Integrating Object Teams and OSGi: Joint Efforts for Superior Modularity

Stephan Herrmann
Technische Universität Berlin

©2007 Stephan Herrmann; made available under the EPL v1.0
Modularity by ObjectTeams/Java:

- **Collaboration modules** ⇒ “team”
  - modules larger than classes
  - instantiable, inheritable
  - nestable

- **Role classes & objects** ⇒ “playedBy”
  - selective interface to base
  - method call interception
    - adaptation à la Aspect-Oriented Programming

Bring advances in programming languages to the Equinox world of components
• **team**
  collaboration module

• **role**
  members of a team

• **playedBy**
  connect role to base

• **callout**
  forward to base

• **callin**
  intercept base method

• **decapsulation**
  break base encapsulation

• **(de)activation**
  dis/enable all callins

---

**OT/J based Architecture**

```
T MyTeam
```

```
| teamField: someType |
| teamMethod(T2): Type2 |
```

```
Role1
```

```
| Role2 |
| roleMeth1() |
| roleMeth2() |
| roleMeth1 -> method1 |
| roleMeth2 <- method2 |
```

```
T SubTeam
```

```
| otherField: otherType |
| otherMethod(T3): Type3 |
```

```
BasePkg
```

```
C1
```

```
C2
```

```
Method1() |
```

```
Method2() |
```

```
«playedBy»
```

```
«playedBy»
```

```
callout
```

```
callin
```

```
SubTeam
```

```
MyTeam
```

```
0..1
```

---

4/10/07
Integration

Object Teams + Equinox

OT/Equinox
Architecture with OT/Equinox

Require-Bundle: B

MANIFEST.MF

Aspect binding = architecture level relationship
Cake & Icing & Eat It!

• OT/Equinox combines the benefits of
  - OSGi, Equinox: deployable components
clean architecture
  - ObjectTeams/Java: teams and roles and playedBy
variants, unanticipated adaptation
adaptation is a first-class module

• OT/Equinox is **not** a compromise
  - aspects become part of the architecture

• Real-world use: self application
  - demonstrates new options for re-use

*www.objectteams.org*