

OSGi Technology for Service Providers

2009-11-05

Introduction to OSGi Alliance

- The OSGi Alliance is a worldwide consortium of technology innovators that advances a proven and mature process to assure interoperability of applications and services based on its component integration platform.
 - It specifies, promotes and certifies a component-oriented service platform.
 - There are technical expert groups for the base platform and various industries – Core Platform, Residential, Mobile, Enterprise and Vehicle.
 - The OSGi Residential Expert Group began in November 2006 to work on the requirements and specifications to tailor and extend the OSGi Service Platform for fixed network connected gateways / routers and connected devices.
- The OSGi Alliance celebrates its 10th anniversary in 2009

What is OSGi?

- OSGi technology provides a proven service-oriented, dynamic module system for Java™
 - Robust, mature and consistent service programming model
 - Modular execution environment for services and applications
- The OSGi Alliance provides a modular embedded execution environment for applications and services and a consistent programming model for building applications. It is the only standardized embedded SOA available.
- Due to its modular architecture the OSGi Service Platform enables
 - New business models and opportunities
 - Reduced total cost of ownership
 - Improved time to market

Different views of “A Service”

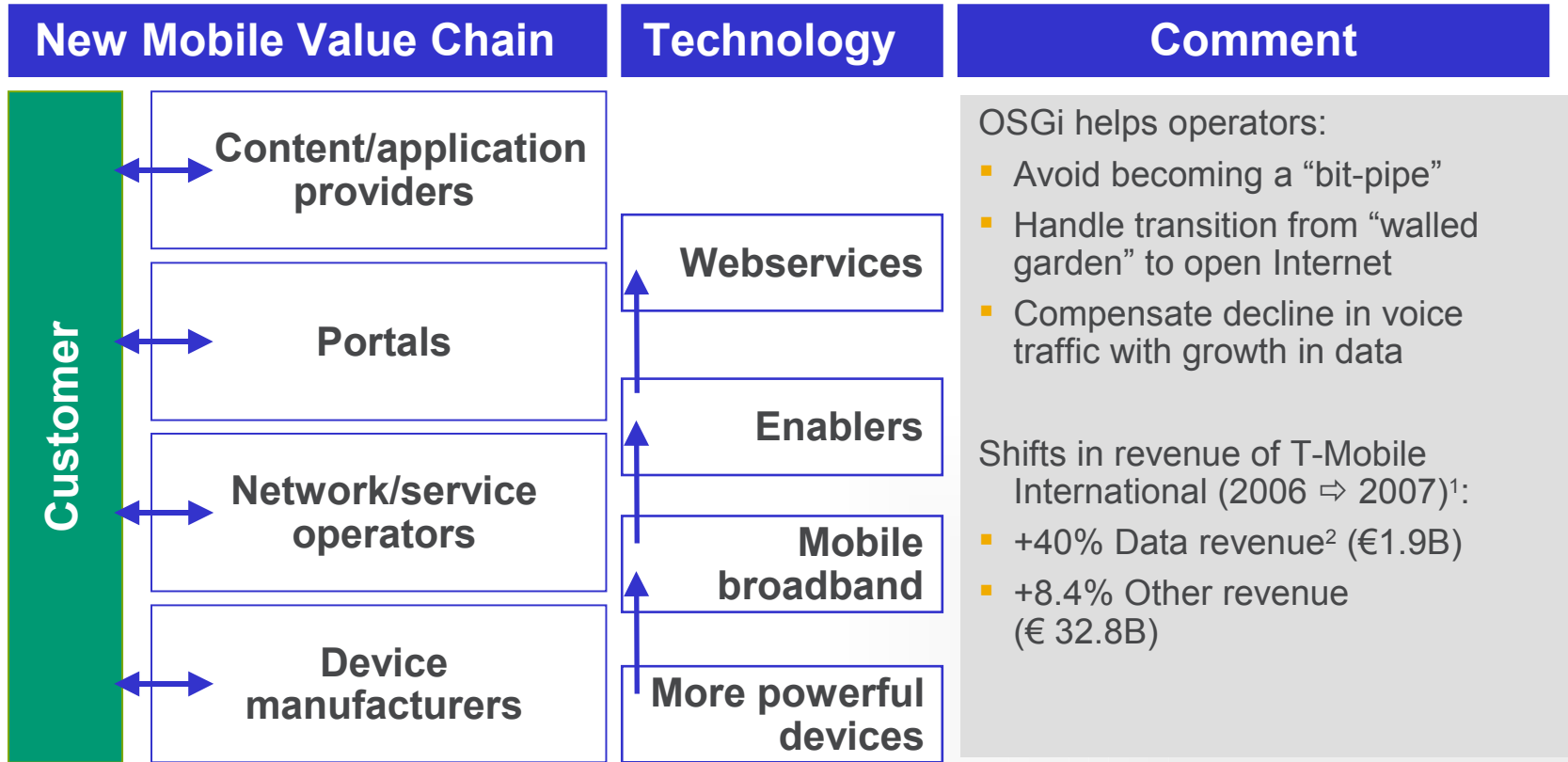
- There are different views of a “service” from the perspective of a Telecommunications Operator as compared to within OSGi Specifications
 - A “service” to an Operator is something a service subscriber can pay for, an end-to-end service like “Voice” or “Broadband” or a service feature like “Call-Waiting” or “Call-forwarding”
 - The definition of a “service” within OSGi Service Platform specifications is generally a much smaller thing, it’s the advertised interface to a software Lego® block (in an OSGi bundle), “...A service is a normal Java object [a POJO] that is registered under one or more Java interfaces within the [OSGi] service registry...”
 - Formal OSGi definition: “Service – An object registered with the service registry under one or more interfaces together with properties. This object can be discovered and used by bundles.”
- **On the following slides we will use the term “service” as defined by Telecommunications Operators**

Service Providers & Operators

Today's challenges in the cross-industrial world for service providers.

- In the face of increasing consumer requirements and expectation, service providers need to:
 - Create new (revenue-generating) personalized services
 - Offer added-value to the end user to differentiate from competitors
 - Re-use existing service elements to offer more complex services
 - For example, the call waiting and call forwarding service elements could be packaged in your calling plan
 - Improve ease-of-use for the consumer
 - Reduce time to market
 - Reduce cost of ownership

Shifts in Mobile Operators' Value Chain: From few to multiple “End-User” Interfaces

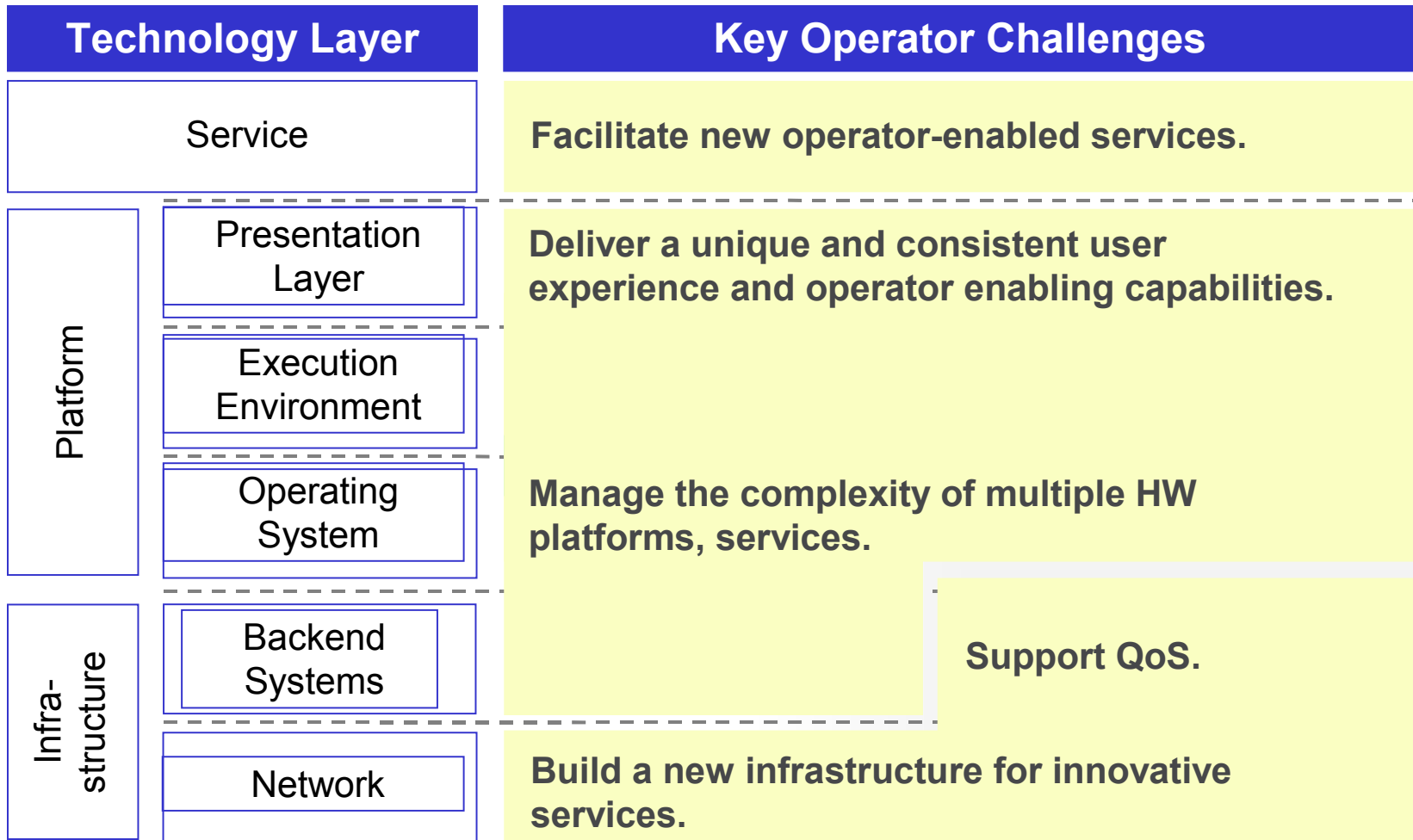


¹ In the T-Mobile countries: Germany, USA, UK, Netherlands, Austria, Czech Republic, Poland, Hungary, Slovakia, Croatia, Macedonia, Montenegro

² Not including SMS

After: The War of the Mobile and Internet Worlds: Regulations and Corporate Players; Jarkko Vesa; ESCP-EAP 2007

Service provider's technical "To Dos" and possibilities to differentiate on technology stack layers



OSGi Technology can play a role here

Key Operator Challenges

Facilitate new operator-enabled services.

Register and expose APIs for 3rd party developments in an OSGi Service Framework

Deliver a unique and consistent user experience and operator enabling capabilities.

Deployment, customization, management of services, and lifecycle management to optimize offers for the customer

Manage the complexity of multiple HW platforms, services.

Java based service framework abstracts OS and HW and enforces SOA based architectures

Support QoS.

Deployment of OSGi Bundles to bridge incompatible QoS schemes to ensure end-to-end QoS

Build a new infrastructure for innovative services.

Business Development

OSGi Technology offers Service Providers many opportunities for strategic market differentiation.

New business models

- Creation of flexible new business models
- Service aggregation through use of pluggable service elements
- Extend current offerings with innovative services
- Open to third-party providers and developers

Deployment of new services

- Rapid development and deployment of new services
- Customized product variations that meet diverse customer needs
- Leverage an established OSGi ecosystem to jumpstart
 - Re-use of modular components

OSGi Technology enables

- Continuous after-sales and maintenance
- Efficient cooperation with first-tier providers and after-market sales
- Product configuration at any point in time
- Ability to diagnose, start-stop-update and install/remove services remotely
- Remote maintenance/ dynamic update

Maintenance

- Dynamic download, maintenance and removal of selected components

- Easier installation, upgrade, and removal of services with no downtime
- Enables automatic upgrades
- Simplifies end-user maintenance

Support for end-users

Rapid development and deployment of new services reduces time-to-market.

Easy product creation

- Customized product variations that meet diverse customer needs
- Leverage an established OSGi ecosystem to jumpstart telco solutions (tools, libraries, system integrators, applications, etc.)
- Re-use of modular components
- Develop future-proof products with extended life cycle without retrofit -decrease in expensive recalls
- Ability to use freely available application development tools
- Leverage 3rd party developers

Reduce deployment effort

- Portable across H/W platforms and software OSS (Operations Support Systems)
- Needs only partial deployment - available components are reused
- Hot deployments
- Common provisioning formats
- Dynamic software downloads

Reduced time-to-market

Consumer satisfaction increases with better aftermarket sales and field maintenance

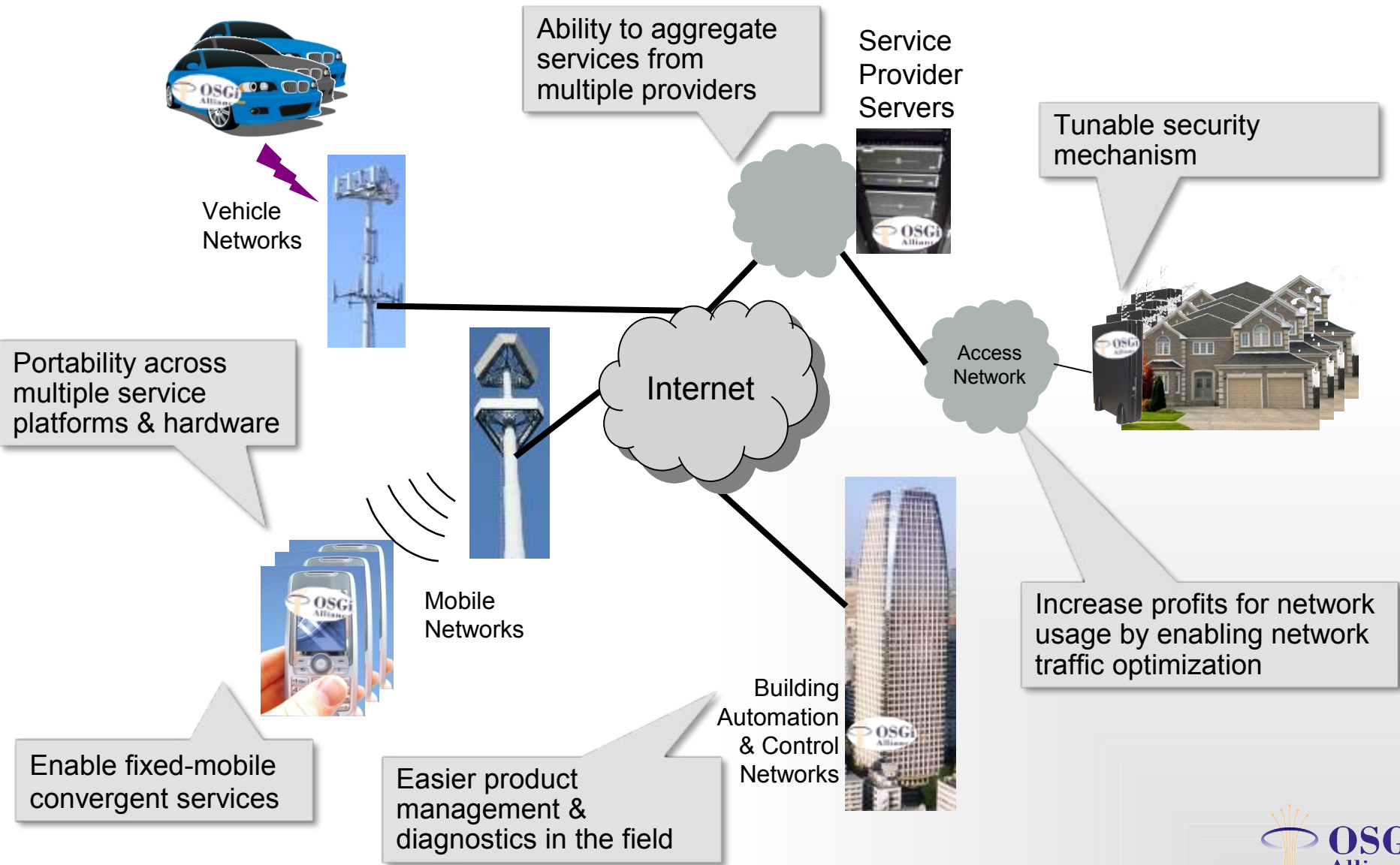
Strategic market differentiation

- Better aftermarket sales and field maintenance through easier service provisioning and aggregation
 - Efficient cooperation with first-tier providers and after-market sales
 - Product configuration at any point in time
 - Ability to diagnose, start-stop-update and install/remove services and applications remotely
 - Remote maintenance and dynamic SW update
 - Establish and change configuration parameters for services
 - Low-cost maintenance
 - Enabling remote management with standard remote management protocols
 - Improve ease-of-use for the consumer

Continuous aftermarket sales

Reduced operating costs

Enabling Seamless Service Availability Across Networks



System & Network Architects

Different views of “A Service”

- There are different views of a “service” from the perspective of a Telecommunications Operator as compared to within OSGi Specifications
 - A “service” to an Operator is something a service subscriber can pay for, an end-to-end service like “Voice” or “Broadband” or a service feature like “Call-Waiting” or “Call-forwarding”
 - The definition of a “service” within the OSGi Service Platform specification is generally a much smaller thing, it’s the advertised interface to a software Lego® block (in an OSGi bundle), “...A service is a normal Java object [a POJO] that is registered under one or more Java interfaces within the [OSGi] service registry...”
 - Formal OSGi definition: “Service – An object registered with the service registry under one or more interfaces together with properties. This object can be discovered and used by bundles.”
- **On the following slides we will use the term “service” as defined within OSGi Alliance Specifications**

Quick Introduction to OSGi Technology

- It's a module system for Java
 - Includes visibility rules, dependency management and versioning of OSGi “bundles” (modules)
- It's fully dynamic
 - Installing, starting, stopping, updating, uninstalling bundles all done dynamically at runtime
- It's service oriented (always has been...)
 - All services are registered in a service registry, available in the well-understood publish, find and bind pattern, again all done dynamically at runtime

OSGi for Simplicity

- Framework for modularity - avoids classpath JAR file hell
- Package/Use/Deploy/Stop just what you need
- Reduced footprint means reduced TCO
- Reuse modular code “out of the box”
- Simplifies third-party module integration
- Simplifies lifecycle and dependency management
- Manages deployments locally or remotely
- Enables smaller, distributed, and diverse teams
- Extensive tool support to hide complexity

OSGi for Flexibility

- No lock in, many providers of core technology including many open source
- Install, start, stop, update, uninstall bundles, all dynamically at runtime means new features can be added or stopped dynamically for rapid adaptability
- Users can run multiple versions of the same module in the same application
- Users can implement their own bundles to extend product capabilities without breaking existing functionality

OSGi for Stability

- Deploy only what you need, reducing complexity, and improving stability
- Set-up visibility rules, dependency management and versioning at the bundle level for ease of deployment, scalability and serviceability
- Ability to run multiple versions simultaneously to test effects and minimize risk of upgrades
- Enables zero-downtime patch/upgrades
- Service registry provides control and predictability of interactions across modules
- Fully integrated with security architecture

Developers

What Java Developers Tell Us

- “Java [alone] does not make it easy to do modular development, i.e., classpath JAR hell”
- “I need to offer new products and services while driving down my IT operating costs”
- “I want to make it easy for my developers to do what is right, and make it difficult for them to do what is wrong”



Why Use OSGi Technology to Build Operator Applications Today?



**Sized
“Just Right”**



**Fully Dynamic /
Hot Pluggable**



Mature

Benefits from building your own applications with OSGi technology – sized “Just Right”

- Reduced Footprint
 - Reduced footprint leads to improved IT efficiency
 - Reduce operating costs
 - Control product usage
 - Pre OSGi products
 - One-size-fits-all products
 - Add functionality to monolithic stack
 - OSGi based products
 - Tailored module sets to target problem at hand
 - Remove unused functionality
 - Package only needed functionality
 - Deploy only required services
 - Start/stop services on demand



Benefits from building your own applications with OSGi technology – sized “Just Right”

■ Improved Extensibility

- Controlled environment for adding modules
 - Class-loading model protects private resources
 - Replaces single class-path model or proprietary solutions
 - Built-in versioning
 - Run multiple versions of module in same application
 - Service registry to control dynamic interactions among modules
- Simplifies and standardizes third-party module integration
- More predictable interactions between modules via service registry



Benefits from building your own applications with OSGi technology – Fully Dynamic / Hot Pluggable



- Improved Serviceability & Availability
 - Dynamic module management
 - Install, start, stop, update and uninstall bundles
 - Extensive dependency management
 - Fully integrated with security architecture
 - Enables zero downtime patch/upgrade
 - Add functionality on demand
 - Remove services that are no longer needed

Benefits from building your own applications with OSGi technology – Mature

- The OSGi Alliance has issued 5 major releases of its set of Service Platform specifications in 10 years, and has become the de facto choice for building modular Java software
- OSGi based dynamic modularity is the basis of all Eclipse software components, and is incorporated in hundreds of software products shipping today



What's Coming for Developers in the New Releases of the OSGi Service Platform?



**Distributed
Connectivity**



**Increased
Productivity
With Blueprint
Component
Model**



**Java EE
Modularity**

Distributed Connectivity

- Enables an organization to deploy its OSGi application on multiple machines.
- Every part of the application can see and use other local and remote OSGi services in a standardized way. Additional metadata also allows integration.



Increased Developer Productivity with Blueprint Service & Component Model

- Boosts developer productivity
- Uses the Blueprint Service metadata model to make it much easier for developers to implement and consume OSGi services
- Developers now just have to understand the business logic of what they are trying to do, and the new Blueprint Component Model helps to abstract from the underlying OSGi core details
- Developers can now easily link from Blueprint Components to OSGi services and hook them together easily in many different ways



Modular Java EE Mapping

- Developers will have the option to deploy existing WARs as-is in an OSGi container
- The new Modular Java EE Mapping enables developers to use the OSGi programming model AND to pick and choose many QoS features from Java EE environments, including persistence, web apps, transaction manager, DB drivers and security, as OSGi bundles
- Estimated availability: Preview 4Q2009, general availability in 1H2010



Conclusion

- OSGi delivers:
 - Applications and infrastructure that are always fit-for-purpose without excess
 - 10 years of maturation of one of the first true SOA technologies in the market
 - The 7/24 capabilities that can be provided by the industry's only standardized fully dynamic / hot-pluggable component system

Getting Started with OSGi

- About OSGi Alliance and its technology
 - <http://www.osgi.org/About/HowOSGi>
- Learn more about OSGi technology now
 - <http://www.osgi.org/Links/BasicEducation>
- And the people driving the revolution
 - <http://www.osgi.org/REG/HomePage>
 - And join in
 - <http://www.osgi.org/About/Join>

Any Questions?

OSGi Alliance Contact:

OSGi Alliance
Bishop Ranch 6
2400 Camino Ramon, Ste 375
San Ramon, CA 94583
Phone: +1.925.275.6625
FAX: +1.925.886.3696

web: www.osgi.org

OSGi is a trademark or registered trademark of the OSGi Alliance in the United States, other countries, or both. Java is a trademark or registered trademark of Sun Microsystems, Inc., in the United States and other countries. All other product or service names are the property of their respective owners.