OSGi Adoption and Usage within the Enterprise #2

What obstacles must OSGi overcome to achieve widespread adoption within the enterprise?		
		Response Count
		155
	answered question	**155**
	skipped question	**107**

Response Text		
1	Tooling	Oct 14, 2009 6:22 PM
2	Getting a decent build system that isn't P2, or otherwise IDE-dependent, for OSGi bundles.	Oct 14, 2009 6:38 PM
3	Less invasive (go Blueprint!). Making packaging a bundle as seamless as creating an EAR or WAR. Facilitating deployment. Basically, making the transition easier for normal Java devs.	Oct 14, 2009 7:12 PM
4	Tooling mainly.	Oct 14, 2009 7:43 PM
5	Documentation. Tutorials and cookbooks.	Oct 14, 2009 8:01 PM
6	Making legacy applications works and addressing some performance problem like memory usage and startup time.	Oct 14, 2009 8:34 PM
7	deprecate "required-bundles": keep it simple	Oct 14, 2009 8:37 PM
8	I think many developers are not familiar with it. It should be promoted more being compared to the ways of making enterprise software used so far. Plus there should be more books talking about OSGi.	Oct 14, 2009 8:39 PM
9	Tools still need to improve a lot.	Oct 14, 2009 9:03 PM
10	Tools to simplify the construction of manifests. Visualization of dependencies and reporting capabilities for container instances to show runtime interactions and requirements. A happy path from current SPI-style and classpath interactions for Java into a more prescriptive modular usage of export and import.	Oct 14, 2009 9:29 PM
11	Ignorance. Btw, I think you missed service-oriented architecture in question 7. OSGi services are only the greatest thing ever :)	Oct 14, 2009 9:37 PM
12	Tool support, Designing support in the sense of some material / tools teaching people how to think in OSGI	Oct 14, 2009 9:55 PM
13	There's a need for better tooling & IDE support. More capable framework consoles are also needed. Standards and best practices that enable gradual migration from existing JEE based systems are also essential. More high level programming models for OSGi - currently using SpringDM, but would like alternatives.	Oct 14, 2009 10:02 PM
14	Tooling	Oct 14, 2009 10:08 PM
15	OSGi must gain enough mindshare/market to remain the de facto option, even after Jigsaw/JSR294 are available (and baked into the JDK)	Oct 15, 2009 12:31 AM
16	ignorance better automated builds using manifest first vs. maven pom.xml	Oct 15, 2009 12:49 AM

Response Text		
17	Better tooling support for development and testing, additional tools for managing and monitoring the runtime. More 3rd party packages need to ship with OSGi metadata. Having the ability to verify correctness at compile time instead of waiting until runtime would also be useful (the mismatch between compile and run time is a problem with our large team).	Oct 15, 2009 1:54 AM
18	The misconception that it is complex and heavyweight.	Oct 15, 2009 2:55 AM
19	* availability of every library used in enterprise development as fully tested OSGi bundle. It should be out of the box and directly from the horse's mouth. * enterprise libraries should now consider providing add-ons that can expose their regular libraries as OSGi services * Finding dependencies for a combination of libraries is still a mess, this should be addressed (libd / obr / karaf-features etc). A common robust and well accepted standard is required. * Although not advised, developer has to often play with startlevels of bundles as the applications tend to get bigger, * A market place where I can shop and add bundles to my apps :)	Oct 15, 2009 3:00 AM
20	compatibility with JEE application servers	Oct 15, 2009 5:25 AM
21	Developers experience must become smoother (updates of bundles often causes 'error-hunting' due to missing/wrong constraints). So Dev and Architects can concentrate on their job/project and acutally use OSGi's benefits.	Oct 15, 2009 5:40 AM
22	Keep the asserts of Traditional Java EE Development and Deployment Model.	Oct 15, 2009 6:28 AM
23	ease of use. better tools.	Oct 15, 2009 6:36 AM
24	integration with application server such as tomcat should be simple and transparent	Oct 15, 2009 7:02 AM
25	- More entreprise class OSGi servers, i.e. building upon an OSGi container and adding log managment, startup/shutdown, tooling for isntalling/updating bundles, etc. - OSGi'fying most of the libraries out there - Continue to improove the tooling	Oct 15, 2009 7:21 AM
26	More IDE support outside of the Eclipse-Ecosystem (Sun javac, ant, maven etc.)	Oct 15, 2009 7:24 AM
27	convince managers to use OSGi	Oct 15, 2009 8:39 AM
28	Availability of a OSGi-based Killer Application (non-IDE)	Oct 15, 2009 8:55 AM
29	Tools! Simple tools that work and does not require any further hacking.	Oct 15, 2009 9:17 AM
30	1. Tooling: better integration with IDEs 2. Documented design best practices	Oct 15, 2009 9:20 AM
31	OSGi should be treatened as a runtime solution and not as a end-user development api. As such, building up a solid toolchain from development to deployment is crucial. J2EE Developers should not need to face "Missing-ImportPackage Exception.." and friends. Deployment solutions must be more widespread (DeploymentPackage, OBR, Apache ACE for example). Too many companies who start to adopt osgi without inhouse experts (which is quite common) fizzle with low level details across the whole development team+process.	Oct 15, 2009 9:23 AM
32	Java EE API are usually not OSGi friendly. This is a challenge to have them working nicely in an OSGi environment.	Oct 15, 2009 9:33 AM
33	- Better version management based on OSGi components - Not to be so connected to Java world (classloaders are example)	Oct 15, 2009 9:35 AM
34	Must be supported as first-class citizen by the enterprise grade JEE servers: weblogic, websphere, etc.	Oct 15, 2009 9:38 AM
35	(Better) and standardized integration with JEE (esp. JPA, webapp) and more documentation/books.	Oct 15, 2009 9:45 AM
36	Education -- Design and Deployment Best Practices Better Development and Testing Tools Clear Blueprint Architectures for different application types (e.g. Web apps)	Oct 15, 2009 9:48 AM

Response Text		
37	More documentation for achieving simple things - less of this 'Spring will handle everything' - it's nice there are frameworks for everything, but using them doesn't help people learn wtf is actually going on under the hood!	Oct 15, 2009 10:03 AM
38	1. Visibility (and what it /actually/ is) 2. Tooling around OSGi (good but often arcane to people new to OSGi) 3. Better stability in the specifications and better specifications (e.g, DS changed during r4, and the Component Model RFC seems to propose to replace it with a standardized Spring DM, which is good but DS must provide backwards compatibility using the same foundations) 4. More bundles for everything. It would be great if the OSGi Alliance and partners could find a way to provide OSGified bundles and evangelism for all the Apache Java stack, for example.	Oct 15, 2009 10:11 AM
39	It's mainly an issue of selling it to project sponsors. It mainly addresses non-functional requirements, and so is a tough sell compared to adding new features. I'm also puzzled about how to evolve an existing application to a osgi architecture - what are the best practices for a phased (i.e. probably multi-year) migration? I'm hoping some of the upcoming OSGi books will address this issue and am very annoyed with myself for missing the recent presentation on this subject in London ;)	Oct 15, 2009 10:26 AM
40	There needs to be a good book available about OSGi	Oct 15, 2009 10:32 AM
41	Closer integration with java and java runtime packaged into bundles. Better tool support, e.g. maven (and a official maven repository for standard bundles would be great). Better documentation aimed at developers and applicable design patterns	Oct 15, 2009 10:37 AM
42	Better Startup time. Better handling of dependencies. Better development tools specifically for OSGi framework.	Oct 15, 2009 10:58 AM
43	An application server with an OSGi runtime container would be great.	Oct 15, 2009 11:07 AM
44	Classloading issues when using remote services (JNDI, RMI). More third party libs provided as bundles.	Oct 15, 2009 11:10 AM
45	understanding of the benefits	Oct 15, 2009 11:21 AM
46	Improve build and deployment mechanisms. More centralized documentation.	Oct 15, 2009 11:40 AM
47	Better support / documentation / examples on how to use together with hibernate / AOP	Oct 15, 2009 11:48 AM
48	1. Too many bundles to manage when creating an Enterprise application - need a library concept that includes a consistent set of bundles at a library version 2. Support for other web containers like OC4J to deply web bundles to 3. Simpler versioning, who keeps track of all these package versions, great in theory but a pain in practice to maintain 4. better tooling - e.g. Bundle-Version in manifest (say the 3rd digit) to be incremented automatically on a project code commit	Oct 15, 2009 12:00 PM
49	It must be easier to interact between bundles - more IDE support. ClassPathloading-Issues must be easier to debug.	Oct 15, 2009 12:02 PM
50	Books / Case Studies	Oct 15, 2009 12:06 PM
51	OSGi is Java based, and in some circles Java is considered a legacy language. For all its promise of modularity and hot swappability it is still very difficult to write modular, hot-swappable, well behaved OSGi code.	Oct 15, 2009 12:11 PM
52	managing and observing service-dependencies	Oct 15, 2009 12:18 PM
53	OSGi is a great concept for nudging people toward true component engineering. However, because it is so tightly bound to Java, it misses an opportunity to really have the universal impact it could have.	Oct 15, 2009 12:18 PM
54	lack of consulting and education services	Oct 15, 2009 12:33 PM

Response Text		
55	Better component frameworks that support yet abstract away the problems inherent with the dynamism of the runtime are needed. Declarative services aren't flexible enough and blueprint is too fine-grained. The Apache Felix iPOJO project, however, looks very promising.	Oct 15, 2009 12:33 PM
56	Tooling	Oct 15, 2009 12:35 PM
57	Most importantly: not listening to the masses of mediocre J2EE fanboys who want to "fix" OSGi without ever having read the actual spec.	Oct 15, 2009 12:37 PM
58	usability, user friendlyness, management tools easier to use . .	Oct 15, 2009 12:42 PM
59	-	Oct 15, 2009 12:44 PM
60	Better tooling.	Oct 15, 2009 12:45 PM
61	The development stack is quite confusing and heavy right now. There are plug-ins for Maven, etc. that help but it can still be burdensome to get dependencies straight in manifest and such. Also, there needs to be more time put into creating and maintaining bundles for OSS. SpringSource is doing a nice job, but they can not do it alone.	Oct 15, 2009 12:46 PM
62	Better IDE and other tooling support.	Oct 15, 2009 12:49 PM
63	- Complexity (cf : the service registry). - It's sometimes difficult to understand error messages / dependencies problems (a bundle may compile but won't work at runtime). - Standardization, should I use Require-Bundle or Import-Package ? - Not very easy to use in coordination with maven.	Oct 15, 2009 12:52 PM
64	Development tools are a big gap right now. There isn't enough in Eclipse, Spring tools are too tied to dm, etc. This and a dearth of OSGi education has been a barrier to wider spread adoption within my customer base.	Oct 15, 2009 12:56 PM
65	Industry support	Oct 15, 2009 12:57 PM
66	Prove widespread successful use in commercial products	Oct 15, 2009 1:02 PM
67	tool support needs to keep evolving mainstream adoption will only follow widespread recognition of the value of modularity and the benefits for agile development	Oct 15, 2009 1:15 PM
68	There has to be a -need-. For smaller systems, it's just not entirely necessary.	Oct 15, 2009 5:00 PM
69	Core concepts of OSGi are imperative - there needs to be better and more formal tutorials/books/etc. on modularity, lifecycle, services. As well as pragmatic use of mature containers like Equinox and off-shoot technologies like P2 - that make OSGi a reality and not just a specification.	Oct 15, 2009 5:03 PM
70	OSGi is an overengineered solution that causes far more problems than it solves. After a year of developing with OSGi we were so disgusted with all of its problems and complexities, that we evaluated using Maven to manage library conflicts (our main reason for adopting OSGi in the first place). In two weeks we were sold and shut down the OSGi project. Re-did everything as regular Java projects managed by Maven.Everyone was way happier. OSGi is a con, one of the biggest productivity hits I've ever seen. Right up there with J2EE 1.4 and Struts. So, yes we evaluated OSGi for 8 months and dumped it. Never going back. Now we test and build via a simple "mvn package" instead of having to deal with PDE, target definitions, classloader conflicts, ClassCastExceptions and all the other "fun" stuff that OSGi brings. Never going back to OSGi development.	Oct 15, 2009 5:17 PM
71	Better build / deploy / provision / configure tooling.	Oct 15, 2009 5:28 PM

	Response Text	
72	The biggest problems with any enterprise environment revolve around integration issues. Leveraging OSGi's strengths in modularity and versioning to simplify integration of existing systems would be a major selling point. Slow startup time in an enterprise environment can be a problem, though not necessarily caused by OSGi itself. Rather, the delayed startup is often due to aggressive scanning of zip files by antivirus solutions. Since OSGi environments consist of many jar (zip) files, the delay caused by real-time virus scanners for each of the zip files can get out of hand quickly.	Oct 15, 2009 5:59 PM
73	Convincing the community to design their applications to use OSGi, but not require OSGi. Dependency management is critical, as is application independence from OSGi itself. OSGi should be used as a framework, not a platform. Applications don't need OSGi to run, but OSGi certainly enables developers to design and build highly modular software that can be dynamically deployed, updated and maintained.	Oct 15, 2009 6:16 PM
74	Applicability to the web.	Oct 15, 2009 6:23 PM
75	tool support always helps, and fantastic documentation	Oct 15, 2009 6:48 PM
76	Mountains of legacy code.	Oct 15, 2009 7:18 PM
77	Politics not technical obstacles	Oct 15, 2009 7:27 PM
78	Many of popular frameworks and some good implementations of most relevant standards have to be OSGi compatible and capable.	Oct 15, 2009 7:50 PM
79	In my world of investment banking, the biggest obstacle is getting people to realise that they have a problem and that OSGi will help solve it. 100 man teams struggling to produce components assembled by another team is a regular occurrence. If you mention OSGi they usually think its 'overkill'.	Oct 15, 2009 8:00 PM
80	hm... good question. Perhaps Annotations would be a great secound way for the configuration by bundles and in Eclipse Equinox "Search Engine Classes" for Services and Bundles are missing.	Oct 15, 2009 8:00 PM
81	Clear migration path from traditional development. Support for enterprise features.	Oct 15, 2009 8:08 PM
82	become more useable - is like pulling teeth atm to convert an existing app to OSGi	Oct 15, 2009 8:17 PM
83	In most implementations and example I have seen, the hurdle of adopting OSGi does not seem close to its benefits. Reminds me a lot of the old EJB model, an inelegant solution that requires a wholesale architecture change for not much benefit.	Oct 15, 2009 8:24 PM
84	-	Oct 15, 2009 8:25 PM
85	good samples/examples ! the example that my boss sent me was *huge*- hard to wrap my head around.	Oct 15, 2009 8:39 PM
86	Simplify usage, more OSGi bundles available on maven repositories or OSGi conversions of libraries.	Oct 15, 2009 9:11 PM
87	I've been using OSGi for about 8 years now in projects ranging from embedded to server, so I'm not the one to ask about obstacles.	Oct 15, 2009 9:42 PM
88	Tooling, Framework independent instrumentation, OSGiefication of Core JSRs (like JPA, JTA,...), trusted bundles with a reliable and well known version scheme.	Oct 15, 2009 10:00 PM
89	needs to be simpler :-)	Oct 15, 2009 11:39 PM
90	1. Finding a budget of Market 2. Convenience Plan for Stable Versioning 3. OSGi Man-Power	Oct 16, 2009 12:14 AM
91	Number of available supported web frameworks.	Oct 16, 2009 12:39 AM
92	Number of available supported web frameworks.	Oct 16, 2009 12:39 AM
93	There's a lack of introductory and advanced books and tutorials.	Oct 16, 2009 1:21 AM

Response Text		
94	OSGi is fairly limited to Java. Java is dying language. Port it to .NET and I would use it again.	Oct 16, 2009 2:02 AM
95	Classloader issues with legacy libraries and frameworks. Better debugging support for strange ccl errors.	Oct 16, 2009 3:07 AM
96	- Lack of containers/application templates for managing the deployment of applications - Compatibility with third-party libraries (i.e. some libraries are not designed for deployment in an OSGi container)	Oct 16, 2009 4:22 AM
97	Better integration with JEE.	Oct 16, 2009 5:08 AM
98	third-party libraries should be fixed/converted to OSGi	Oct 16, 2009 5:27 AM
99	The memory consumption is definitely a bottleneck .	Oct 16, 2009 5:29 AM
100	More quality open-source bundles.	Oct 16, 2009 6:27 AM
101	- Need to simplify programming with osgi - Relate osgi programming with the OO programming and provide a transition path from the non-osgi programming to osgi-based programming - Enterprise software is quickly getting complex, distributed and outsourced. Need to emphasis how OSGI can help in addressing these concerns for enterprise development. - People are just getting comfortable with the webservices and SOA thingies. And now comes another programming model: OSGI. Need to provide rationale and motivation to learn the yet another new way of programming.	Oct 16, 2009 6:42 AM
102	Greater awareness of OSGi and its benefits (to achieve a better availability of OSGi-ready software components), stabilization especially of startup-behaviour. Comment on survey: Question 6 is incomplete, needs additional entry for "pure non-web-related server software", Question 8 lacks a free text box for additional possibilities like "getting bugs fixed for third-parts bundles".	Oct 16, 2009 6:56 AM
103	Java 5 support	Oct 16, 2009 7:43 AM
104	communicating benefits and consequences	Oct 16, 2009 7:57 AM
105	A better (IDE) way to discover and use offered services (that's why I prefer to use Eclipse extension points, since I can explore them easily)	Oct 16, 2009 8:01 AM
106	Straightforward migration path from JEE.	Oct 16, 2009 8:12 AM
107	Installing and configuring OSGi frameworks as standalone environments should be simplified.	Oct 16, 2009 8:22 AM
108	Building bundles is quite difficult. Right now I use maven-bundle-plugin, which eases the pain.	Oct 16, 2009 8:25 AM
109	It must provide an easier way to deploy typical Java webapps inside the OSGI runtime.	Oct 16, 2009 9:15 AM
110	Improved support for both unit and integration testing form within every popular IDE. Improved packaging constructs (such as SpringSource plans). Improved support for crosscutting concerns (aspects).	Oct 16, 2009 9:22 AM
111	Compatibility with existing common component provided. Database connection pooling or Resource Pooling or sharing.	Oct 16, 2009 10:07 AM
112	I have been in Java/J2EE development for almost a decade and had never heard of OSGi until now.	Oct 16, 2009 10:46 AM
113	It must become much simpler to get simple things done. Specifications like split packages can cause much trouble for others.	Oct 16, 2009 10:49 AM
114	knowledge	Oct 16, 2009 10:55 AM
115	- Libraries that are not yet compatible to the dynamic OSGi nature (e. g. new File() other than FileInputStream()) - Too little literature about creating / structuring web applications on a OSGi-basis	Oct 16, 2009 11:43 AM
116	Classloading for libraries (first of all Hibernate) is sometimes tricky.	Oct 16, 2009 11:51 AM

Response Text		
117	adoption/implementation of the standard in a uniform way	Oct 16, 2009 12:04 PM
118	In the case of Java EE solutions: More seemlessly blend into the current set of technologies. It will be hard to convince customers to drop i.e. WAS / EARs and i.e. completely migrate to Spring dm Server instead.	Oct 16, 2009 12:20 PM
119	Tooling, Documentation (i.e. best practices), JEE Apis must work in OSGi to leverage existing developer communitiy	Oct 16, 2009 1:43 PM
120	At a platform implementation layer it will find widespread adoption, although the overhead and exactness required of developers to accurately represent dependencies is a challenge that will create some anti-bodies. From an application developer perspective, they likely will end up using it indirectly or through simplified models; raw development against plain OSGi directly is an exact science that probably will need the help of tooling and/or simpler models to become productive and simple enough to gain wide adoption.	Oct 16, 2009 2:10 PM
121	full, 2-way integration with Maven	Oct 16, 2009 3:00 PM
122	Development tools, JEE feature, App Server support	Oct 16, 2009 3:12 PM
123	Should be adopted by the main application servers such that using it on the server side does not mean forgoing the features available on application servers.	Oct 16, 2009 3:29 PM
124	OSGi is trying to change the very process and toolset of an enterprise developer, and the benefit is often not there to make such a bug change.	Oct 16, 2009 5:44 PM
125	Better tooling. Higher visibility beyond dev circuit.	Oct 16, 2009 6:08 PM
126	Rather than buried within existing product lines - OSGi needs to be exposed to the development community at large - not just Spring, but iPOJO, DS and all variants. The recent Aries project is a step in the right direction. So to is the Apache/SIGIL tooling project.	Oct 16, 2009 6:08 PM
127	Building and packaging for deployment. Primitive test frameworks.	Oct 16, 2009 6:29 PM
128	Lack of understanding	Oct 16, 2009 7:14 PM
129	Steep learning curve, immature toolset, most tasks are currently harder when compared to traditional j2ee rather than easier because of the immaturity of the tools and platform.	Oct 16, 2009 7:20 PM
130	Complexity	Oct 16, 2009 7:41 PM
131	Top notch tool support! PDE is great, but we need it to be even better.	Oct 16, 2009 8:59 PM
132	none	Oct 16, 2009 9:11 PM
133	tooling support is missing	Oct 16, 2009 10:38 PM
134	tooling	Oct 16, 2009 11:41 PM
135	Usability (read: tools) could be better	Oct 16, 2009 11:52 PM
136	Better tooling support.	Oct 17, 2009 1:40 AM
137	Awareness. In many cases, OSGi technology is already in the enterprise and many don't necessarily know it.	Oct 18, 2009 12:35 AM
138	Not sure if there exists any obstacles, but the tooling support could be better (dev and test). Reduce confusion about existing similiar solutions: Spring, Spring DM, iPojo, BluePrint, Declarative Serives etc	Oct 18, 2009 10:18 AM
139	Simplicity, it adds a lot of complexity in development.	Oct 18, 2009 1:57 PM
140	Tooling, tooling, tooling. Spring Bundlor is a first nice try. App-Servers should be able to download OSGi dependency bundles from the web like Maven does it.	Oct 18, 2009 2:18 PM
141	None!....bundles should be specific to the app....i see no benefit is generic support. "Supports OSGI" is an implementation detail not an API spec.	Oct 18, 2009 5:04 PM

Response Text		
142	From what I've investigated so far (Spring DM, Platforms etc) it looks mostly ready to go. What would be of greatest benefit is platforms such as Tomcat supporting it straight out of the box. SpringSource look to be aiming to do that sort of thing already.	Oct 18, 2009 8:40 PM
143	Better tooling and education.	Oct 19, 2009 1:14 AM
144	Better availability of good tooling to create and modify bundles. Genuinely distributed runtime frameworks.	Oct 19, 2009 11:11 AM
145	Currently, I have trouble finding good documentation on what OSGI really provides, as well as documentation on best practises etc in OSGI.	Oct 19, 2009 1:34 PM
146	More awareness of the issues that OSGi solves. Many developers I find don't understand they have a problem, and that modularity is the solution. The overall state of components and modularity in the industry is appalling.	Oct 19, 2009 2:32 PM
147	Framework adoption.	Oct 19, 2009 2:38 PM
148	Must resolve the issues with the Jigsaw project. Should define build time equivalent of its runtime dependency resolution system.	Oct 19, 2009 2:52 PM
149	distributed osgi	Oct 19, 2009 3:37 PM
150	Better documentation, tutorials, (esp. some dealing with special issues). In case of Felix, better error messages to ease debugging.	Oct 19, 2009 4:08 PM
151	Provide usecases and advantages for us working with good old centralized webapps.	Oct 19, 2009 4:26 PM
152	Migration path, support for legacy [java] applications	Oct 19, 2009 4:31 PM
153	Complexity is a huge obstacle. It is hard to explain what OSGi does because of it's technical nature. The business value of OSGi must be explainable in terms that not very technical users can understand. Also, OSGi needs to paint a very clear picture of it's relationship with JEE - it's at one level competitive and at another level it does not care. This is a confusing message for many users.	Oct 19, 2009 5:53 PM
154	I have read about OSGI several times and what it is/does and why I should care never seems to stick. Maybe I'm daft, but something that describes "in the old days we had to do XX and it sucked because of YY and now OSGI comes along and it looks like ZZ and so we rejoice" would make it simpler to understand.	Oct 19, 2009 6:40 PM
155	What is OSGi and why do I care?	Oct 20, 2009 1:41 AM