New Generation Mediation Solutions

A ScalAgent Distributed Technologies White Paper

Scalable and Flexible Mediation Infrastructure for Networked Smart Devices

Authors: Roland Balter and Luc Bellissard

Email: contact@scalagent.com

March 31, 2003

Executive Summary

The use of Internet as a general purpose communication system is growing very fast in all sectors of activity. Multiple forms of equipments are now connected to Internet and have gained increased embedded intelligence. This evolution leads to new interactive services that are embedded on these smart devices while interoperating with existing enterprise information systems. In this context the objective of mediation infrastructures are twofold: i) reduce operation costs by automated administration of smart devices and hosted services; ii) create new sources of revenue from valuable operational and business information retrieved from smart devices and hosted services. This paper describes the design of a flexible and scalable mediation infrastructure for networked smart devices that meets these requirements.

The Challenges of Operating Networked Smart Devices

The use of Internet as a general purpose communication system is growing very fast in all sectors of activity (banking, manufacturing, energy, healthcare, building/home automation, etc.). Multiple forms of devices and appliances are now connected to Internet and have gained increased embedded intelligence. This evolution leads to emerging interactive services that are embedded on smart objects while interoperating with existing enterprise information systems. Operation support for equipments and hosted services is a major issue for the various actors involved: end-users (i.e. devices owner), device manufacturers, service providers. The term "operation" refers here to the set of management functions involved in the device/service life-cycle: deployment, configuration, monitoring, maintenance and reconfiguration as necessary. The purpose of an operation support system (OSS for short) is twofold. On the one hand automated administration allows a reduction of operation costs associated with the management of a large number of heterogeneous smart devices. On the other hand, intelligent devices contain valuable operational and business information that can be retrieved in real-time to feed business applications such as remote supervision, billing, service promotion and customer relationship management, etc.

From the technical point of view, these objectives are achieved through the definition of an efficient and flexible operation support system that provides two types of services:

- *Service Activation* to install and configure software components on remote devices. Components implement both end-user services and system support functions.
- *Data Mediation* to collect and process usage data from remote devices (and hosted services) and to deliver them to business applications within the enterprise information system (e.g. monitoring, billing, CRM, etc.).

This paper focuses on the design of a **distributed mediation infrastructure** that meets the needs of operation support systems for networked smart devices. Mediation infrastructures are in the heart of IT systems as they provide an integration facility between a set of heterogeneous remote autonomous devices and business applications. They enable the seamless integration of usage data from devices with on-going business activities in real-time, thus leading to faster reaction times, proactive customer services and improved operational efficiency.

Mediation and its Requirements

The term "mediation" was born in the telecom area, where the mediation building-block is now recognized as a strategic element at the forefront of numerous business applications such as rating and billing, supervision and QoS management. Emerging applications involving smart devices - such as health care systems, building/home automation, energy systems, etc. - are today facing similar problems, but the solutions adopted so far in the telecom area are not totally suited for them because of specific issues involved in the management of networked smart objects.
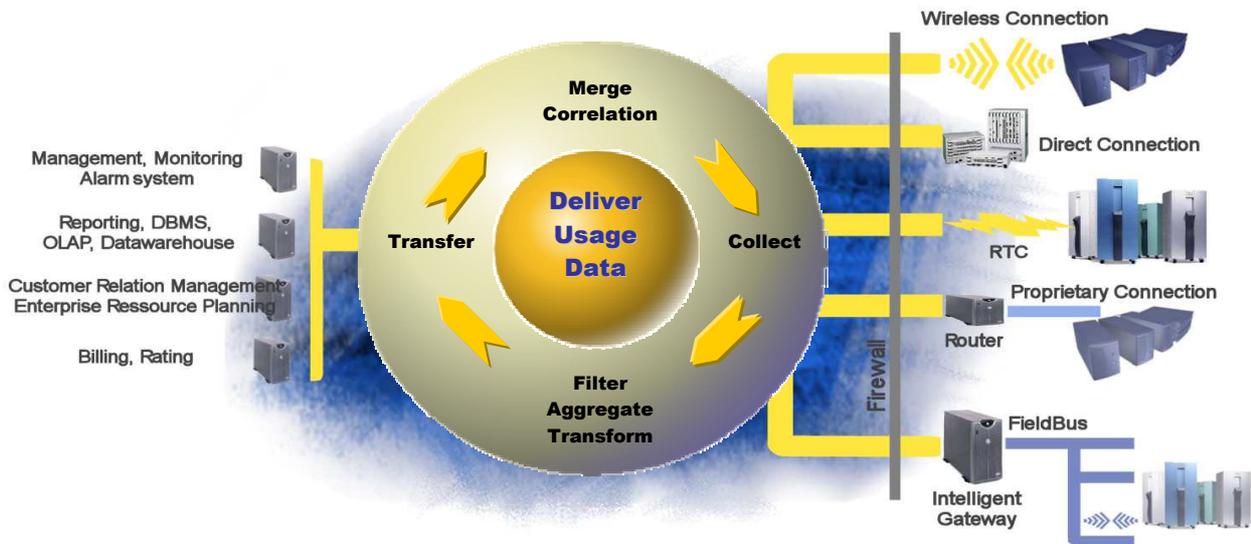
> ### What is Mediation ?
> Data Mediation is the process of collecting and processing usage data from networked devices and to deliver them to business applications in real time
> ### What are Mediation Systems ?
> Mediation Systems are the business software infrastructure integrating widely distributed devices and information systems through the Internet or large-scale private network.

Mediation systems for networked smart devices address the following requirements :

- *Scalability* due to the very large number of heterogeneous devices involved. In addition, devices are joining and leaving the system frequently. Moreover, a large number of devices usually imply large volumes of usage data to be collected and processed. This leads to technical issues such as limited network bandwidth and computing power available to handle usage data.
- *Flexibility* to adapt the mediation infrastructure to the characteristics of the physical environment and to the user needs. The infrastructure should also be able to evolve rapidly to address new requirements (e.g. new device or new service such as a new billing policy) and changing operational conditions (e.g. networking topology). The level of flexibility can be measured by the time - and the development cost - needed to adapt the existing infrastructure to evolving parameters. This requires dynamic reconfiguration capabilities.
- *Cost-effectiveness* of the overall mediation solution despite the overhead due to additional software functions embedded in "cheap" smart devices. Today this productivity challenge is achievable thanks to recent technological evolutions in communication and computing capabilities in the field of smart devices.

Mediation solutions currently available on the market have severe limitations that do not meet the criteria mentioned above. They usually take the form of a monolithic packaged product that combines data acquisition and processing as well as part of the business application code. This "integrated" approach lacks flexibility in the context of rapidly evolving distributed heterogeneous environments. Moreover these solutions follow a centralized client-server architecture. Data are retrieved from remote devices using basic protocols (e.g. FTP, HTTP, SNMP, etc.) and the whole data processing is concentrated on the application server. This approach does not meet the scalability requirement and also suffers from a lack of reliability. Finally, operating and maintaining these solutions in a large-scale perspective is still difficult and thus expensive.

Design Choices for a Mediation System

Architecture - Distributed Intelligence for the Right Function at the Right Place

It is recognized today that centralized architectures introduce bottlenecks both at the communication and computing levels. They also suffer from a lack of availability as they are subject to single-point failures. Replicated architectures (i.e. cluster systems or replicated data servers) bring a partial answer to this problem but these solutions are complex and expensive.

At the opposite, bringing the processing functions close to the data sources provides two major advantages: i) raw data are processed locally, only pertinent information is reported to the management node thus saving network bandwidth. ii) the overall computing load can be balanced between the various nodes in a configurable way (smart device itself, intermediate consolidation node, central management node). Implementing this *distributed intelligence* approach can be achieved as follows:

- Use *component-oriented programming* to structure the management intelligence in fine-grained computing units and deploy them on the right node based on application requirements. Software components and architecture definition languages (ADL) are used to achieve this goal.
- Rely on *Java programming* language to be independent of a given host environment. Java holds great promises as the de facto standard for information appliances.

Middleware Services - Asynchronous Communication for Reliability and Scalability

Asynchronous communication systems (so-called "message bus") have numerous advantages compared to RPC client-server approaches (like SOAP/XML for example) [1]. Asynchronous communication is better suited to deal with disconnected mode encountered in ubiquitous environments. Reliable communication is achieved by persistent message queuing facilities. In addition, some message buses provide also checkpoint and recovery mechanisms that help designers to build full reliable applications. Scalability is achieved through a distributed implementation of the message bus (i.e. "snowflake" architecture). This approach leads to configurable topologies that can be tailored to the physical constraints (network and nodes capabilities) of a given environment. Additional features are needed to completely meet the mediation requirements:

- *Availability of lightweight versions* of the message bus to be embedded in devices with limited resources (memory foot-print, computing power, etc.).
- *Compliance with widespread e-business standards* so that information transported by the communication system could be easily delivered to business applications (JMS, HTTP, XML, Corba, etc.).

The choice of a componentized architecture has been argued above. Following this approach a mediation solution for networked devices can be viewed as a collection of distributed interacting software components. Design and management tools are required to maintain the consistency of the overall application despite the changes involved in the application lifetime, as they are described later. They rely on a set of middleware services for the deployment, monitoring and control of individual software components. The complete description of the middleware and all componentized application management services is available in [2].

Programming Model - Distributed Components and Event-driven Execution Model

Most of asynchronous communication systems available today provide a proprietary application programming interface (API) that takes the form of a function library available through various language mappings. The *JMS* specification (*Java Messaging Service*) is a first step forward as it provides a Java API independent of a given bus implementation. However, the use of APIs is still cumbersome and error prone. We advocate that better support should be given to application programmers. This can take the form of a programming model adapted to the description of mediation functions. From the functional point of view the "*Actor*" paradigm [3] is a good basis for such a model as it was designed for an asynchronous world. Component services are triggered on the reception of events sent by other components using either point-to-point or publish-subscribe interactions. The model should also provide features for the control of non-functional properties such as atomicity, persistency, security, reconfiguration, etc. This can be achieved through the component approach that clearly separates the functional part of the component (described by its interface) and the non-functional part, described in a so-called "*container*".

In addition, the component structure can be used to encapsulate external software packages such as operation code on smart devices and legacy applications on the management node. Interoperability with this special type of components is achieved through specific gateway components (called "*connectors*"). Therefore, the component paradigm provides a uniform view of the overall distributed application, which is exploited by a set of tools as explained below.

It should be noted that distributed components embedded in smart devices can be considered as *intelligent agents* integrated in a global solution.

Development and Administration Tools - ADL (Architecture Description Languages)

The development of distributed applications is still difficult and time-consuming. Therefore we think that the component-based approach combined with appropriate development and administration tools should greatly help users in the design, deployment and management of distributed applications. Experience drawn from many years of research in this area has convinced us to use an **ADL** (*Architecture Description Languages*) to describe an application as an assembling of interacting software components. ADL allow components, their interfaces and properties to be specified, as well as links between them [4]. This overall description is used during the whole application life-cycle to provide a complete and consistent view of what the distributed application is. It is thus possible to exploit this formal representation to pilot the deployment process, to monitor the application components and later-on to reconfigure part of the application in order to adapt it to evolving user requirements and to changing run-time conditions. This configuration-adaptation capability is a major advantage for developers as applications can be tailored according to their needs or those of their own customers.

It should be noted that ADL-based development tools are complementary to (and can cooperate with) modeling tools, especially those based on the UML formalism, since ADLs are close to UML component and deployment diagrams.

The ScalAgent Mediation Solution

The *ScalAgent* Mediation infrastructure has been designed to provide an actual implementation following the design guidelines described in the previous section. The architecture of this infrastructure is depicted in the figure below.

Middleware

The *ScalAgent* MOM (Message-Oriented Middleware) provides an asynchronous messaging service that guarantees messages delivery and causal ordering of messages. The MOM ensures reliable communication paths between components in Internet-based large scale wide-area networks.

The *ScalAgent* MOM is 100% Java, so that it can run on a wide spectrum of stations, servers and Internet appliances. Therefore various classes of distributed applications are supported including those that require communications between smart objects and application and data servers. From the functional and performance points of view the *ScalAgent* MOM can be favorably compared to other Java-based message buses available on the market.

This MOM can be accessed in two ways using complementary application programmatic interfaces. One is compliant to the JMS™ standard (*Java Messaging Service*). A more sophisticated API, based on the component paradigm, is also provided to application designers. The *ScalAgent* MOM and its JMS API are available today as an open-source package (called **JORAM**) on the ObjectWeb platform [5]. The component-based API is accessible through the *ScalAgent Mediation Suite Software Development Kit*.

Component Model

The *ScalAgent* component-based programming model has been designed to be integrated with a MOM and to be bound with ADL tools. *ScalAgent* components are distributed Java objects that communicate by sending and receiving messages. A message is a Java class that extends a pre-defined class of the run-time system. A component is made of two parts: a functional part (or *SCBean*) and a container part (or

SCContainer). A hierarchical construction of components is achieved through the use of the Olan ADL [6].

The functional interface of a *SCBean* distinguishes provided interfaces from required interfaces. A required interface of a component can be connected to a provided interface of another component if the former conforms to the latter. Connections between provided interfaces and required interfaces are described by ADL statements. The *SCBean* execution model extends the traditional event/reaction model with additional (optional) features such as persistency and atomic execution.

A *SCContainer* encapsulates a *SCBean*. It manages configurable non-functional properties (e.g. persistency, atomicity, security, deployment, monitoring) and implements the mapping with the underlying middleware services.

Mediation Framework

Mediation components implement the mediation processing logic. The framework provides a set of generic operations that can be configured and specialized as necessary. Basic operations include functions such as: collection of usage data generated by various types of smart devices, parsing, filtering, aggregation, as well as correlation and merging of several data sources, and finally the transformation to different formats.

In a first step mediation chains are built from the assembling of mediation components (using ADL tools). At this stage, this functional description of the mediation solution is independent of a given network configuration. In a further step components are deployed on their execution nodes. The deployment process is automated through the use of a specific tool (see below) based on the description of the mediation chain on one hand and a set of physical run-time conditions on the other hand.

Mediation servers are in charge of hosting and operating mediation components, ensuring secure communications and guaranteed delivery of usage data despite transient network failures and disconnected mode. Servers can operate on any Java enabled host environment. Depending on the type of hardware, two types of mediation servers can be used.

- *Mediation Servers* are the full version of the mediation runtime support. They are configured to ensure part or all of the following properties: persistency of smart agents, guaranteed delivery of mediation data, global ordering of messages, secure communication channels, web-based management, JMX™ enabled management.
- *Mediation eServers* are a lightweight version of the mediation runtime support that provide only a subset of server properties according to hardware capabilities and application needs. Mediation eServers have been developed for various types of devices such as: industrial/home gateways (e.g. an OSGI gateway), industrial automatons, Java smart cards, PDAs, SmartPhones, etc.

In addition, specific "*business connectors*" components provide gateways to Enterprise Application Integration middleware standards, such as: JMS™, EJB™ (RMI/IIOP), CORBA (IIOP), etc.

Development and administration tools

ADL-based tools assist the application programmer in the design and customization of the mediation solution. They also provide support for the remote control of the installation, configuration and monitoring of software components whatever the complexity of the distributed solution could be. These tools are used through a friendly graphical user interface. The tools currently available include:

- *Component Description Tool*, to specify the application architecture: basic components as well as the composition and cooperation rules between components.
- *Component Assembling Tool* (or *Configuration Tool*), to define and customize a particular instance of an application: instantiation and configuration of a set of appropriate application components.
- *Application Deployment tool, t*o install the actual components into their target environment and to set up the links between them. At the end of the deployment stage, the application is ready for execution.
- *Monitoring Tool*, to control the distributed execution of components.
- *Reconfiguration Tool*, to adapt the overall application structure to evolving business environments. Changes can occur on components (addition, removing, replacement) as well as

on links between components. The ADL-based description of the entire distributed application is stored in a repository. This description can be modified in two ways: statically using the ADL tools or dynamically using an API. The latter allows new service components to be added as a result of a discovery process.

Example - the ScalAgent Mediation Infrastructure in use

The following example illustrates the use of the *ScalAgent* mediation infrastructure for the remote supervision of uninterrupted power supply (UPS) devices. UPS devices have been equipped with a communication board that embeds an IP stack and a JVM. The supervision logic is implemented on a J2EE™ application server, composed of JOnAS [7], an EJB application server and Tomcat, a servlet server. The overall architecture is depicted in the figure below.

A Mediation eServer is running on each UPS device. On the supervision node a full mediation server is running and interoperates with a J2EE™ application server via a JMS connector.

The mediation application is described as a set of interacting components (represented in the picture as grey round boxes). The deployment process (dotted arrows), piloted from a central application management node, allows the mediation components to be installed on their execution node.

On the UPS side, components collect usage data (e.g. energy level, QoS parameters, alarms, etc.) in real-time, and compute pertinent indicators that are transmitted to the supervision node. Indicators coming from all the UPS devices are processed and consolidated on the supervision node to provide reporting data on one hand and business data on the other hand.

Modifications on the application structure (e.g. evolution of the physical configuration or adding components to compute new indicators required by the operator) are reported in the global application description maintained on the application management node. Then the automated deployment process is performed to allow parts of the distributed application to be reconfigured as necessary.

Additional information about this use-case as well as other showcases can be found in [8].
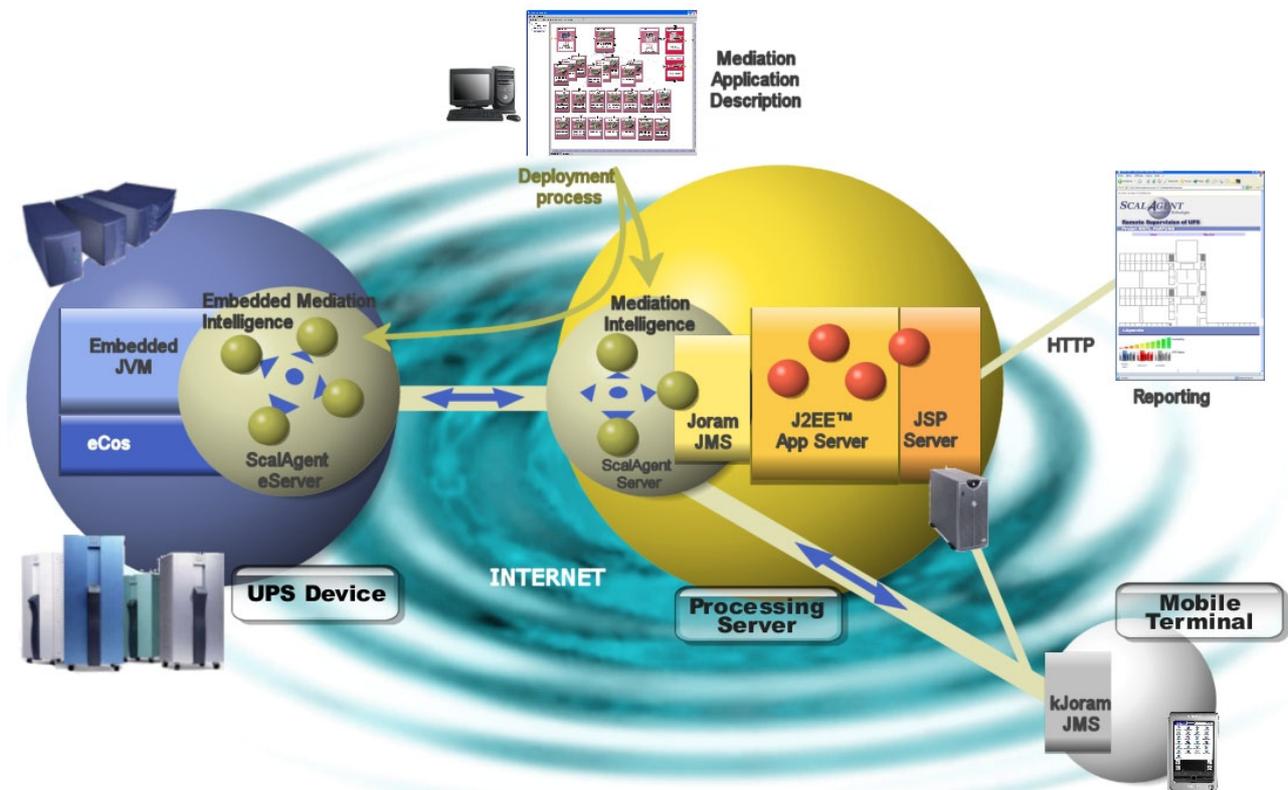
Leverage your Networked Infrastructure with Advanced Mediation Solutions

The use of networked smart devices is growing very fast, thus leading to large-scale distributed application scenarios that require accurate operation support systems (OSS) for the various actors involved: device manufacturers, service providers, end-users. The overall objective is to reduce operation costs on one hand and to generate new sources of revenue from smart devices on the other hand.

Mediation infrastructures play a major role in achieving this goal as they provide the integration infrastructure between a set of remote intelligent devices and business-oriented applications such as supervision and QoS management, billing, customer relationship management, etc. Mediation solutions built for the telecom world so far are not totally appropriate because they are still lacking features for specific requirements encountered in the management of large-scale networked devices. On the one hand, applications are evolving very fast, as new types of device are emerging. Moreover, network configurations and user requirements are subject to frequent changes. Flexibility is thus a key issue in the design of a mediation solution. On the other hand scalability is another major feature as the number of smart devices operated can be very large. Flexibility and scalability are not well addressed by today infrastructures, and new technologies are required.

The *ScalAgent Mediation Suite* is an outstanding software infrastructure that combines the advantages of numerous advanced technologies, based on extensive research work in large-scale distributed systems. This infrastructure has been designed to implement a *flexible, reliable and scalable OSS tailored for networked devices*. This infrastructure allows users to focus on their core business and benefit from the following key advantages:

- *Reduced Time to Market* for the definition and deployment of new added-value services.
- *Reduced operation costs* through automated procedures for the deployment and evolution of the mediation solution.
- *Increased performance and scalability* through distributed intelligence and highly reliable distributed architecture.
- *Investment protection* through non-intrusive and open mediation solutions.
- *Conformance to de facto standards* such as Java™, XML, HTTP, SSL J2EE™, . .

Development is ongoing to consolidate the set of "connectors" to existing field bus protocols and gateways (e.g. OSGI Gateway) on one hand, and to business applications on the other hand.

To conclude, it should be noted that mediation infrastructures share a number of properties with distributed infrastructures involved in application integration and data integration. Therefore, another direction of work is to study how the design choices and technologies described in this paper can be reused and adapted to address the construction of a scalable and flexible EAI (*Enterprise Application Integration*) infrastructure.

References

1. Banawar, G., Chandra, T., Strom, R., and Sturman, D., *A Case for Message Oriented middleware,* In Lecture Notes in Computer Science, volume 1693, Distributed Computing 13th. International Symposium, September 1999.
2. Bellissard, L., de Palma, N., Freyssinet, A., Herrmann, M., and Lacourte, S., "An Agent Platform for reliable Asynchronous Distributed Programming", *proc. of Symposium on Reliable Distributed Systems (SRDS'99),* Lausanne, Switzerland, October 1999.
3. Agha, A., "Actors: *A Model of Concurrent Computation in Distributed Systems*", In the MIT Press, ISBN 0-262-01092-5, Cambridge, MA, 1986.
4. Issarny, V., Saridakis, T., and Zarras, A., ``*A Survey of Architecture Description Languages''*, C3DS Deliverable A3.1, EspriT LTR Project N24962, 1998
5. JORAM: Java Open Reliable Asynchronous Messaging, Objectweb, 2002, **www.objectweb.org/joram/**.
6. Balter, R., Bellissard, L., Boyer, F., Riveill, M., and Vion-Dury, J.Y., "Architecturing and Configuring Distributed Applications with Olan", Proc. Of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing (Middleware'98), The Lake District, UK, September 1998.
7. JOnAS: Java Open Application Server, ObjectWeb, 2003, **www.objectweb.org/jonas**
8. Mediation solution Showcases : **www.scalagent.com/pages/en/solutions/showcase.htm**

About ScalAgent Distributed Technologies

ScalAgent Distributed Technologies is an emerging start-up company, spin-off from Bull and INRIA, which aims at developing advanced mediation solutions for large-scale Internet-based distributed applications in the Internet/Java world.

For many years, our technical and business professionals have gained considerable experience in the design and development of advanced prototypes in distributed object-oriented systems for the telecom and industry domains. This expertise and the technology building blocks developed so far are exploited to help our customers build the mediation solutions that meet their own requirements.

For his customers, ScalAgent Distributed Technologies develops mediation solutions tailored for their business requirements and for the constraints of large-scale networked infrastructures. These mediation services are designed to be easily integrated within global management solutions built by integrators for device manufacturers and for service-application-content providers.

To learn more, please visit us at **www.scalagent.com** *.*