Fig. 3) and so in the service composition layer.

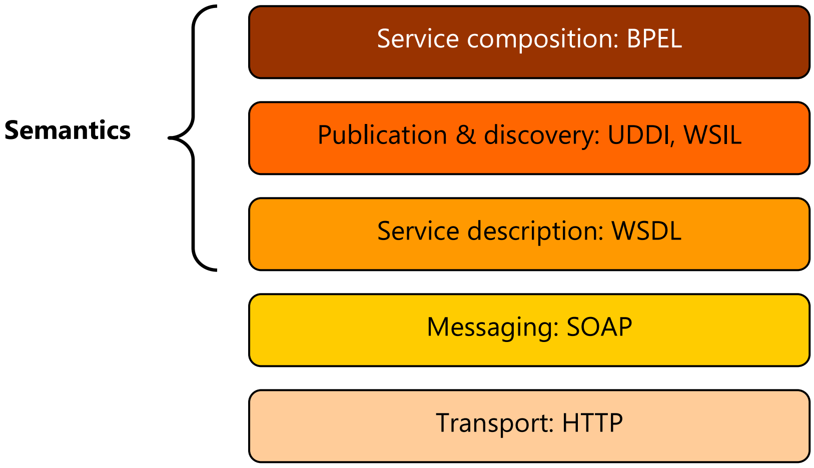
Semantics
Service composition: BPEL
Publication & discovery: UDDI, WSIL
Service description: WSDL
Messaging: SOAP
Transport: HTTP

Figure 3 – Place of Semantics in the standard Web service stack

The stack has been enriched with new W3C specifications, such OWL-S for example, for providing support of semantics. These kind of languages – our "semantic descriptions", protected by an Alcatel patent, have the same goal – provide a set of XML constructs for describing the properties and capabilities of Web services, in order to facilitates the automation of Web services processing, including automatic discovery and composition.

In the previous step ("Discover" mechanism), we have extracted the "semantics" from the service descriptions to find the matching services. In the current step, we automatically build the business process with some specific *semantic elements* (that we call "semantic tags") included in the service description in order to help the composition.

For example, the *semantic elements* give information about the input and output parameters: in a business process, output parameters of a service can be used as input parameters for another service.

In order to illustrate this point, let us take a simple example where the business process has three activities, that are service invocations. The input parameters are:

- Email content
- Email destinations

The first activity (get email language) has one input parameter, the email content, and one output parameter, the language of the email. This output parameter is an input parameter of the second activity (translate email into English). Finally, the output parameter of the translation service becomes an input parameter of the email service (see Fig.4).

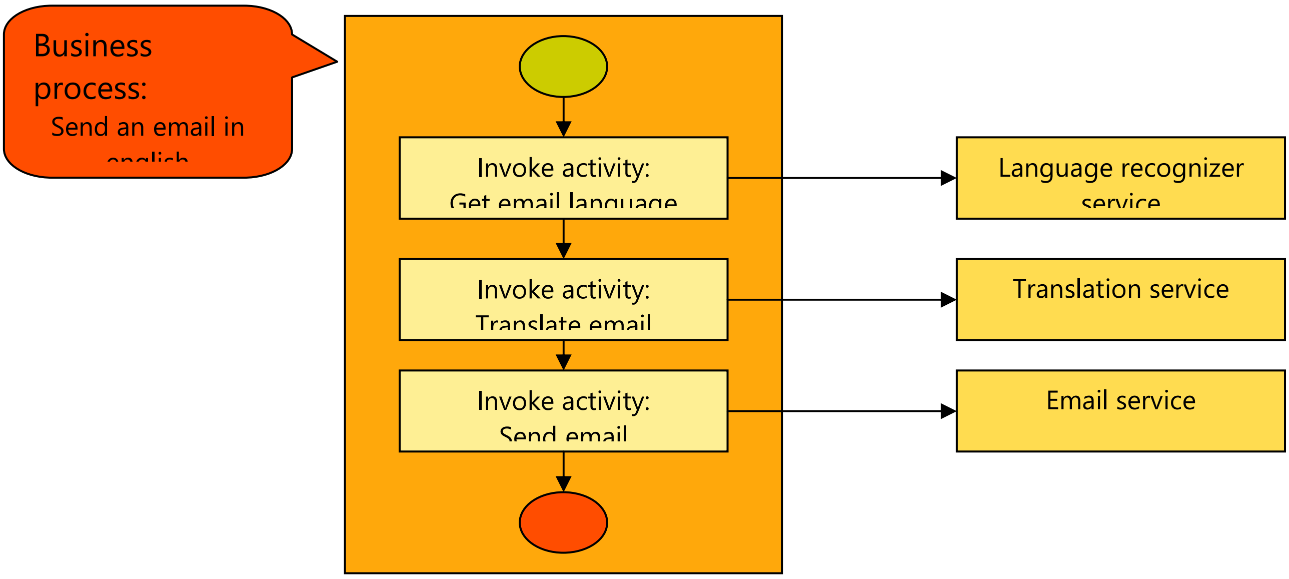
Business process:
Send an email in english
Invoke activity: Get email language
Invoke activity: Translate email
Invoke activity: Send email
Language recognizer service
Translation service
Email service

Figure 4 – An example of a simple Business Process where the output of a service is the input of another

The "semantic tags" are important to use the parameters in the right way. The parameter types are not sufficient to match output parameters of a service with input parameters of another services. The "semantic tags" help this composition.

In a business process, there is not only service invocation activity: there are also condition activities, loop activities, error processing, etc.

These activities are also taken into account: for example, if an activity has an array of string as output parameter and if the next service has only a string as input parameter, the business process includes a loop activity for processing all values of the array.

Once the business process has been defined and described with a business process language, it is time to deploy it.

The delivery mechanism: serving the newborn service to the user

With the Service-Oriented Architecture (please see the next chapter to get detailed informations on SOA), we have a good level of dynamicity on server side. But, on client side, as the client of the services is an application that is installed on each machine, it must also be as dynamic as possible to be able to invoke any services.

There are two problems:

- How to access the new service?
- How the user knows there is a new service?

To access the new service, the user must have a service client application on his terminal. But as all the services are different, the service clients are different (service parameters, user interface, etc.).

So, the service deployment step must include:

- A service client building,
- A user notification.

The service client application (see Fig.5) must include the following functionalities:

- Business Process Server notification management.
- Service client management. Its role :
    - Download, start, stop, delete the service clients
    - Process events from the *notification manager*
    - Manage the user interface according to the authorized service clients
- Service clients execution: their main role is to access the services. They have their own user interface.

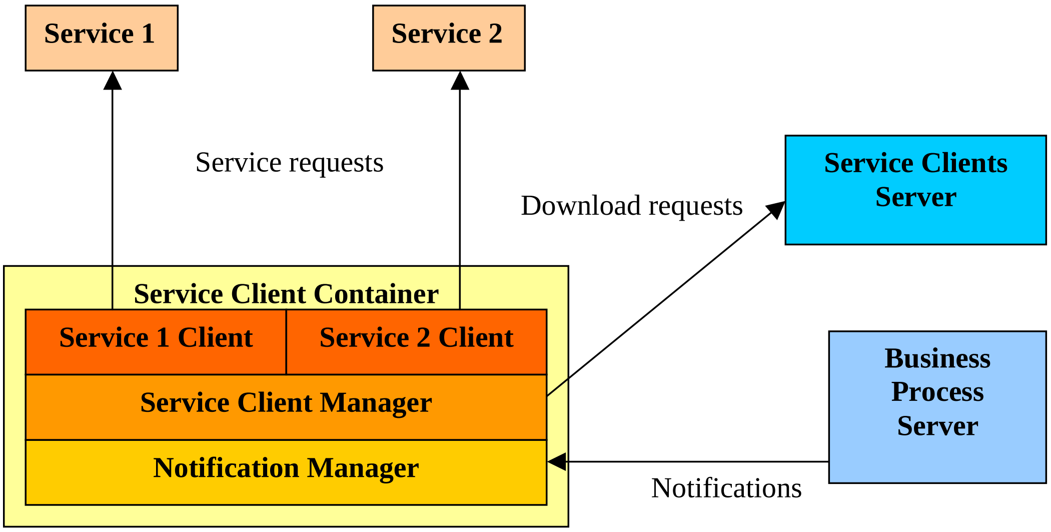

Figure 5 – Detailed functional view of a Service Client Application

A new business process deployment includes the following steps:

- Building of a new service client. It is built from the new service description. For each service argument, a user interface element is created.
- Deployment of the service client.
- Notification of the user that a new service is available and that he can access it.

When the user receives the notification, he activates the service:

- The *service client manager* downloads the service client
- The service client is activated
- The service client invokes the new service

The main advantages of this kind of service client container are:

- Install once: only the Service Client Container has to be installed on each machine, the other parts of the application (service clients) are downloaded
- Context-aware: the application checks the hardware and software environment for proposing only usable applications and for using user's preferred tools such as web browser
- Dynamic upgrade: when running, if the Service Client Container receives a notification from the Business Process Server for any modification, it is immediately taken into account; this mechanism is transparent for the user and does not need any restart.

Managing the "Discover-Orchestrate-Deliver" concept onto a Service-Oriented Architecture

Service-Oriented Architecture (SOA) is an architectural style whose goal is to achieve loose coupling among interacting software agents. In this context, a service is seen as a unit of work done by a service provider to achieve desired end results for a service consumer. Both provider and consumer are roles played by software agents on behalf of their owners.

To achieve loose coupling among interacting software agents, SOA employs two architectural constraints:

1. A small set of simple and ubiquitous interfaces to all participating software agents. Only generic semantics are encoded at these interfaces, that are universally available for all providers and consumers.

2. Descriptive messages constrained by an extensible schema delivered through the interfaces. No, or only minimal, system behavior is prescribed by messages. A schema limits the vocabulary and structure of messages – and an extensible schema allows new versions of services to be introduced without breaking existing services.

Since we have only a few generic interfaces available, we must express application-specific semantics in messages. We can send any kind of message over our interfaces, but there are three main rules to follow before we can say that an architecture is service-oriented:

1. The messages must be descriptive, rather than instructive, because the service provider is responsible for solving the problem – when you go to a restaurant, you tell the waiter your order and preferences but you don't tell the cook how to cook your dish step by step.
2. Service providers will be unable to understand your request if your messages are not written in a format, structure, and vocabulary that is understood by all parties. Limiting the vocabulary and structure of messages is a necessity for any efficient communication.
3. Extensibility is vitally important: the world is an ever-changing place and so is any environment in which a software system lives. Those changes demand corresponding changes in the software system, service consumers, providers, and the messages they exchange. If messages are not extensible, consumers and providers will be locked into one particular version of a service.

There are numerous additional constraints one can apply on SOA in order to improve its scalability, performance and reliability. Among them, we mainly retain the stateless service concept. Each message that a consumer sends to a provider must contain all necessary information for the provider to process it. This constraint makes a service provider more scalable because the provider does not have to store state information between requests. This is effectively "service in mass production" since each request can be treated as generic. It is also claimed that this constraint improves visibility because any monitoring software can inspect one single request and figure out its intention. There are no intermediate states to worry about, so recovery from partial failure is also relatively easy. This makes a service more reliable.

Taking in account these main particularities of SOA, we can consider the main challenges for our Discovery-Orchestrate-Deliver concept in a Service-Oriented Architecture are the

descriptions of messages and services, the management of services and the process for facilitating the access to them.
At "Discover" level, the semantic descriptions of services – as they are made from the external descriptions of messages found in WSDL – make more easy the classification of services in the service repository, because these descriptions are structured upon a taxonomy (see the Glossary) of the concepts manipulated by the services. This taxonomy facilitates the logical ordering of services by domains, subdomains, topics and subtopics, and consequently facilitates their management. These semantic descriptions, as they can match with the service requirements contained in the user's original request, are also a good mean to make more easy the discovery of corresponding services.
At "Orchestrate" level, the "composition handlers" contained in the semantic descriptions of services are a good enabler for an automatic orchestration process able to aggregate several convenient services.
The semantic descriptions of a new service, and mainly their taxonomic abilities, are still used at "Deliver" level, when the DOD mechanism has to store them as logically as possible and to insert them in the existing structure of the semantic repository.

Building rich services through OSDE with IMS services and elementary Web Services
Alcatel's Open Service Delivery Environment (see Fig. 6) is the preferential platform to develop and deliver rich services by aggregating elementary services. OSDE provide several kind of services, among which IMS services that are built by the IMS SDP.
The openness of OSDE allows it to integrate our DOD solution under the form of a module inside the Service Enablers building block (see this block in Fig.6). Taking into account the capabilities of the OSDE platform and leaning on them, the functioning of our solution is made easy through the following points:

- after the translation of user's request into semantic requirements, the Discover part of our DOD solution finds the elementary services to combine not only on Internet/intranet, but among the IMS services and other elementary ones provided by the OSDE platform;
- after gathering these elementary services, the Orchestrate part of DOD uses the capabilities of the Service Orchestration building block of the OSDE platform, by sending to it some specific orchestration messages generated by our "Orchestrate mechanism"; thus, this mechanism is finally lightened into a simple translator: semantic descriptions that contain composition handlers are simply translated into the specific orchestration messages that are requested by the OSDE Service Orchestrator, and the work of orchestration itself is totally transferred to this OSDE building block;
- consequently, the "Deliver mechanism" is also totally taken into account by the delivery mechanism which is internal to OSDE, all the more that the existing OSDE Service Repository contains a notification engine, implementing our requested local intelligence allowing to send to the user the notification informing him that the new asked service is available.
- however, to get all the benefit of the semantic approach embedded in our DOD solution, it is necessary to complete the existing OSDE platform by a specific Service Semantic Repository, in which the pre-Discover modules (see flow 4 in Fig. 2) will pour the

semantic descriptions of the needed services; afterwards, the re-processing of these descriptions is made possible at any time by the interactions between the OSDE-and-DOD building blocks: translation, by the Discover modules, of the semantic descriptions read in the Semantic Repository into orchestration messages that are sent to the OSDE Service Orchestrator; then, aggregation of the elementary services and delivery-notification of the new service.

Thus, new services delivered to end users are enabled thanks to the deployment of the IMS infrastructure provided by OSDE, an existing Alcatel offer.

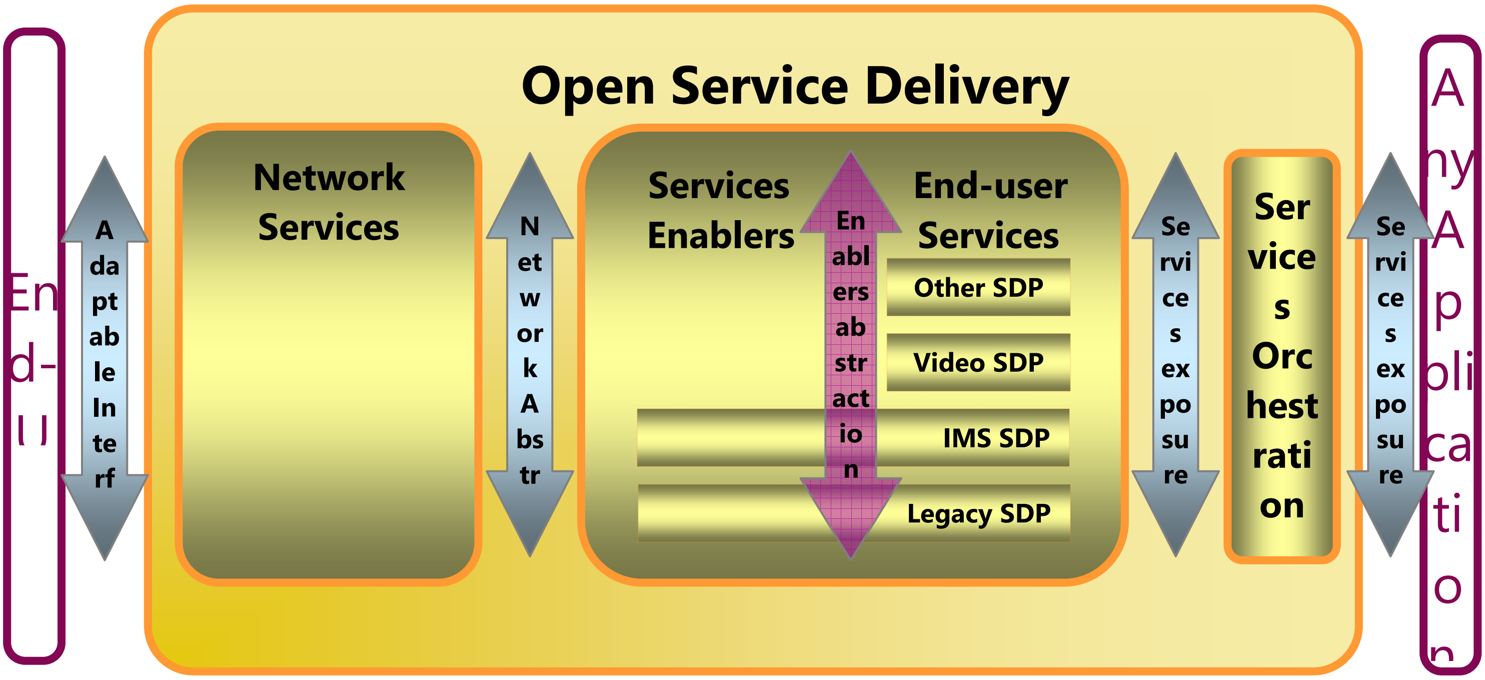

Figure 6 – Alcatel's Open Service Delivery Environment

(Note to the Editor : please take the original schema in the attached PowerPoint document, because the transfer from PowerPoint to Word has changed the look of the drawing…)

3. Conclusion (250-300 words =1 p)

Recap the main strengths of the proposed solution/technology and what value it brings to the industry/customer. In particular, refer to how it will help bring about a User-Centric Broadband world.
If relevant, give hints about the future and evolution of this technology.

Within a wide-communicating world where services become more and more sophisticated and are on the beginning to expose their semantic dimension, we consider it is important to propose solutions in which anybody could be able to create his own services in some simple clicks. This paper reveals one of the reflexion axis we currently have in this direction, and we hope we could convince the reader that a semantic approach, through the use of semantic descriptions of services, helps in an important way the mechanisms for building, holding and processing any kind of Web service.

Moreover, we have shown how SOA-based mechanisms, leant on the IMS SDP platform, can provide high-value capabilities for new generation service creation and delivery: a simpler way to express a service, a faster way to get and use it, a cheaper solution for the end-user because it is simpler and faster.

We are convinced that IMS services are the good base to develop such kind of rich services and use interaction, including dynamic search, discovery and publishing of web information associated to a multi-media call session, and we think Web services are the good means to do it.

With the Open Service Delivery Environment and the IMS SDP, Alcatel owns some powerful tools for service discovery, creation and publishing based onto Service-Oriented Architectures.

Glossary of terms and abbreviations:

Abbreviations, acronyms and technical terms must always be fully presented with the full explanation of what it means, the first time they are mentioned.
Set up a complete list of the abbreviations/acronyms in the table below.

Abbreviation/acronym	Description	Category (see list below)
BPEL	Business Process Execution Language	
DOD	Discover, Orchestrate and Deliver	
IMS	IP Multimedia Subsystem	
OSDE	Open Service Delivery Environment	
OWL	Web Ontology Language	
SDP	Service Development Platform	
SOA	Service-Oriented Architecture	
Taxonomy	(from Greek "taxis", arrangement or division, and "nomos", law) : the science of classification according	

	to a pre-determined system, with the resulting catalog used to provide a conceptual framework for discussion, analysis, or information retrieval. The development of a "good" taxonomy takes into account the importance of separating elements of a group (taxon) into subgroups (taxa) that are mutually exclusive, unambiguous, and taken together, include all possibilities.	
W3C	World-Wide Web Consortium	
WS	Web Service	
WSDL	Web Service Description Language	
XML	Extensible Mark-up Language	

Categories for key words

Access
Carrier Data
Carrier Voice/Multimedia
Enterprise
Metro
Mobile
Modems
Network Management
Operation Support Systems
Optical Components

Professional Services
Software Applications

Telephones
Transport Automation
Wireless Access and Transmission

References & contacts

Anything that can provide more information / detail about the subject:
Others external articles, papers, keynotes, patents…
Recommended books on the subject…
CAUTION : If you make references to http links, please indicate the title of the document, authors, correct address, date, and document type.

BPEL Editorial Team, *BPEL Learning Guide*, February 2005, http://searchwebservices.techtarget.com/originalContent/0,289142,sid26_gci880731,00.html

Dubray Jean-Jacques, *BPML for Web services*, June 2004, in http://www.ebpml.org/bpel4ws.htm

Bonnin Bruno, Larvet Philippe, Fontaine Patrick, Ferres Lamia, *A Multi-Actor Agnostic Platform for Web Services Development and Deployment*, June 2005, Article published in W3C Workshop on Frameworks for Semantics in Web Services, Innsbruck (Austria)

Kavantzas Nickolaos, *WS-CDL, Web Service Choreography Description Language*, December 2004, http://www.ebpml.org/ws_-_cdl.htm and http://www.w3.org/TR/ws-cdl-10/

Kavantzas Nickolaos & al., *Process-centric realization of SOA : BPEL moves into the limelight*, Web Services Journal, Nov.2004, http://www.findarticles.com/p/articles/mi_m0MLV/is_11_4/ai_n7071401

Nanda Mangala & al., *Decentralized Orchestration of Composite Web Services*, IBM Research Computer Science, Innovation Matters, November 2004, http://www.research.ibm.com/compsci/project_spotlight/distributed/

Peltz Chris, *Web services orchestration, a review of emerging technologies, tools, and standards*, Hewlett-Packard Co, January 2003, http://devresource.hp.com/drc/technical_white_papers/WSOrch/WSOrchestration.pdf

Smith Howard, *BPM and MDA, Competitors, Alternatives or Complementary*, Business Process Trends, White Paper, July 2003, http://www.bptrends.com/publicationfiles/07%2D03%20WP%20BPM%20and%20MDA%20Reply%20%2D%20Smith%2Epdf

SOA explained: http://webservices.xml.com/pub/a/ws/2003/09/30/soa.html

Details on authors:

Your photo (ID) is requested in color and in 300 dpi resolution. Try to use terms which are understandable to non-Alcatel people in the description of your main position and the department in which you are working. Keep it short and clear.

Lead author

Name	LARVET
First name	Philippe
Main position	Technical expert
Functional entity	CTO – Research & Innovation, Marcoussis (France)
Picture	

Biography	P.L. is a Technical Expert engineer at the Alcatel R and I Center of Marcoussis since beginning of 2004. His research topics include semantic definitions of web services and natural language processing. From 1998 to 2003, he had a mission of innovation in SRD-VND at Velizy, where he worked on introducing object technologies on E10 projects, and on Knowledge Management by developing tools and procedures to capture know-how and to enrich lexical dictionaries included in SRD-VND documentation system. Before joining Alcatel, he worked since 1991 in several industrial companies as a software engineering expert in the scope of real-time and embedded systems, upon numerous projects in object technologies at the level of specifications and design. He is the author of several publications, among them a book published in 1994 at InterEditions, Paris (*Analyse des systèmes : de l'approche fonctionnelle à l'approche objet)* Philippe.Larvet@alcatel.fr

Contributor

Name	BONNIN
First name	Bruno
Main position	Research engineer
Functional entity	CTO – Research & Innovation, Marcoussis (France)
Picture	
Biography	Bruno Bonnin is graduated in software engineering (MSc) from the

	University of Nantes, France. Today, he is a research engineer in the Mobility and Business Applications (MoBA) project at the Research & Innovation Center, Marcoussis (France). His research topics include web services orchestration mechanisms, Service-Oriented Architectures and design of graphical intelligent environments facilitating service composition. Bruno.Bonnin@alcatel.fr

Related information

Top 5 Keywords

Extract from your article the 5 most relevant keywords and classify them according to their relevance to the subject covered. Then, refer to Alcatel Terminology Database to check if a definition exists for each of your 5 keywords. If it is not the case, please provide a definition.

Writing tips for the definition:
A definition does not exceed more than two sentences.
Please avoid typical writing mistakes such as "A proxy server is a server..." or "A service provider provides...".

Rank (1=most relevant)	Keyword	Category (see table below)	Proposed definition
1	Web service	Prof.services, Softw.applic.	A stateless, contextless piece of software easily accessible on Internet/Intranet through a specific Application server.
2	semantic	Softw.applic.	Concerns the meaning, the signification. The semantics is the study of language and linguistic signs (words, expressions, sentences) from the point of view of the meaning (greek "semantikos", "that means").
3	orchestration	Prof.services Softw.applic.	A way to compose services by insisting on the assembly of processes (choreography is centered on the exchanges of messages and events).
4	discovery	Prof.services Softw.applic.	A mechanism that helps to find, in a specific service repository, the right service(s) responding to a given requirement.
5	delivery	Prof.services Softw.applic.	A mechanism that puts a given service at the disposal of its registered clients.

Categories for key words

Access
Carrier Data
Carrier Voice/Multimedia
Enterprise
Metro
Mobile
Modems
Network Management
Operation Support Systems
Optical Components
Optical Fiber and Cable
Optical Transmission
Professional Services
Software Applications
Space Systems
Submarine Systems
Telephones
Transport Automation
Wireless Access and Transmission

Tagging your article

See the following table, complete it and please return it to us by fax or by e-mail. For each question, several answers are possible but please could you **prioritize** them?

1- According to you, what are the main technologies addressed by the article/paper?

This is a multi-valued attribute. Please put a level of relevance if several apply (1= most relevant).

	ATM		MPLS
	CDMA		NGN Voice
	CSTA / CTI		NM / TMN
	DECT / PWT		PSTN Voice
	DSL		QSIG
	DWDM		Satellite
	Frame Relay		SDH
	FTTH / PON		SONET
	Gigabit Ethernet / 10 GigE	X	Service-Oriented Architecture
	GPRS / EDGE	X	SOAP-HTTP
	GSM		SS7 / C7
X	IMS		UMTS / 3G
	IN		VLAN
X	IP		VoIP
	ISDN	X	Web services
	LMDS / Wireless IP	X	WSDL
	Microwave	X	XML

2- Who could be the targeted customer of your paper?

This is a multi-valued attribute. Please check all that apply.

	Carriers' carriers, IXCs		OEMs
	CLECs		Public Utilities
X	Consumers	X	Resellers/Distributors
	Government	X	Small/Medium Enterprise
	ILECs		Space Agencies
	ISPs		Systems Integrators
X	Medium/Large Enterprise		Transportation

X	Mobile Service Providers		TV Broadcasters

3- What are the main category in which the subject of the article/paper falls ?

This is a multi-valued attribute. Please check all that apply.

	Access		Optical Fiber and Cable
	Carrier Data		Optical Transmission
	Carrier Voice/Multimedia	X	Professional Services
X	Enterprise	X	Software Applications
	Metro		Space Systems
X	Mobile		Submarine Systems
	Modems	X	Telephones
	Network Management		Transport Automation
	Operation Support Systems		Wireless Access and Transmission
	Optical Components		

4- According to you, what is the readership for your paper ?

One choice only

	Non Technical
	Moderately Technical
X	Technical

Check List

Before sending your article to the local editor or editorial team, please remember to check the contents using the list below. The enclosed article includes:

- Name, unit and town of each author
- Your Internet address
- Your biography
- Abstract
- Main title
- Introductory punchy sentence to arouse the curiosity of the potential reader.
- Introduction
- Body of the article, with headings & sub-headings
- Conclusion
- Glossary and abbreviations
- All figures, tables and photographs
- Graphical elements references and captions for all your illustrations (figures, tables and photographs)
- References & Contacts
- Contact and More information
- 5 key words
- Tags for articles

If your file does not contain all the elements above mentioned,
IT WILL BE RETURNED TO YOU, before being read.