Appli-Bus technology

The **Appli-Bus technology** is relevant to application integration in business-critical, high-performance contexts. Appli-Bus core technology can be defined as an efficient and secure object-oriented middleware. Based on the principles of a software bus (pluggable entities, event driven, distributed), it provides :

- Seamless integration in a heterogeneous distributed environment. The access to objects is uniform: one does not need to know their physical location nor the mechanisms required to reach them (networks, hosts, industrial protocols).
- Fast development of fault tolerant applications in a distributed environment. A ready to use duplication service is provided to allow objects to be managed in a stateful redundant application context.

Appli-Bus key ideas

High throughput in networked environment:
The bandwidth within any type of computing node is and will remain incomparably higher than the one of the network. Any design decision that can save network bandwidth consumption, at the expense of CPU load, is candidate for implementation within the Appli-Bus architecture.

Dynamic architecture: Platform extensibility means functional enhancements and deployment over more computing nodes over the lifetime of an application. In architectural terms, this goal is achieved in Appli-Bus by the adoption of a component model, yet minimizing the number of APIs required to interconnect the various components required in an application. The Appli-Bus technology is entirely available through 3 APIs : one API for client applications, one API for server components and one internal API. In fact, the first two APIs are user-friendly wrappers to the latter one. The design of a single, versatile API is a key feature to achieve the goal of an extensible, dynamic application platform that also offers a high level of stability in time.

Industrial Quality of Service: A set of services specific to industrial applications must be present in any framework aimed to be used as a core Technology. It includes secured event-driven communication and system high-availability.
Event driven model (some entities produce some events, some others are consuming them) is much more efficient than the polling mechanism that is generally found in industrial systems. In event driven systems, only data of interest is exchanged.
High available systems are expecting services that allow total availability (99.99%) of their services. Replication of software and hardware is generally implemented. Appli-Bus high-availability service replicates compute nodes and processes and allows these replicas to communicate over an unlimited number of physical connections (networks). These features provide high-availability and extremely limited downtime and data loss under system failure, using common hardware platforms. The use of common hardware platforms allows notably enhancing the long-term perenniality of systems and reducing the overall cost of ownership.

Appli-Bus application targets

The Appli-Bus technology is fit to applications in the supervision and control of critical systems. This technology has been successfully used in :

- Control command in the transportation sector (airport, subway, …).
- Control command in video surveillance applications.

…
This set of application domains is not restrictive. We can easily imagine Appli-Bus technology being used within the domain of Telecom, Information Technology or any domain needing efficiency and safety.

Appli-Bus main design

Architects and developers familiar with CORBA concepts will easily understand Appli-Bus design.

As with any CORBA compliant object broker, Appli-Bus allows client applications to access distant objects services in a transparent way.
Providing a simple and transparent access to distant objects is not enough when dealing with the tough functional requirements of real-world business or safety critical applications. The Appli-Bus technology goes one step further and provides many desirable features and capabilities to help address such applications.

The reader should be warned that Appli-Bus technology does not rely on the IDL language to describe Appli-Bus objects and is not CORBA compliant. The Appli-Bus technology provides a set of non-CORBA goodies that includes:

- Dynamic proxy (Stub and Skeleton). No need of static IDL compilation.
- A component framework.
- A systematic objects configuration process.
- Real asynchronous calls.
- Redundant and safe communication.
- Duplication service.
- Optimized nameserver : structured to resolve object names efficiently (pattern resolution).
- Generic gateways in order to interoperate with standard technologies (Java, SNMP, CORBA,…).

➔ This set of unique features means a stiffer learning curve compared to standard CORBA Orb implementations but it also leads to more capabilities and faster development in the long term.

Appli-Bus components

The Appli-Bus technology is designed around a **kernel** integrating the following services :

- object broker (CORBA like)
- naming
- configuration
- call grouping (call efficiency)
- notification
- duplication
- safety

2 APIs are provided in order to make use of these services:

- A client dedicated API used to make calls on distributed objects.
- A server dedicated API used to implement objects.

Some modules are provided in order to make the **integration** quicker:

- A Java gateway, in order to allow Java applications to access Appli-Bus objects.
- An SNMP gateway, in order to allow SNMP manager to manage Appli-Bus objects or applications.
- A CORBA gateway, in order to allow CORBA application and Appli-Bus applications to interoperate.
- An HTTP gateway, in order to allow WEB clients to access Appli-Bus objects.

A set of tools are provided in order to develop, test and integrate.

- A customized makefile environment to build and link easily new developments
- A scripting tool to test quickly in a simple way new developed objects