Enterprise OSGi –
How to tackle the problems of large scale applications in OSGi

Nicole Wengatz, Siemens AG
Tim Diekmann, Siemens Communications, Inc.
Manfred Hutt, Siemens Enterprise Communications GmbH & Co KG

Agenda

1. Where and why do we use OSGi for our enterprise applications?
2. OSGi R3 is a good start, but has shortcomings in our application space.
3. OSGi R4 delivers more, but there is always room for improvement.
4. Still some missing parts, let's join the OSGi Enterprise Expert Group
5. What are we planning to do next...

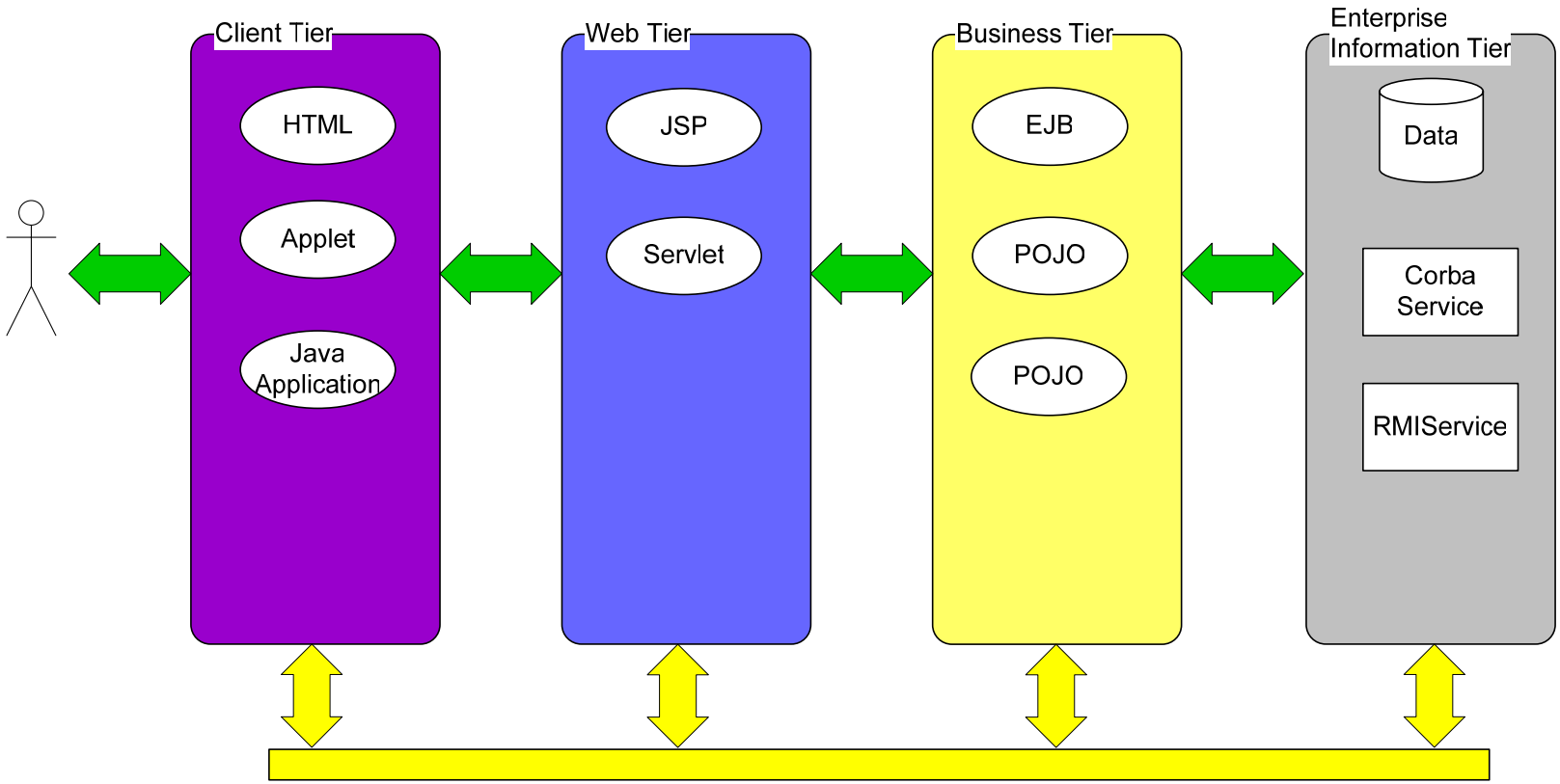
Agenda

1. Where and why do we use OSGi for our enterprise applications?
2. OSGi R3 is a good start, but has shortcomings in our application space.
3. OSGi R4 delivers more, but there is always room for improvement.
4. Still some missing parts, let's join the OSGi Enterprise Expert Group
5. What are we planning to do next...

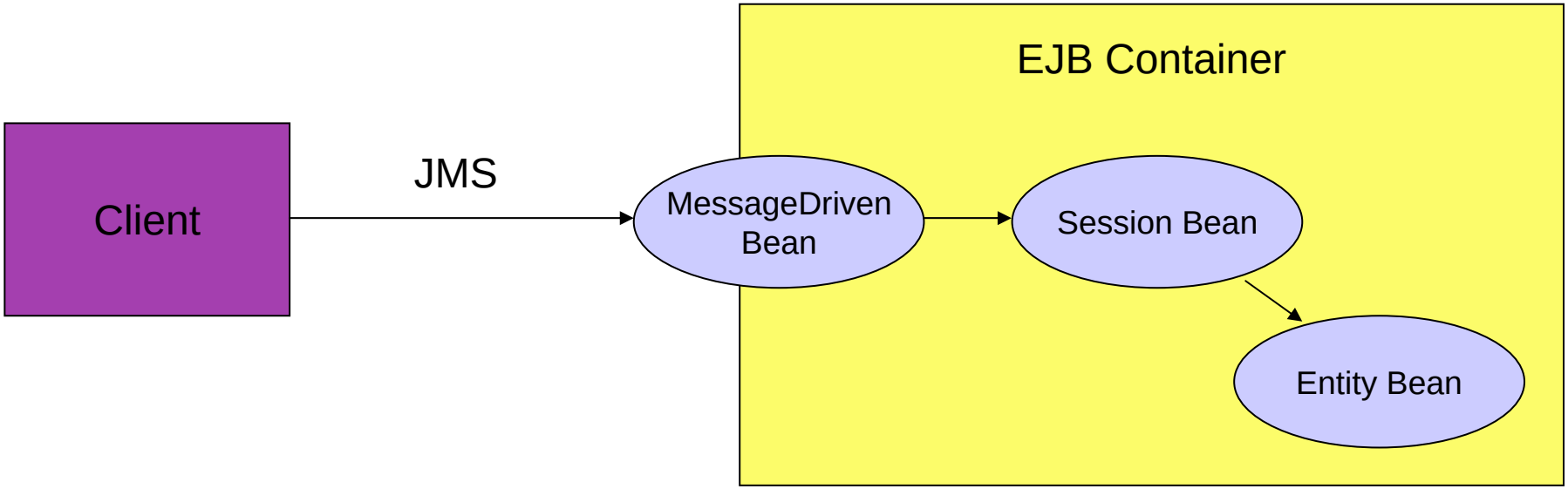
Siemens OpenSOA Application Product Line

- A Product line for soft real-time applications in the unified communications market
  - Enables Product composition out of existing SW assets (Services)
  - Enables Product integration with other Business Applications & Processes
- Key requirements
  - Reduce time-to-market
  - Maximize re-use of existing portfolio
  - Increase and ensure scalability, availability, reliability
  - Ease integration into existing IT infrastructures
- Key decisions
  - Platform independence
  - Service Oriented Architecture
  - Component Container technology

Java Enterprise World: A Short History of Time



Technology Option: Java EE and JMS



Technology Decision: OSGi

- Why Not EJB Container?
  - JMS based request/reply in combination with MessageDrivenBean too heavyweight
  - JMS aimed at traditional business application / integration domains (i.e. guaranteed message delivery)
  - EJB restrictions
  - Message Driven Beans not designed for lightweight events
- Further Evaluation
  - JMX Container
  - OSGi
- Decisions made:
  - Use OSGi as base, enhance OSGi with missing functionality

Agenda

1. Where and why do we use OSGi for our enterprise applications?
2. OSGi R3 is a good start, but has shortcomings in our application space.
3. OSGi R4 delivers more, but there is always room for improvement.
4. Still some missing parts, let's join the OSGi Enterprise Expert Group
5. What are we planning to do next...

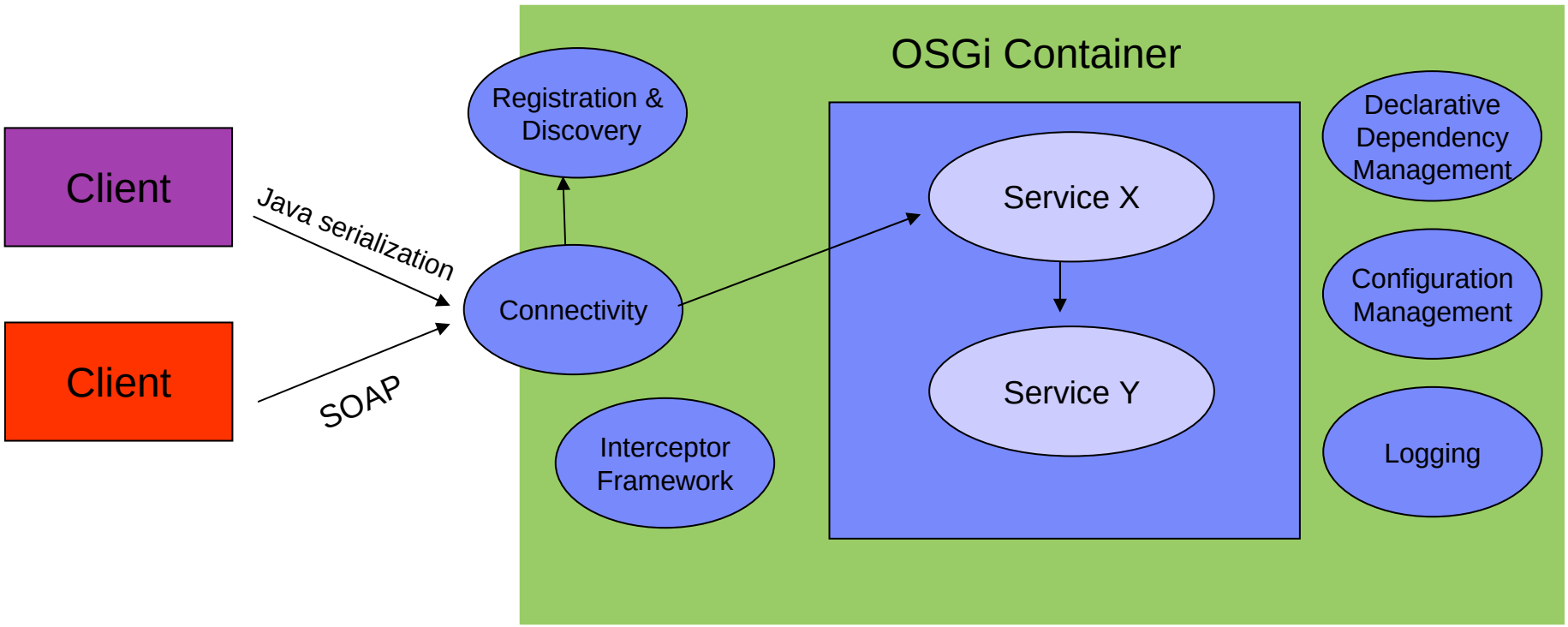
OSGi (R3) container has many advantages

- It's lean and mean, provides us
  - Native support for SOA applications
  - Hosting environment for services with minimal footprint
  - Component model
  - Full lifecycle of services
  - Platform independence, vendor independence
  - Interface based, abstraction from implementation, supports separation of concerns
  - Allows multithreading
  - Provides registry and discovery of available services
  - Tool support, e.g. Eclipse
- Plus much more that we did not use

OSGi R3 shortcomings for our application domain

- Restricted to single container
- Service model limited to OSGi container environment
- The OSGi R3 specification does not address
  - support for multiple communication patterns
  - declarative dependency management
  - support interceptor mechanism, e.g. Spring interceptor framework
  - support for deployment and configuration of non-OSGi artifacts that accompany an enterprise application
  - support for user based authentication & authorization
- Listeners and trackers have to be coded manually

Runtime Environment: Extending the OSGi R3 container



Our solution approach to multi-container

- Enhanced service model
  - local service vs remote service
- Inter container communication support
  - individual remotely addressable instances
- Service registry beyond the border of a single container
  - distributed service registry
  - affinity
  - configurable responsibilities (properties) for individual instances

Our solution approach to inter-container and intra-container communication

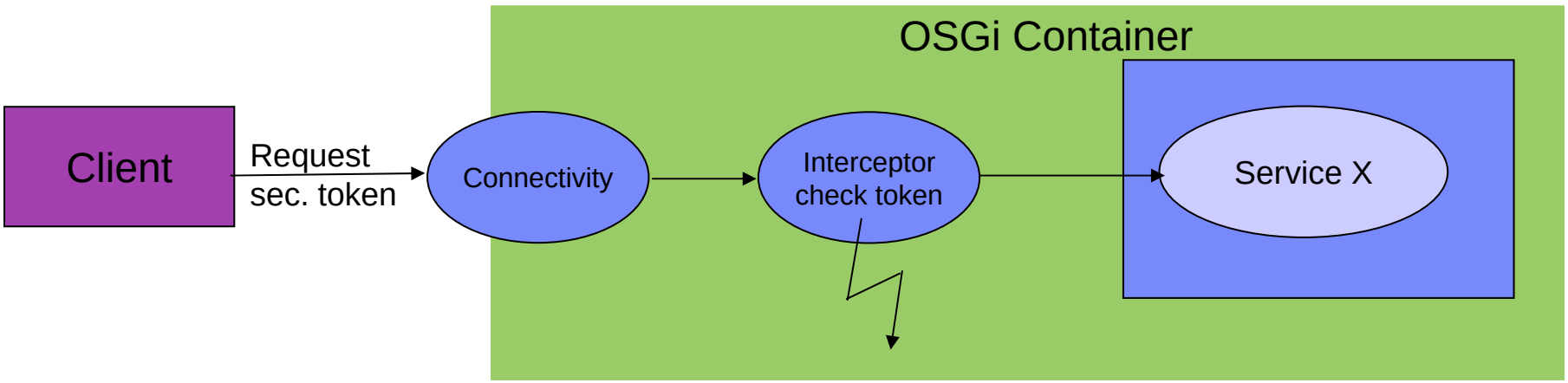
- Integration of service bus (Message oriented middleware – MOM)
  - client to remote service, local service to remote service
  - remote service to remote service in different container
  - local service to local service inside same container
- Support of multiple communication patterns
  - request – reply
  - request – multiple reply
  - event based – publish/subscribe
- Support of multiple communication protocols
  - JAVA serialization over plain TCP/IP sockets
  - JMS (for events)
  - HTTP(S)
  - SOAP over HTTP(S)

Our solution approach to declarative dependency management

- Add dependency manager to each component and service
- Add Deployment Descriptor to every bundle
  - XML file with defined schema
  - describing dependencies to other components or services
  - provided interfaces
  - interceptors
- Register interfaces as OSGi service (component in our terms) or service, which can be reached from outside
- Use inversion of control pattern for injection of dependencies

Our solution approach to security

- Support of user based authentication & authorization
- Use of Spring AOP interception for enforcement
  - Authentication interceptors added declaratively to service
- Support of resource based security
  - e.g. access control lists



What we have reached so far

- Platform independent base for our product line
  - OSGi gives us the base for free
- Scalability
  - Use multiple containers and load balancers
  - Communication hides the target location, client needs not to be aware of it
- Availability
  - Distributing services allows for different failover scenarios

Agenda

1. Where and why do we use OSGi for our enterprise applications?
2. OSGi R3 is a good start, but has shortcomings in our application space.
3. OSGi R4 delivers more, but there is always room for improvement.
4. Still some missing parts, let's join the OSGi Enterprise Expert Group
5. What are we planning to do next...

OSGi R4 came with improvements

- Declarative Services (DS)
- Deployment Admin Service
- Configuration Admin Service (already available in R3, but only introduced in our project with R4 container)
- Improved tool chain, e.g. Eclipse PDE

Shortcomings of OSGi R4

- DS is not flexible and powerful enough for enterprise requirements:
  - Semantics in the spec do not apply to our problem space, e.g. restart of services in case of configuration changes or disposal of stateful services if required dependency went down and no suitable instance is available.
  - Support for POJO dependency injection and interceptors still missing.
  - Interaction with Configuration Admin Service not well defined.
- Still no support for multi-container deployment
- No answer to scalability and availability of services
- We still miss a differentiation between services which are remotely accessible and services which are only locally accessible.

What did we take from OSGi R4

- Take ideas of Declarative Services and adapt to our needs
  - Enterprise Declarative Services (EDS)
- Use CAS
  - Enhanced integration with EDS
- Deployment Admin Service
  - Needs to be enhanced to support of multiple versions of same bundle
- PDE tool chain enhanced by additional tools

Our solution approach to virtualization of services

- Container hierarchy in a single system across multiple nodes
  - containers host services (local and remote)
  - nodes host containers (and other non-OSGi processes, e.g. web container)
  - system addresses all nodes
- Central configuration management for all containers
  - system, node, container management
- Single registry system wide, service discovery mechanism
  - distributed remote service registry
  - every remote service becomes available to any other service and to external clients
- Multiple services instances on multiple containers provide for increased reliability, availability, and scalability
- Abstraction of hosting location
  - client is interested in service based on interface contract, not in implementation

Agenda

1. Where and why do we use OSGi for our enterprise applications?
2. OSGi R3 is a good start, but has shortcomings in our application space.
3. OSGi R4 delivers more, but there is always room for improvement.
4. Still some missing parts, let's join the OSGi Enterprise Expert Group
5. What are we planning to do next...

Enterprise Expert Group (EEG)

- Other companies ran into same issues
- All our solutions are proprietary and non-interoperable
- Standardization of solutions enables integration with other vendors
  - supports product and solution business
  - enables partnerships with other vendors
- Huge interest demonstrated by other companies to help driving changes in OSGi

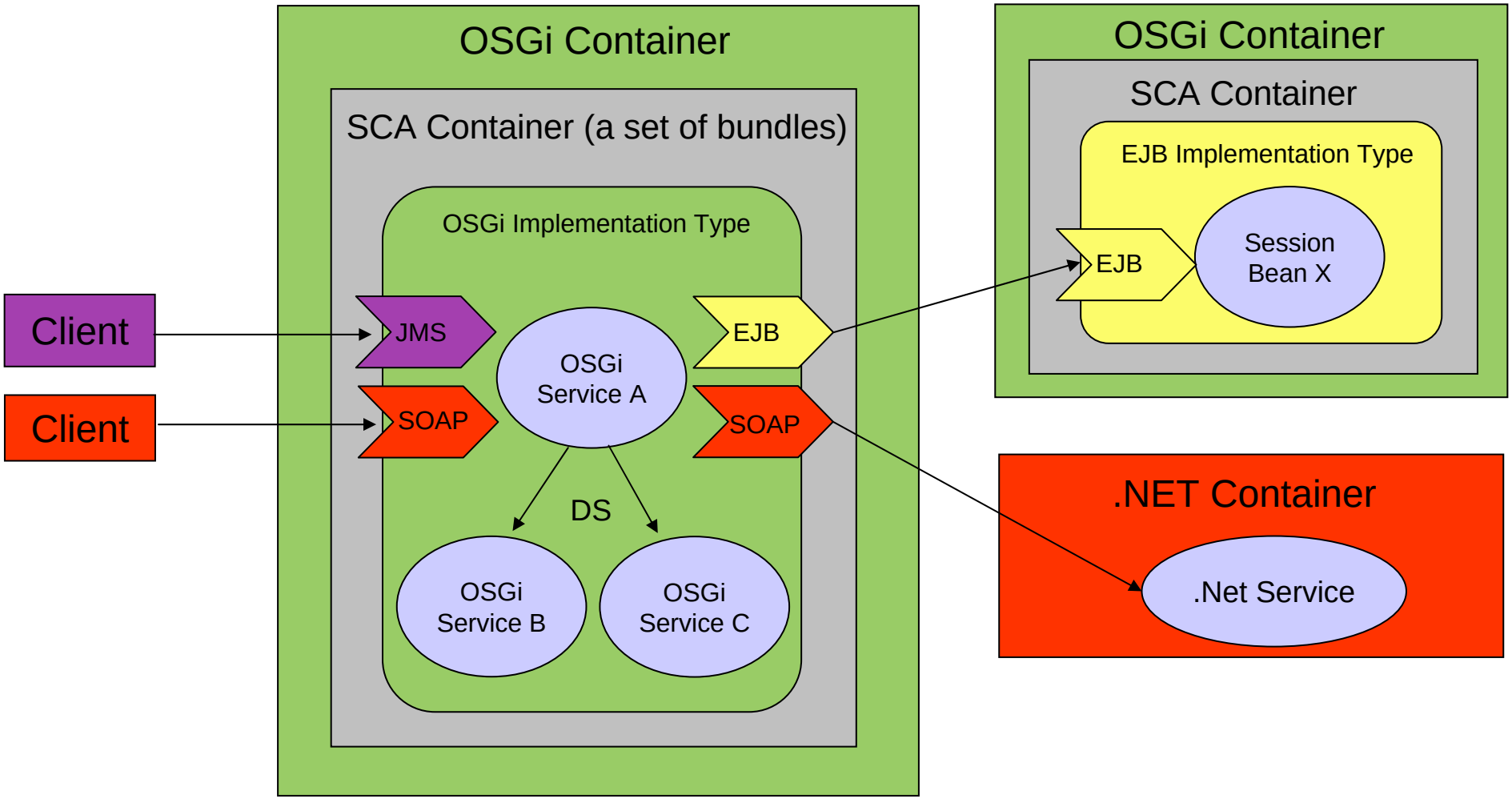
Agenda

1. Where and why do we use OSGi for our enterprise applications?
2. OSGi R3 is a good start, but has shortcomings in our application space.
3. OSGi R4 delivers more, but there is always room for improvement.
4. Still some missing parts, let's join the OSGi Enterprise Expert Group
5. What are we planning to do next...

Next steps

- Drive the standardization of enterprise specific solutions.
- Example:
  - We have a “Home build” Communication Framework which is integrated via a proprietary way in our OSGi service container.
  - Our Goal is to replace the Communication Framework in the **mid-term** with off-the-shelf middleware and to move into the direction of an OSGi / SCA (Service Component Architecture) compliant communication middleware.

Proposal: Use SCA for distributed communication (1)



Proposal: Use SCA for distributed communication (2)

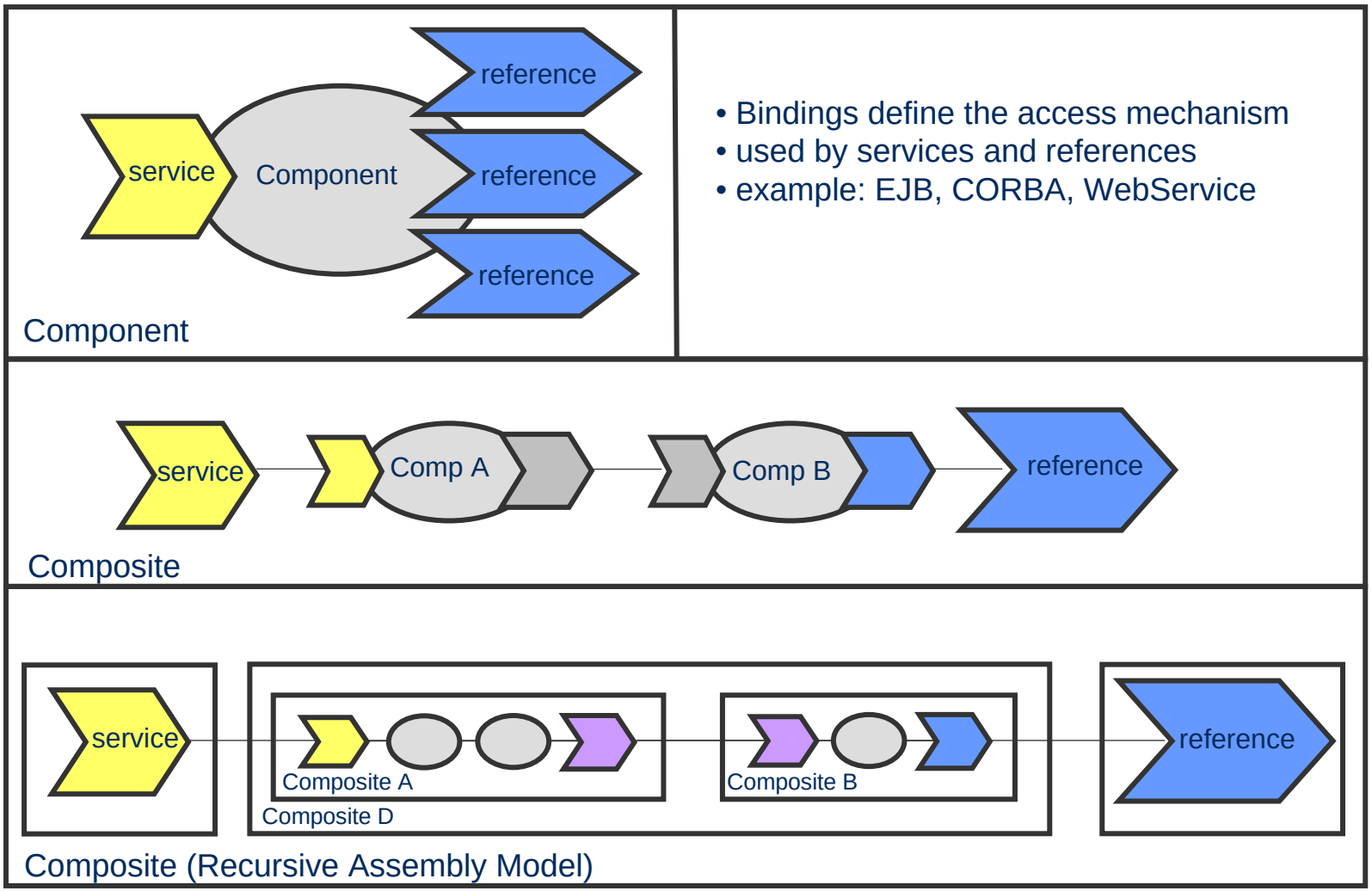
- OSGi container hosts SCA container, SCA container is implemented as set of OSGi bundles.
- OSGi bundles contain in addition to the business logic the SCA composite file which contains the declarative configuration for:
  - SCA service bindings (via which protocol the OSGi service is accessible) and
  - SCA reference bindings (via which protocol the OSGi service is going to access services running in other containers).
- For dependencies inside an OSGi container the OSGi R4 Declarative Services will be used.

Conclusions

- We started with EJB, but OSGi is better suited for most of our requirements.
- Our experiences with OSGi are very good.
- To fulfill the enterprise requirements some parts are still missing. Our goal is to define standard solutions for the missing parts in Enterprise Expert Group.
- Integration is a big issue inside Siemens (not only for the Siemens OpenSOA project). The power combination “OSGi and SCA” allows to use always the best suited technology and to integrate easily in heterogeneous environments.
- (Enterprise) OSGi is cool ☺

Backup

Possible *Solution*: OSGi and SCA combined



Java Enterprise World:
Always use the best suited technology