JOnAS 5

White Paper - *V1.2*

JOnAS 5, the new generation dynamic application platform from OW2, provides many innovative features to Java EE^{TM} platform administrators, operators and developers. Its brand new OSGi based architecture, combined with the JASMINe tool, enables complex but non intrusive administration operations on production environments, its powerful and dynamic clustering solutions adapt to any kind of usage, and its EJB3 container EasyBeans makes application development and deployment still more easy.

JOnAS 5 brings many innovations combining research from the members of its development community across Europe (Bull, France Telecom, INRIA, LIFL, UJF, UPM...), Brazil (UNIFOR...), and China (CVICSE, Peking University...). The objective is prepare for the dynamic SOAs of the future, while solving today's enterprise needs for mission-critical scalability and availability. As a result, JOnAS 5 offers:

1. Breakthrough flexibility and modularity through a unique **OSGi^{TM} based architecture**
2. Easy application development and deployment, through smart Java EE 5 support. The key component of this aspect is the lightweight and performant **EJB3 container** EasyBeans
3. The already powerful dedicated administration console (jonasAdmin) is complemented by the JASMINe tool, aiming at distributed applicative architecture design, deployment and monitoring, first targeting JOnAS clusters. In a near future, combined with the new flexible JOnAS architecture, this tool will provide self management features.
4. The existing **clustering** mechanisms have been redesigned and completed, to provide top level and highly dynamic solutions to scalability and high availability problems

Already deployed in production for thousands of enterprises and government applications worldwide, JOnAS is a recognized enterprise class application server, at the heart of the OW2 middleware solutions. This means

- JOnAS 4 is J2EE^{TM} 1.4 certified, and JOnAS 5 is in the process of Java EE 5 certification
- JOnAS provides scalability and high availibility through clustering solutions and many internal optimisation mechanisms
- JOnAS is ready for enterprise integration, providing many connectors, web services and IIOP support, facilities to integrate with enterprise directories and security solutions
- JOnAS provides Integrated Development Environment support, mainly through its Eclipse WTP integration
- JOnAS Administration is enabled through a high level web console, allowing multi-instances and cluster management, based on a standard JMX infrastructure. All management operations are also available through scripts and APIs (EJB, Web Services, or JMX Mbeans)
- A professional support is provided by Bull SAS, OW2 strategic member

This document describes the four main aspects of the new JOnAS application server, i.e. the OSGI^{TM} based architecture, the EasyBeans EJB3 container, the JASMINe tool, and the clustering mechanisms.

1. JOnAS Services-oriented Architecture, OSGi^{TM} based

The main benefits of the new JOnAS 5 architecture are **flexibility**, **dynamic adaptability**, and **modularity** of the application server. This will reduce administration and maintenance costs, by making administration operations easier and sometime automatic, and by limiting services interruption.

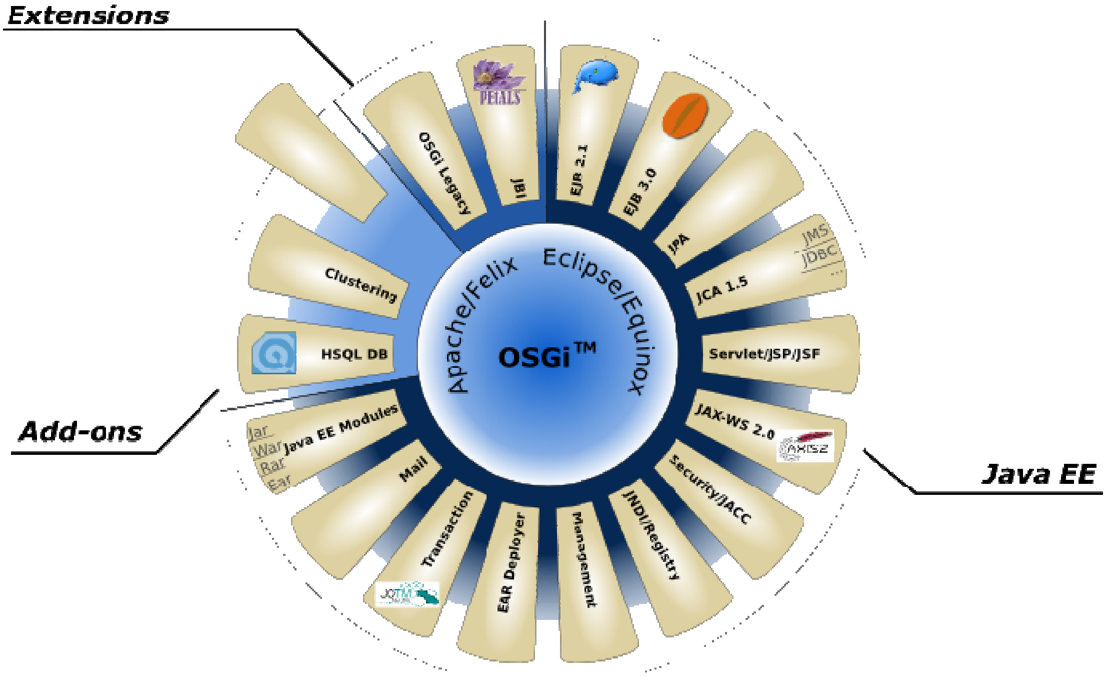
Completely rewritten to be based on OSGi^{TM} modules, JOnAS 5 implements a service-oriented architecture in the application server itself, enabling the server and its services to be dynamically adapted and extended depending on users' needs and the constraints of their environment. As a result, with JOnAS end users benefit from a breakthrough modular and flexible platform, that can be dynamically adapted to their needs, upgraded without interruption of service, extended on the fly with new services and opened up to the multitude of third-party services from the OSGi^{TM} world, such as RFID. With this architecture, that can dynamically adapt itself to the deployed application needs and thus ensure reduced and optimized resource consumption, JOnAS will also be able to target embedded application servers market.

Firstly applied to automotive and home devices, OSGi^{TM} is an open specification defining how to deliver managed services over a network. It defines

- a platform also named gateway, the framework to deploy, execute and manage services
- a set of basic services (like log or http ones)
- and the deployment unit, the « bundle »

Services are implemented by Java objects, packaged in bundles. OSGi^{TM} is now used for middleware, it is for example the basis of the Eclipse plugin deployment support, BEA and IBM are adopting OSGi for their enterprise Java offers. There are several OSGi^{TM} framework implementations like Felix from Apache, or Equinox from Eclipse.

JOnAS code has been entirely refactored, to cleanly isolate technical services from each other, and to package them into OSGi bundles, the resulting architecture is as in the figure below.



JOnAS 5 Architecture

The main benefits of such architecture are

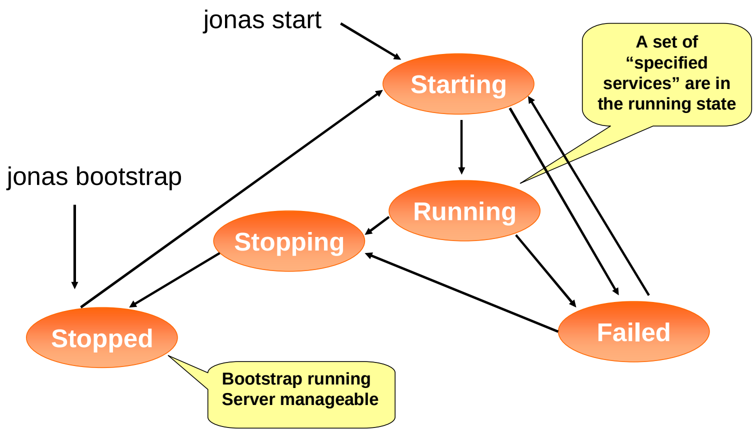
- Dynamic Configuration and re-configuration of servers
  - Services can be stopped, reconfigured, and started at runtime

    - On demand incremental services delivery: services can be started when required by other services or applications

- Modularity
    - Services are delivered in « bundles »
    - Code readability
    - Reduced system footprint by starting only the strictly required services and by stopping the no more required ones
    - Explicit dependencies are clearly defined between services
- OSGi other add ons
    - Remote management
    - Services lifecycle management (incl. versions)
    - Services dependencies management (OBR: OSGi Bundles Repository)
    - Dynamic class loading (flat class loaders)
    - OSGi world accessibility (RFID, probes …)

The first immediate benefit of this architecture for the JOnAS application server is a better server state control, relying on the services states. In addition, services can be controlled individually, at runtime, without stopping the entire server.

There is a new concept of JOnAS bootstrap, corresponding to a server stopped state, where the OSGI platform is running, with only two mandatory services started, the registry and the JMX service, some management MBeans are created and a JMX connector registered for remote management. The administrator specifies a set of required services, to define what will be a running server state (i.e. the server is running if all these services are successfully started), some services may be listed as non critical, meaning that they will not prevent the server to reach the started state in case they cannot start. The figure below illustrates the server life cycle.


jonas start
Starting
A set of "specified services" are in the running state
jonas bootstrap
Running
Stopping
Stopped
Failed
Bootstrap running Server manageable

JOnAS Server State Diagram

2. JOnAS EJB3 container: EasyBeans

Three critical aspects of Enterprise Java application development are covered by the JOnAS EJB3 container EasyBeans:

- Ease of development: coding and testing enterprise application components (EJBs) should be easy
- Efficient implementation results: the resulting applications should deliver high performance
- Ease of deployment: applications should be easy to deploy on target execution environments

EasyBeans is an OW2 project implementing a lightweight and modular EJB3 container. It is a core implementation, meaning that the persistence part is provided by a third party JPA implementation. Currently Hibernate EJB3, Apache OpenJPA, TopLink Essential are supported.

EasyBeans adopts a new container architecture, based on code injection thanks to the ASM OW2 project and on smart proxy for object invocation. The result is a very efficient container with reduced memory footprint (the number of in memory objects has been reduced). Of course EasyBeans provides EJB3 failover and load balancing features.

There are several distributions of the EasyBeans container:

- integrated in JOnAS 5 as EJB3 container
- deployable on JOnAS 4 as a connector (RAR file)
- OSGi bundles
- standalone (including a reduced set of mandatory services)
- deployable with Tomcat or Jetty (WAR file)

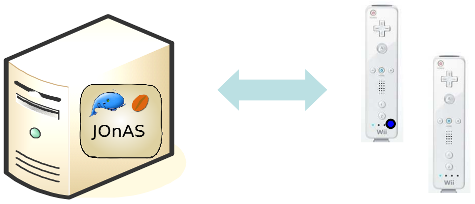
The key objective of Java EE 5, and more especially of EJB3 was « Ease of Development », this was achieved through elements such as annotations, interceptors, dependencies injection, and JPA persistence (programming persistent objects as POJO), suppression of Home interface. EasyBeans has gone several steps further in this direction by making the developer job still easier, it provides

- Automatic hot deployment (also available in JOnAS 5 for all kinds of application)
- exploded archives
- Simple log API
- Large-scale deployments are also simplified with a smart and reduced client API capable of downloading higher level APIs from the server (so there is no specific client side deployment)

Another interesting feature of EasyBeans is the capability to access the OSGi world in a very easy way for the developer. A simple annotation allows to inject a resource corresponding to an OSGi BundleContext, thus providing access to OSGi services from an EJB.

```
/**
* Bundle context that will be injected by the container
*/
@OSGiResource
private BundleContext bundleContextField = null;
```

A JOnAS 5 demonstration available on the OW2 JOnAS web site illustrates this through a Java EE application including web components and EJB3 interacting with OSGi services of Nintendo^{TM} Wiimotes. This Java EE application can control and receive events from the Wiimotes.



JOnAS 5 Demo

3. JOnAS Advanced Management: JASMINe

Complex distributed enterprise application deployments are very difficult to be set up, administered and monitored. The JASMINe tool can dramatically reduce the cost of such administration activities.

As a result of the associated OW2 project JASMINe, JOnAS 5 now benefits from the most advanced cluster configuration and management tools in the Open Source world today.

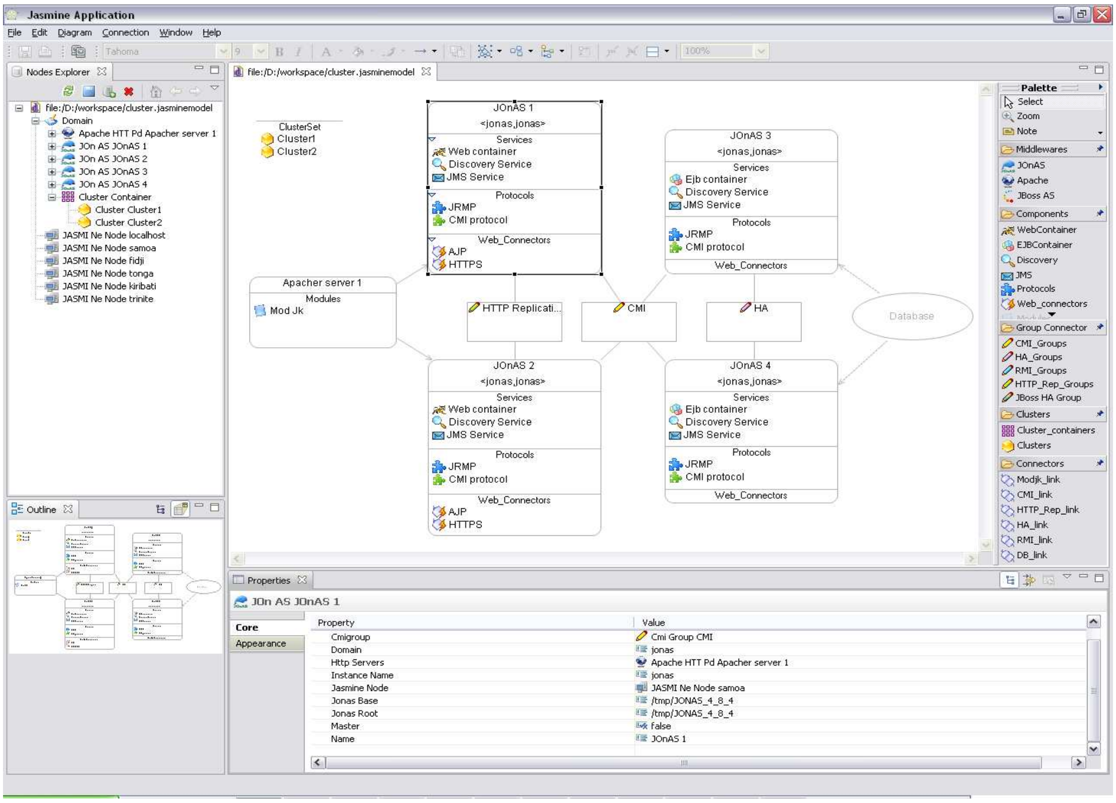
The **design**, **deployment** and **monitoring** of complex SOA platforms, including application servers (clusters), ESB platforms, Workflow and BPEL engines, etc. are very complex and error prone tasks. Configuring just one complete Java EE JOnAS application server may involve up to 46 configuration files, so it is very difficult to configure platforms including clusters of JOnAS and other middleware components by hand.

Java EE cluster monitoring is quite difficult as well: the operator has to deal with a huge number of indicators located at different levels (OS, JVM, Java EE, application) and distributed across different machines. In such environment, error detection is not obvious for a human and may take a long time. Low reactivity is quite problematic for critical systems where the interruption of service must be minimized.

JASMINe is a smart tool providing such facilities which aims at reducing the administration costs.

The first application of the JASMINe tool is JOnAS cluster configuration, deployment and monitoring.

JASMINe provides an RCP GUI, based on Eclipse EMF/GMF, for the designer to describe the cluster configuration. It then relies on the JADE/OSGi framework to deploy this configuration across the infrastructure; JADE is the result of research conducted at INRIA (French Research Institute on Computer Science) regarding systems configuration representation, based on Architecture Definition Language (ADL) and component model, and targeting self management. This GUI provides all JOnAS cluster elements as graphical icons that the user can drag and drop to design a configuration, as shown in the screen shot below.



JOn AS JOnAS 1

Property	Value
Cmigroup	Cmi Group CMI
Domain	jonas
Http Servers	Apache HTT Pd Apacher server 1
Instance Name	jonas
Jasmine Node	JASMI Ne Node samoa
Jonas Base	/tmp/JONAS_4_8_4
Jonas Root	/tmp/JONAS_4_8_4
Master	false
Name	JOnAS 1

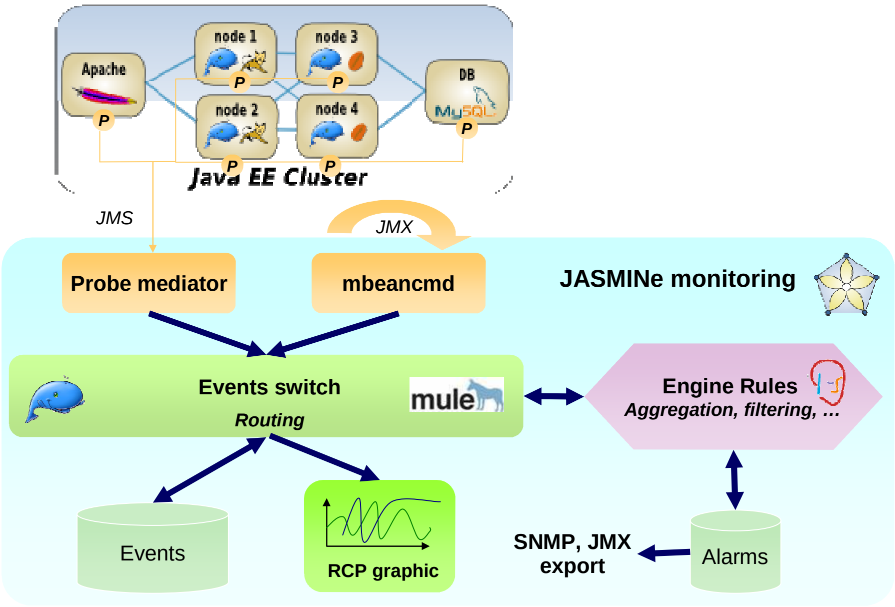
JASMINe Design Interface

Once deployed, the cluster configuration can be monitored through JASMINe, allowing error detection, performance analysis.

JASMINe provides a probes infrastructure, getting information from the managed system through a JMS interface and using an XML format. Collected information comes from System (CPU, Memory, etc.), JVM (Memory Heap, CPU, etc.), application server (error log, ejb instances, datasource, transactions, etc.), and other middleware components (MySQL, Apache, etc.).

An additional way to get monitoring information is the MbeanCmd tool, integrated into JASMINe, which allows to very easily get information from available Mbeans. This tool is available as a Java command (mbean.jar), thus enabling scripting, it relies on the JMX Remote interface and provides the capability to get and set MBean attributes, to invoke MBean methods. It is used for very easily probing the most relevant JOnAS indicators like transactions, datasources, http connectors, threads pools, jms statistics, etc.. It allows file storage (CSV) or database storage of the results, provides a replay mode and a graphic console.

In JASMINe, monitoring information is routed through an event bus (currently Mule), either to an event database, and/or to « live » graphics in the RCP GUI (this last point will be available in a near future). This information is aggregated and filtered through a rules engine (currently DROOLS). This rules engine is used for error detection and to generate alarms.



JASMINe Monitoring Architecture

JASMINe will also enable autonomous behaviour, like self-repair and self-optimization according to a set of management rules. To achieve this, it relies on the JADE framework that will maintain a system representation causally connected to the actual managed system, and that will be used through self-management rules, to get information about the system state and to trigger correcting actions.

4. JOnAS Clustering

Setting up clustering solutions is a complex task that requires the following properties, provided by JOnAS

- Clustering mechanisms should be available and fully functional at all levels of the n-tiers architecture, and should be scalable
- Cluster configuration should be dynamically adaptable to the environment changes
- Cluster deployment operations should be easy and scalable
- Cluster management should be enabled through powerful administration tools

JOnAS offers the perfect Open Source solution to organizations wishing to deploy **highly scalable and available** application services for mission-critical applications. To achieve this goal, JOnAS implements load balancing, replication and failover mechanisms both at HTTP and RMI levels, to deal with both web and EJB components.

Regarding HTTP, JOnAS 5 relies on Tomcat 6 mechanisms and the mod_jk plugin that provides load balancing and session replication. This last aspect has been enhanced in the last tomcat version through a primary-backup mode, enabling both failover and scalability (limiting session replication to a predefined set of nodes, and not replicating to the entire cluster, makes possible the support of failover even in large clusters).

At the RMI level, the JOnAS clustering protocol CMI has been enhanced. CMI v2 now supports both EJB2 and EJB3, being integrated with both EJB containers. Its new design makes it independent of the underlying protocol, JRMP, IIOP or IRMI, and it may also be used outside JOnAS for clustering RMI objects. CMI v2 has been designed for performance, by separating service and control flow, and by handling pools of stubs. CMI v2 brings two key features for dynamic clustering support:

- the load balancing policy is not wired in the stubs, but is dynamically downloaded by the clients from the server, and consequently can be changed at runtime, through the jonasAdmin console (or through JMX). Available policies are round robin, first available, ha singleton, random; the user may also define its own policies
- large scale cluster clients deployment is enabled, since only a reduced API is installed on the client side (smart proxy), independent from the application and from the JOnAS version, other classes are dynamically downloaded from the server

JOnAS 5 also provides a « smooth stopping » feature for cluster nodes, meaning that one server belonging to a cluster can be disabled and finally stopped without service/session interruption. This can be achieved through the jonasAdmin console or the JMX interface, then new connections to the disabled server are refused, while existing clients continue to be served.

Some additional features introduced in JOnAS 5 are related to cluster management, as the capability to visualize the logs of servers belonging to a domain within the jonasAdmin console, and the port of the JOnAS discovery service on top of Jgroups, providing more configuration capabilities (the Discovery service allows to automatically discover servers entering or leaving a domain).

Future work regarding cluster in JOnAS 5 is about dynamic replication at EJB level (some kind of dynamic primary-backup replication mode), cluster cache at the application level by assessing the Terracotta solution, and the introduction of an abstract layer on top of the group communication protocol, in order to easily change from one implementation (like JGroups) to another.

To conclude, JOnAS is a pure open source project hosted by OW2, an independent industry consortium dedicated to developing Open Source code middleware. It benefits from a worldwide developers and users community. It is an enterprise class application server, supported by industrial contributors and users, and providing advanced key features in the area of administration and clustering. JOnAS now adopts a native OSGi architecture, providing modularity and flexibility; it is implemented as a set of OSGi bundles, with technical services implemented as OSGi services. The new server architecture allows new services to be added, or JOnAS provided services to be replaced by alternative service implementations. Services can be started, stopped and re-configured at runtime. JOnAS is tightly coupled to two other OW2 projects, EasyBeans the lightweight EJB3 container and JASMINe, a powerful tool for designing, configuring, deploying and monitoring JOnAS clusters. It is also the cornerstone of the OW2 reference architecture, combining middleware components and targeting dynamic SOAs which are the basis of the flexible, self-adapting information systems of tomorrow.

JOnAS answers the needs of organizations that want to deploy open Java EE application services, while benefiting from the low TCO, the innovations, the flexibility and the large solution ecosystem of a leading international open source software community.