

Technology and Standards

Infiniflow is based on industry-wide standards. This provides customers the peace of mind that they are investing in a non proprietary, vendor-locked solution.

Infiniflow is Java based and, more importantly, leverages two industry standards that are set to transform the enterprise application world, namely **OSGi™** and **SCA** (Service Component Architecture).

OSGi

OSGi is an established and mature industry standard that has been designed from the ground-up to provide a highly agile local JVM runtime. OSGi concentrates on component packaging, component life-cycle management, dynamic service registration and component collaboration, all with the view to achieving a simple, loosely coupled, interchangeable component framework that is both flexible and easy to maintain.



OSGi also addresses the management issues caused by components within business systems evolving at different rates (a.k.a. DLL/Jar hell), as OSGi allows components of the same type, but of different versions, to be used in the same JVM runtime.

OSGi was established in 1999, under the stewardship of the **OSGi Alliance** (www.osgi.org/) and it has cultivated a successful track record across a number of market segments including embedded, 'smart home' and mobile. Members of the OSGi Alliance include a broad range of leading IT vendors from many different markets. OSGi also has an extensive installed base in non-enterprise market verticals, with a number of high quality commercially supported open-source platform implementations – **Equinox** (www.eclipse.org/equinox/), **Felix** (cwiki.apache.org/FELIX/index.html) and **Knopflerfish** (www.knopflerfish.org/).

Since late 2005 there has been a huge surge in interest and activity in OSGi from the enterprise software vendor community which has led to the establishment of an Enterprise Expert Group (EEG) - within the OSGi Alliance - of which Paremus is an active member.

JEE application server vendors are moving en-mass towards being built on an OSGi component framework (see **InfoQ August 2006** - www.infoq.com/news/OSGi-Use-Increases). Indeed, all the leading application server vendors (including IBM, Oracle, BEA and RedHat) and open source projects (including ObjectWeb's Jonas) have announced the intention to decompose their traditional monolithic application servers into framework solutions with pluggable OSGi components.

The open source **Eclipse** (wiki.eclipse.org/index.php/FAQ_How_does_OSGi_and_the_new_runtime_affect_me%3F) development is already based on the OSGi component standard. The popular Spring Framework also already supports OSGi and the Spring 2.1 release due in mid 2007 will be fully **OSGi compliant** (www.springframework.org/osgi) as the framework will be decomposed in to many OSGi bundles. Benefits envisioned include:

- Increased separation of application logic into modules
- The ability to concurrently deploy multiple versions of a module
- The ability to dynamically discover and use services provided by other modules within a the local JVM
- The ability to dynamically deploy to, update, and remove modules from, a running system
- Use of the Spring Framework to instantiate, configure, assemble and decorate components within and across modules.

- A simple and familiar programming model for enterprise developers to exploit the features of the OSGi Service Platform

Spring's OSGi support also makes the development of OSGi applications simpler, and developers more productive, by building on the ease-of-use and power of the Spring Framework. For further details please see the **OSGi Alliance** website.

Service Component Architecture (SCA)



Service Component Architecture (SCA) is one of the key deliverables of the Open SOA Collaboration.

To quote from the **Open SOA Collaboration** website, it *"represents an informal group of industry leaders that share a common interest: defining a language-neutral programming model that meets the needs of enterprise developers who are developing software that exploits Service Oriented Architecture characteristics and benefits"*.

SCA has huge enterprise software vendor industry support (over 20 vendors e.g. BEA, IBM, Oracle, Red Hat and Sybase, and many others including Paremus). It was announced in March 2007 that SCA had completed its incubation period and was being formally submitted to OASIS for advancement through its open standards process.

SCA provides an industry standard way of describing which components are required, and how they should be wired together to create the desired composite application and system. SCA provides the equivalent of standard architectural blueprints; a way of describing a system in terms of its composite services, the components that are used to create those composite services, and how these entities are interconnected.

SCA provides a structural description for a composite System, which may include required application and infrastructure components, their configurations, and which bindings to use. An SCA System is described in terms of a collection of Composites, where each Composite may itself be described in terms of further Composites, or atomic Components. SCA Systems are implementation neutral; neither protocol or application infrastructure are mandated. Connectivity between entities within an SCA System is defined by the bindings associated with each entity in the System. For example, in a System initially comprising of two Composites that communicate with each other via SOAP, the decision to move from SOAP to JMS, is simply achieved by changing the relevant SOAP SCA bindings to the appropriate JMS SCA bindings.

An SCA System description need not only describe a composite application that is constrained to a local runtime; an SCA System may also comprise of a number of services spanning many nodes, including all required software infrastructure components; specific messaging, coordination, caching and transactional middleware services required by the System, and associated SCA bindings for those services.

For further information on Service Component Architecture (SCA) please see the **OSOA Collaboration** (www.osoa.org/) website.