



# **OSGi Alliance Community Event**

## **Creating Enterprise Services for the Siemens OpenSOA Product Line**

Nicole Wengatz, Siemens AG

Manfred Hutt, Siemens Enterprise Communications GmbH & Co KG



## Agenda

- Brief Overview of Siemens OpenSOA
- OpenSOA ToolChain
- Demo
- Conclusions and Next Steps

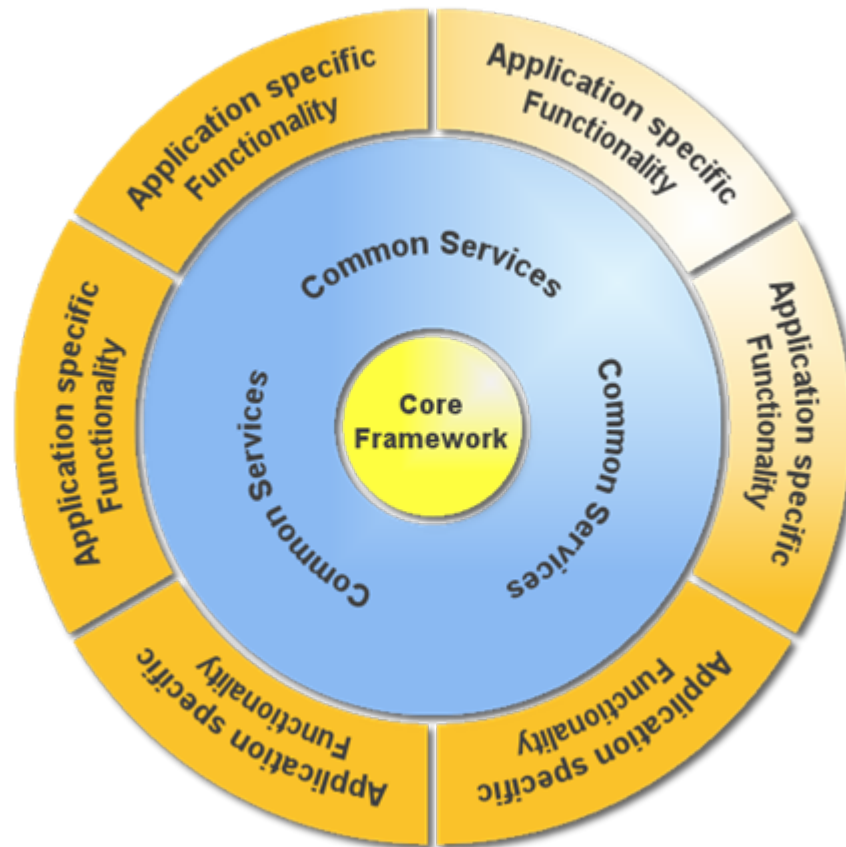


## Siemens OpenSOA Application Product Line

- 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
  - Simplify integration into existing IT infrastructures
- Key decisions
  - Platform independence
  - Service Oriented Architecture
  - Component Container technology



## Siemens OpenSOA Application Product Line

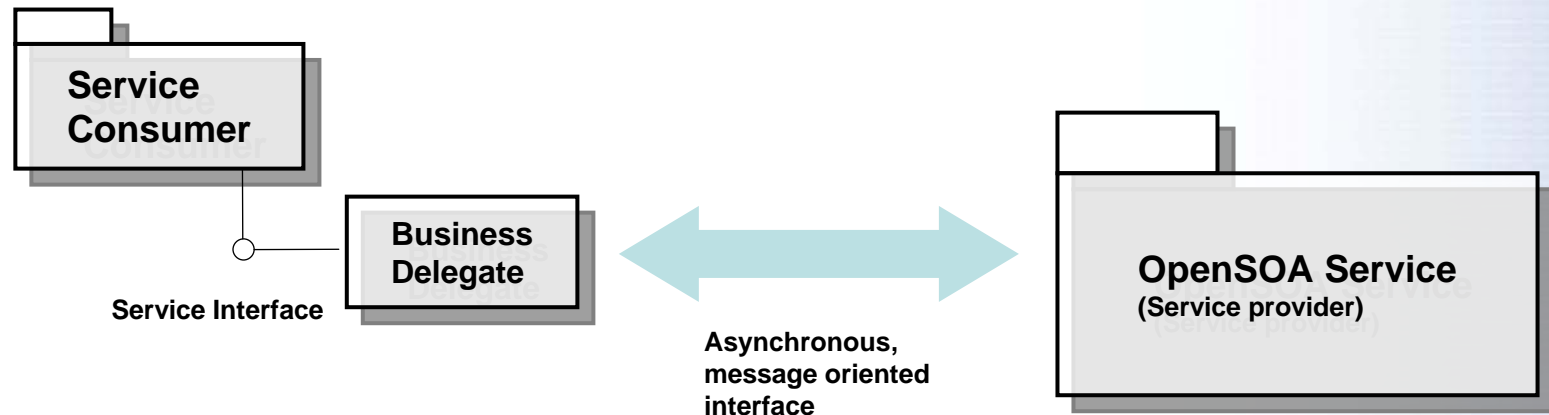


- Reusable software assets, that allow better and easier integration into IT-environment and customers business processes
- Harmonized SDK & API strategy for fast response to customer requirements and flexibility to build vertical solutions
- Common user interfaces



## OpenSOA Service

- An OpenSOA Service provides a functionality exposed through a well defined interface (contract).
- OpenSOA Services communicate through asynchronous message exchanges.
- OpenSOA Services can be accessed via different protocols (http(s), plain sockets (optional ssl)) using different message encodings (Java serialization, SOAP).





## Extending the OSGi Service Model

- OpenSOA service is an OSGi service that
  - is hosted in an OSGi container (the usual OSGi model)
  - is remotely accessible (registration and discovery)
  - provides message based, asynchronous interfaces.
  - has an extended lifecycle model
  - has an extended dependency model
  - allows to intercept service invocations.
- OpenSOA component is an (extended) OSGi service that
  - is accessible in a local container context only
  - provides synchronous method invocation interfaces.

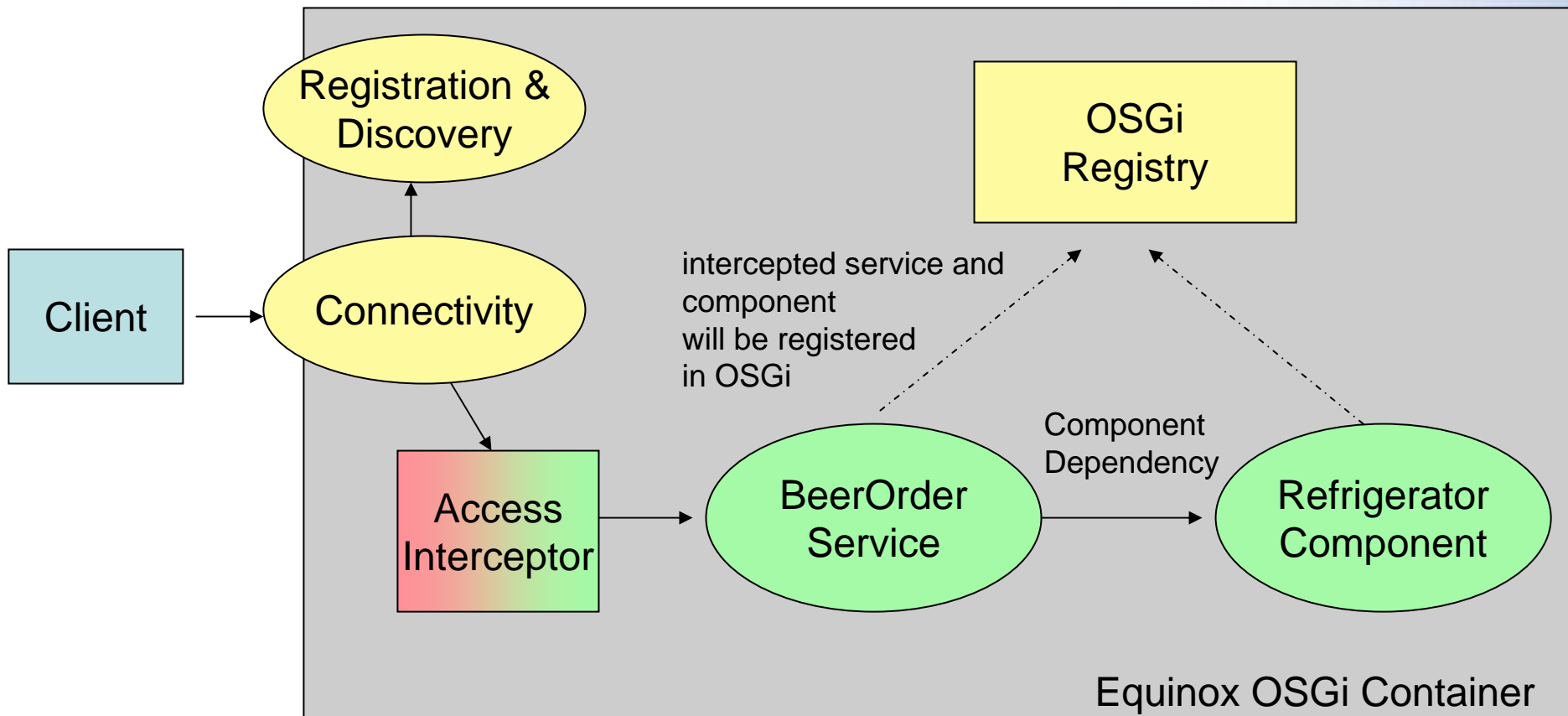


## OpenSOA Toolchain

- Project Generator: allows you to create the following types of projects:
  - service
  - component
  - library
- SIDL Contract Modeling
- Dependency Selection
- Code Generation

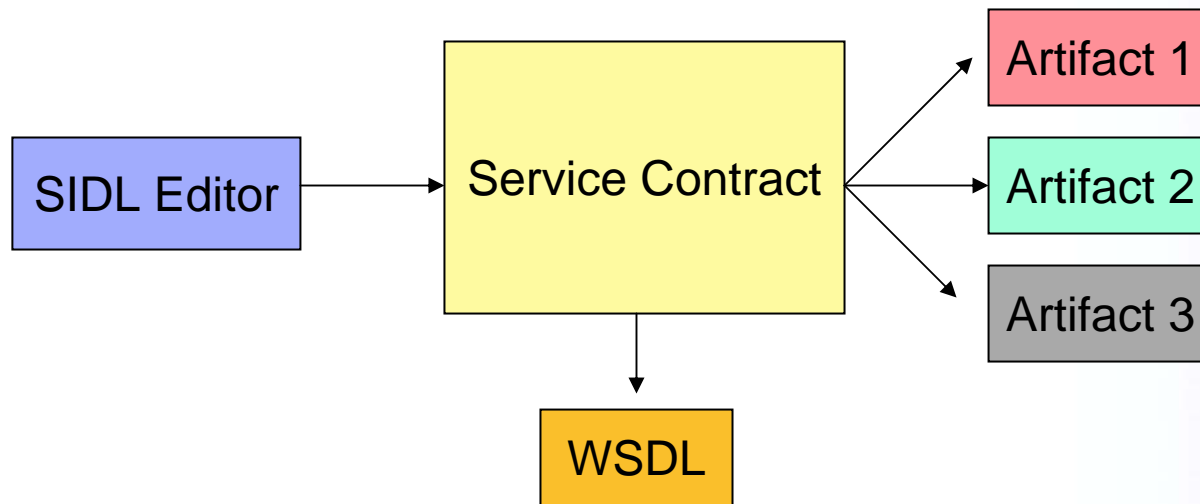


## Demo





# SIDL Contract Modeling and Code Generation





-> Demo 😊

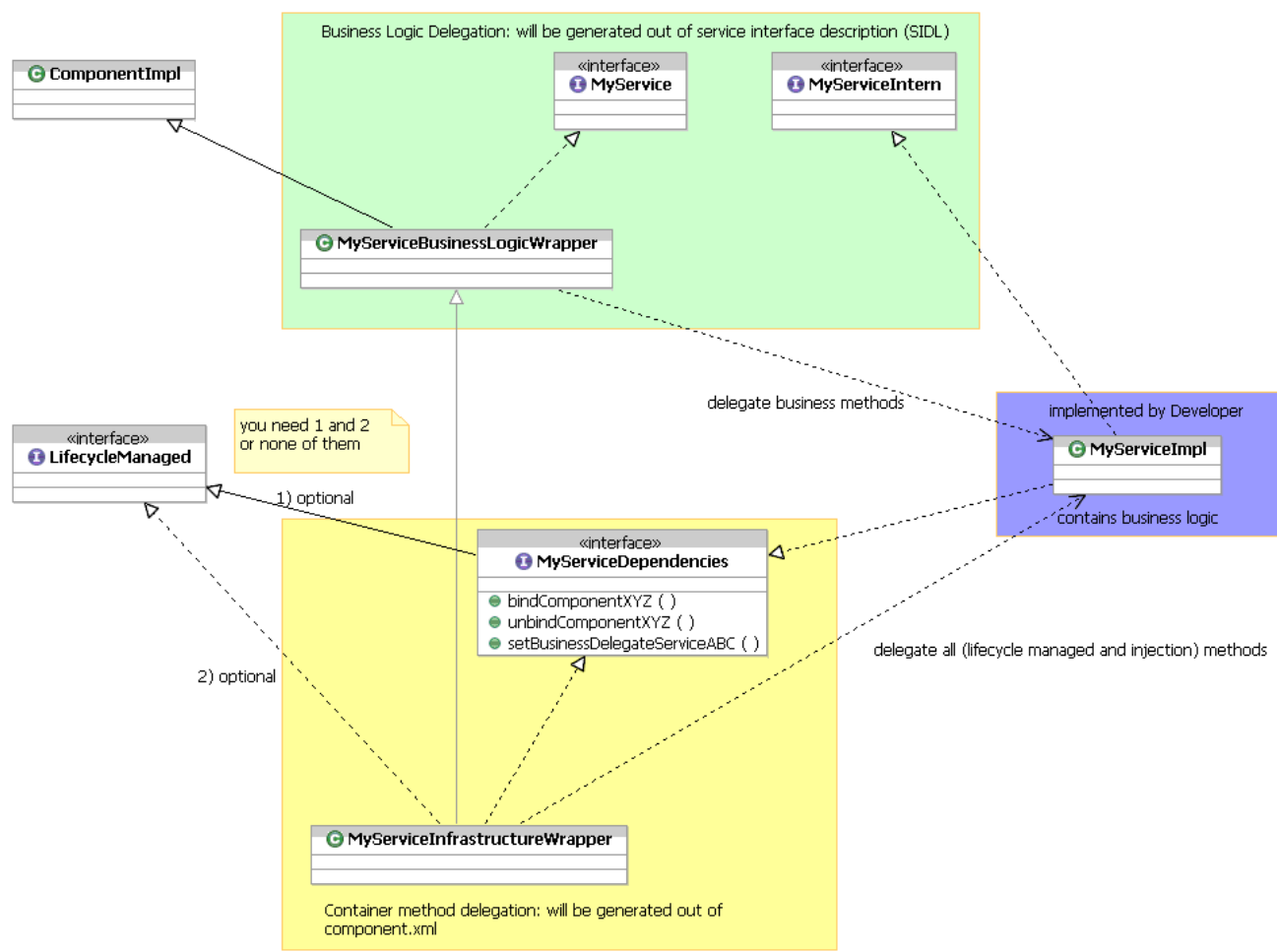


## Dependency Selection and Generation

- The EDS (Enterprise Declarative Service, extension of the OSGi R4 DS) is responsible for the lifecycle and dependency management of the OpenSOA services and components. It provides:
  - Lifecycle Management e.g. start, stop, activate and deactivate component instances
  - Dependency management (e.g. Service A requires Component B)
  - Creation and injection of business delegates and object references
  - Creation of interceptor chain
  - Injection of configuration parameters



# Code Generation





## Demo

- ToolChain
- Service with Component Dependency
- Interceptor
- Distributed OSGi
- Test



-> Demo 😊



## Conclusions and Next Steps

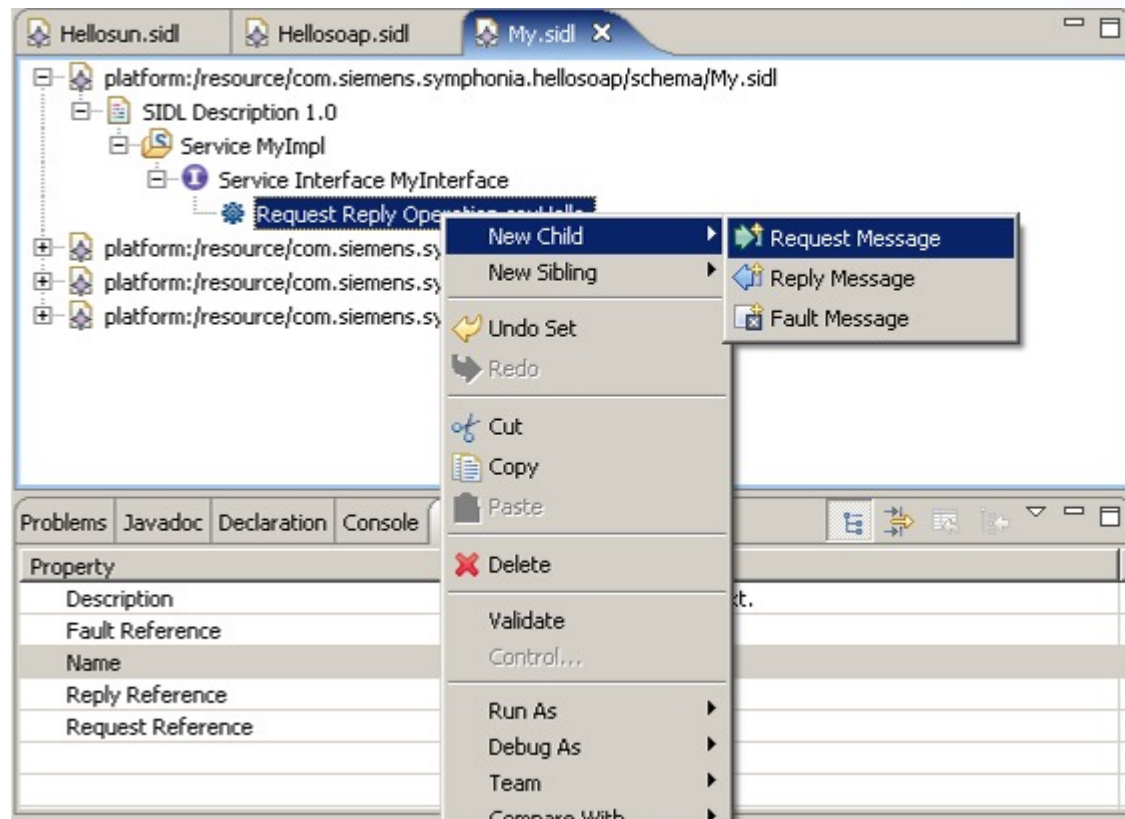
- Our experiences with OSGi are very good.
- To fulfill the enterprise requirements some parts are still missing.
- Our goal is to define standard solutions for the missing parts in Enterprise Expert Group.
  - Distributed OSGi
  - Component Model, EDS vs. Spring-OSGi
- Open-Source ToolChain



# Backup

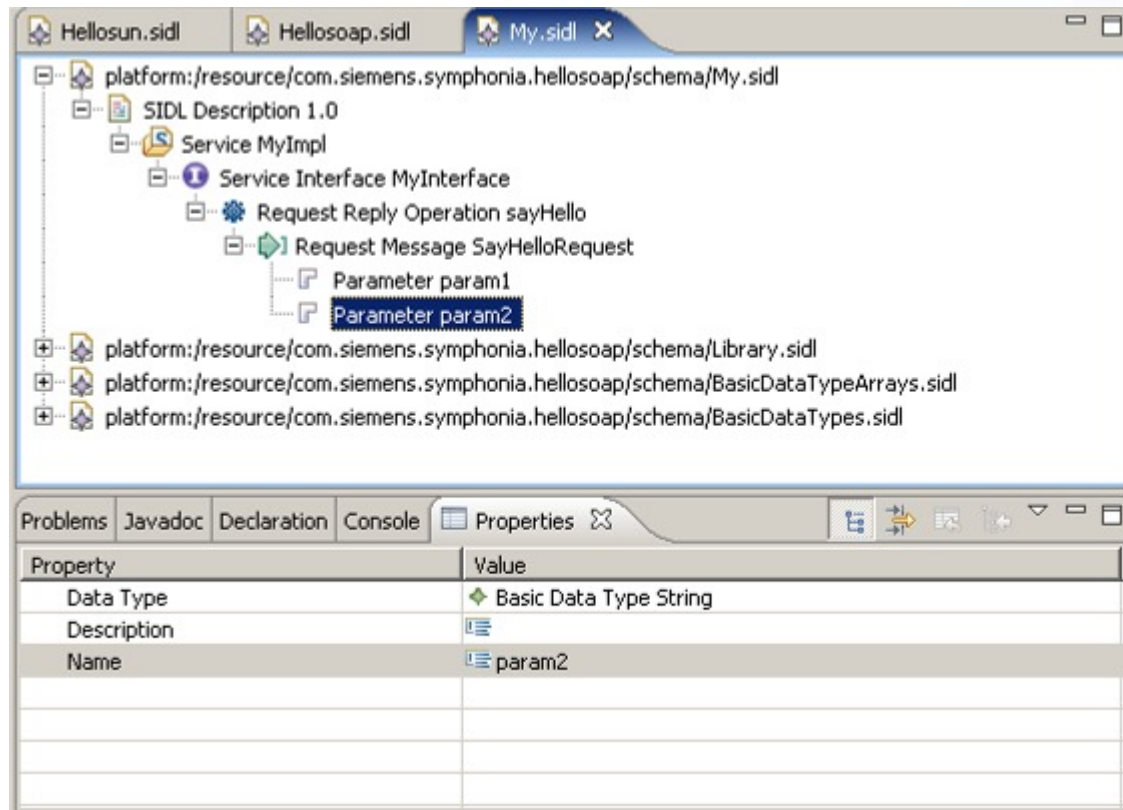


## SIDL Contract Modeling





## SIDL Contract Modeling



The screenshot shows an IDE window with three tabs: 'HelloSun.sidl', 'HelloSoap.sidl', and 'My.sidl'. The 'My.sidl' tab is active and displays a tree view of the contract model. The tree structure is as follows:

- platform:/resource/com.siemens.symphonia.hellosoap/schema/My.sidl
  - SIDL Description 1.0
    - Service MyImpl
      - Service Interface MyInterface
        - Request Reply Operation sayHello
          - Request Message SayHelloRequest
            - Parameter param1
            - Parameter param2

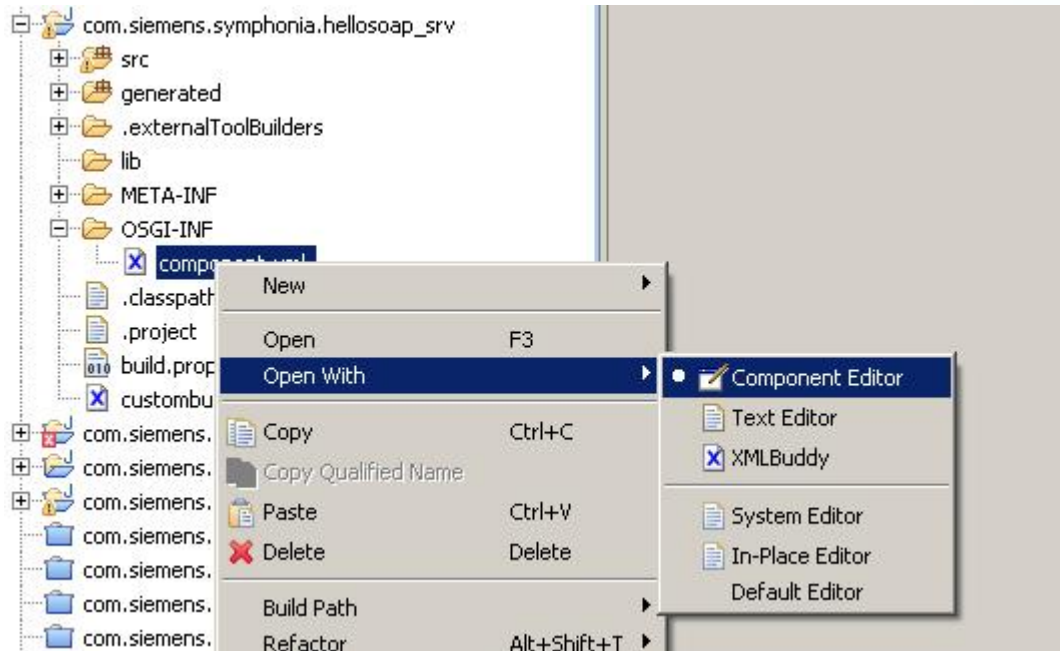
- platform:/resource/com.siemens.symphonia.hellosoap/schema/Library.sidl
- platform:/resource/com.siemens.symphonia.hellosoap/schema/BasicDataTypesArrays.sidl
- platform:/resource/com.siemens.symphonia.hellosoap/schema/BasicDataTypes.sidl

Below the tree view is a 'Properties' panel with a table showing the properties of the selected 'Parameter param2'.

Property	Value
Data Type	Basic Data Type String
Description	
Name	param2



## Component Editor





# Add Reference

\*com.siemens.symphonia.duckservice\_srv

## References

**References**

Name	Interface	Injection	Cardinality	Responsibilities	
com.siemens.symphonia.duck_comp	com.siemens.symphonia.duck.Duck	Interface	0..1		Add...
com.siemens.symphonia.restaurant_comp	com.siemens.symphonia.restaurant.Restaurant	ServiceRef...	0..1		Edit...
					Remove

**Add Reference**

Name:

Interface:

Injection:

Cardinality:

Responsibility

Name	Value	
eat	duck	

Add...  
Remove

OK Cancel

Names\_Provides | References | ObjectReferences | Interceptors | BD-References | component.xml