Extending the JDT

- Can be fun, if
  - API exposes what you need to see
  - Extension points exist where you want to adapt
- JDT offers a wealth of API and extension points
- Not every RFE can create new API
  - Conflicts with other clients
  - Performance impact
  - Maintenance costs
- What if your RFE gets rejected?
  - **Abandon your project?**
  - **Copy&Paste**
  - **Use Object Teams**

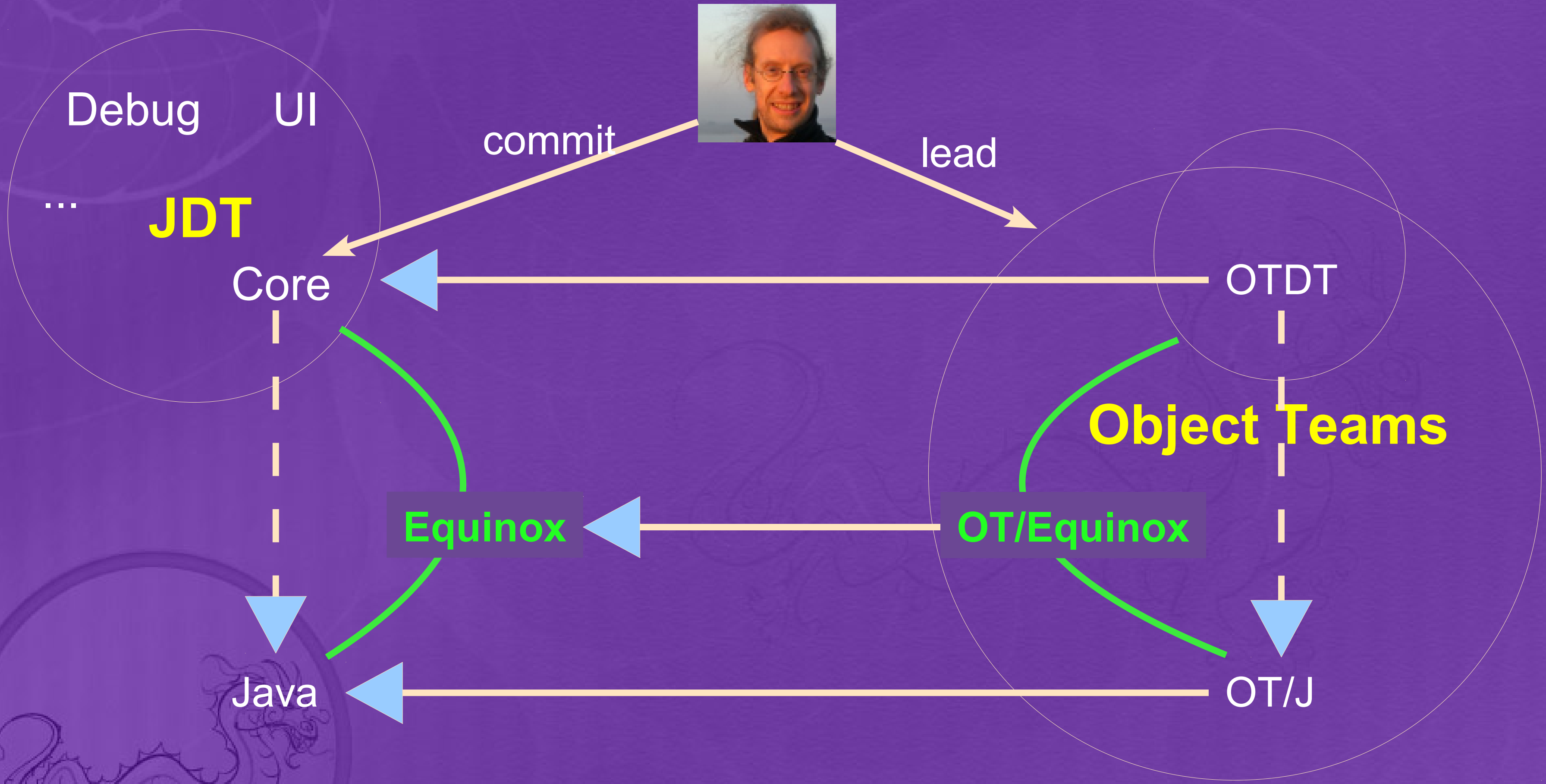
eclipseCON 2012
Reston, Virginia
March 26 - 29

The JDT Dragon meets Object Teams

Ayushman Jain
IBM Bangalore, India
ayushman.jain@in.ibm.com

Stephan Herrmann
GK Software AG, Berlin
stephan@cs.tu-berlin.de

Relation between two Projects



Cheat with Style

- **What if API is missing?**
  - Ignore restrictions, even private
- **What if extension point is missing?**
  - We have other means for extending:
    - specialize instances not classes
- **What about maintainability?**
  - Do minimal harm
  - Group extensions to higher-level modules: **teams**.

Exercise Stopwatch

- New → Example → Stop Watch Example
- src / ... / Main.java
  → Run As → Java Application

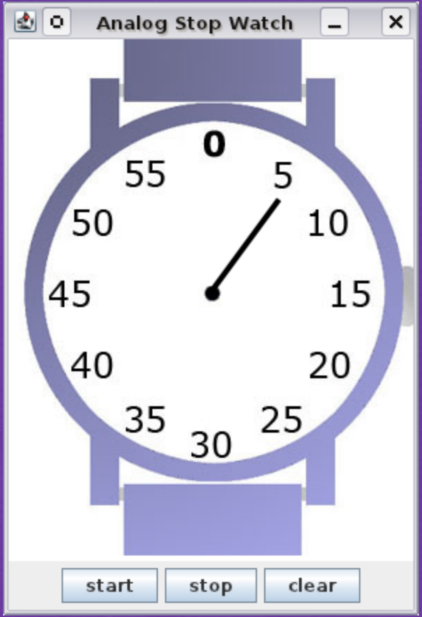
Note: to stop application click ■ (Console view)

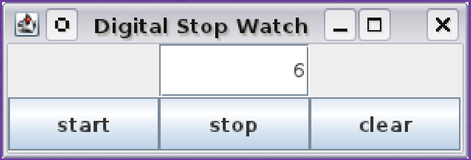
Find role class WatchUI.WatchDisplay:

- When are instances of this role created ?
- When is its method update() called?

Extra (Demo):

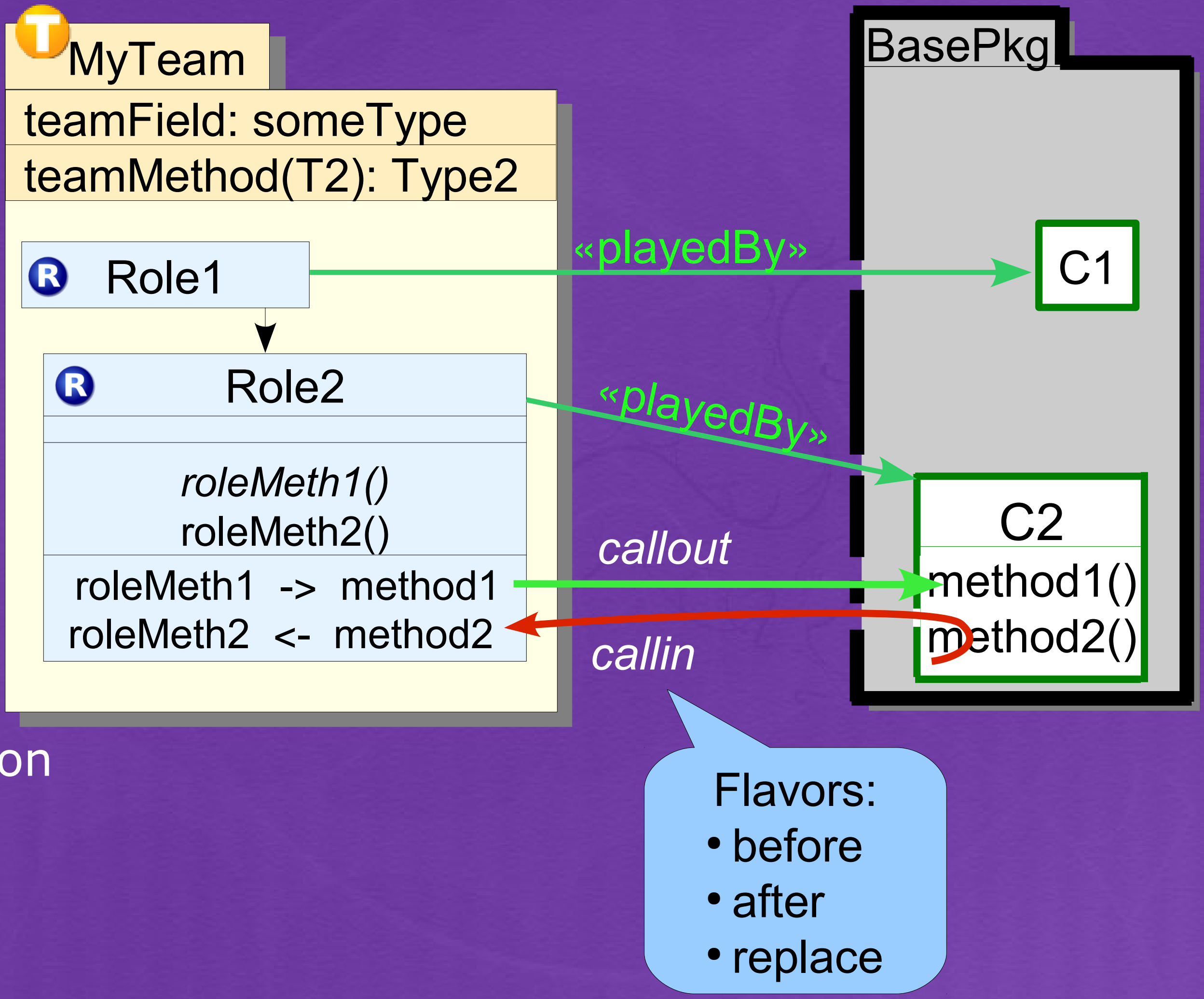
- Terminate application when watch is reset at 3 seconds.



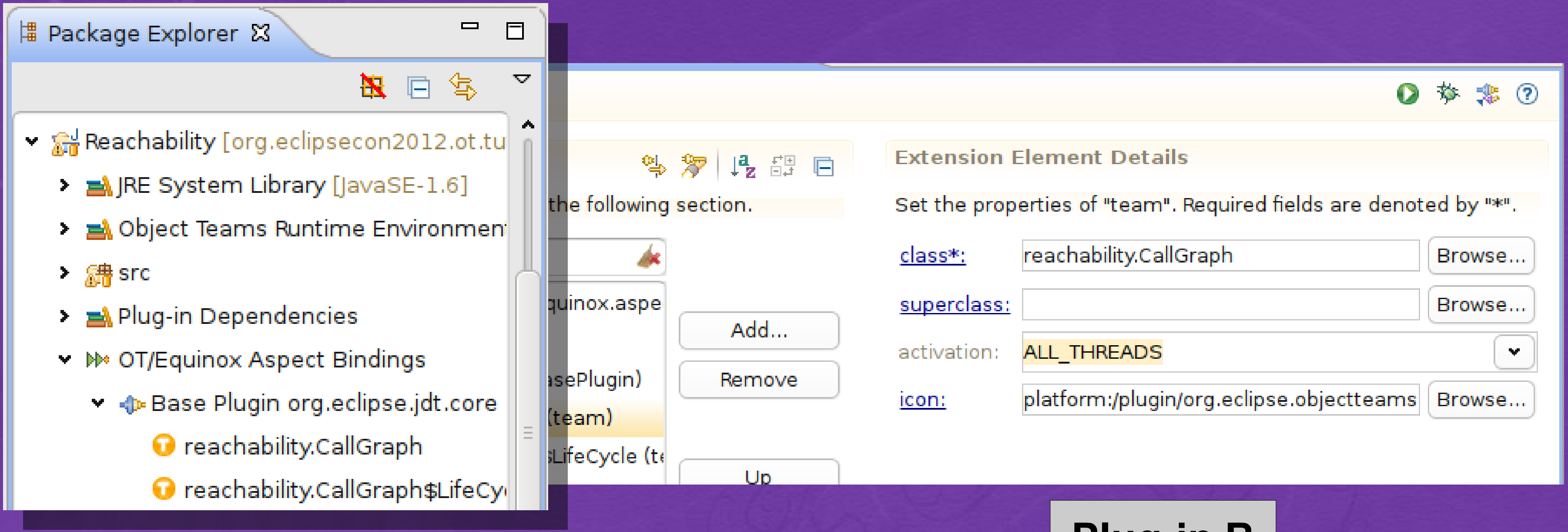


Object Teams in a nutshell

- **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



OT/Equinox


Package Explorer

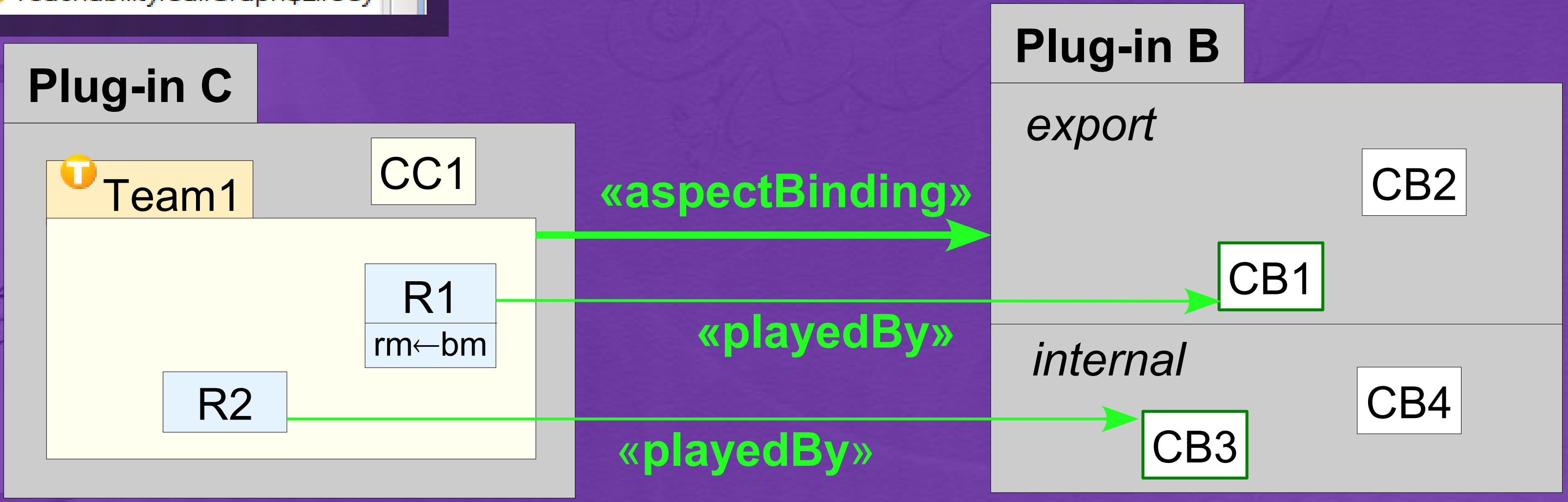
Reachability [org.eclipsecon2012.ot.tu
JRE System Library [JavaSE-1.6]
Object Teams Runtime Environmen
src
Plug-in Dependencies
OT/Equinox Aspect Bindings
Base Plugin org.eclipse.jdt.core
reachability.CallGraph
reachability.CallGraph$LifeCy

the following section.
quinox.aspe
asePlugin)
(team)
LifeCycle (te

Extension Element Details

Set the properties of "team". Required fields are denoted by "*".

class*:	reachability.CallGraph	Browse...
superclass:		Browse...
activation:	ALL_THREADS	
icon:	platform:/plugin/org.eclipse.objectteams	Browse...



Exercise: AntiDemo Plug-in

- Write a new Plug-in
  - adapting org.eclipse.jdt.core
- Change the Java naming rules
  - Class names cannot start with “Foo”
  - Hint: `JavaConventions#validateXZY()`
- In a runtime workbench
  - Try to create a class “Foobar”
  - Be inventive!

Note: you may need to scroll to see
☑ Enable OT/Equinox

Demo: Reachability Analysis

Implement a Plug-in that finds unreachable code.

- All “main” methods are considered reachable.
- All methods called from a reachable method are also reachable.
  - Method calls inside dead code should not be considered. (e.g. “if (false) m();”)
- Analyzing method calls must consider polymorphism /overriding.
- Methods that are only called from unreachable methods are not reachable (including “islands”: cycles of unreachable methods).

Need a whole-system call graph.
+ local flow analysis
+ inheritance information

Design

Piggy-back on the JDT compiler

- Find all MethodDeclarations during resolve
  - Record start nodes, like “main” methods
- Create a graph of MethodBindings
  - Connect nodes when analysing MessageSends
  - Ignore calls in dead code
- Start from set of all methods
  - subtract all methods reachable from a start node
  - consider method overriding
- Only work during full build
  - report remaining methods after subtracting

No Limits – No Pain

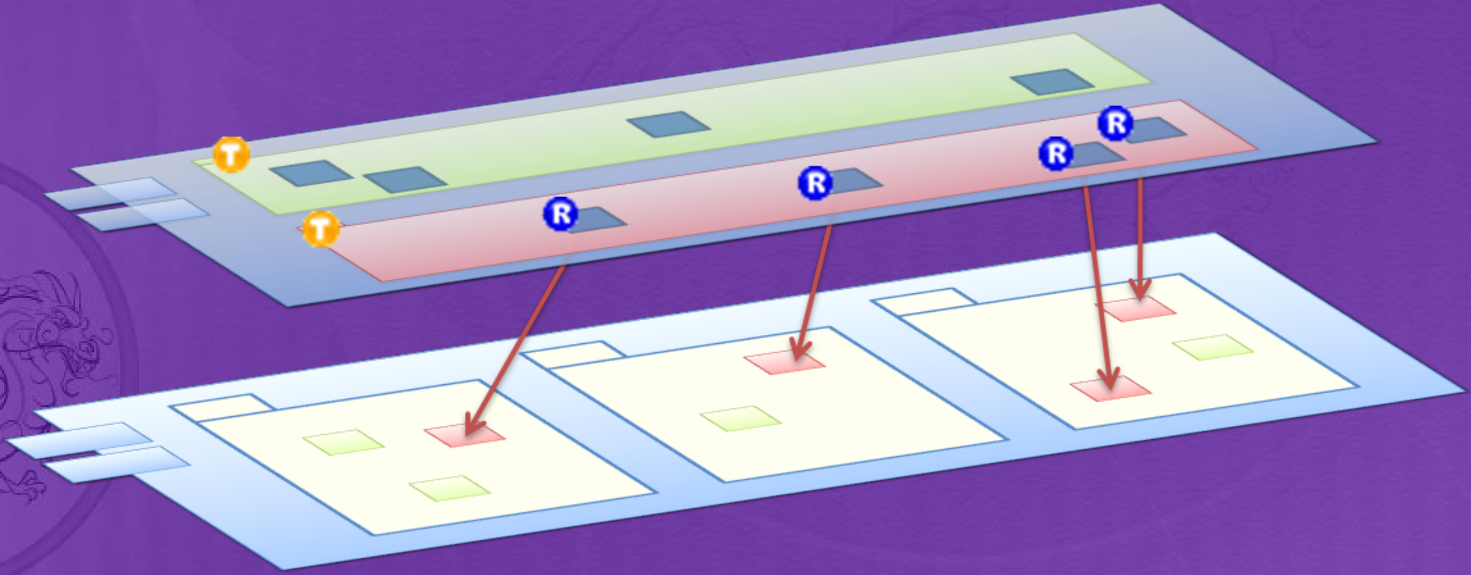
- **playedBy**: every object is extensible
  - **callout**: every method / field can be made accessible
  - **callin**: every method is overridable
- warnings for decapsulation: proceed at your own (low) risk
- interface (bidirectional) fully explicit

- lean & mean = powerful & maintainable

Extending the JDT into new Dimensions

Each feature is a module / team

- Define suitable structure using teams & roles
- Create connections using playedBy, callout & callin



Summary

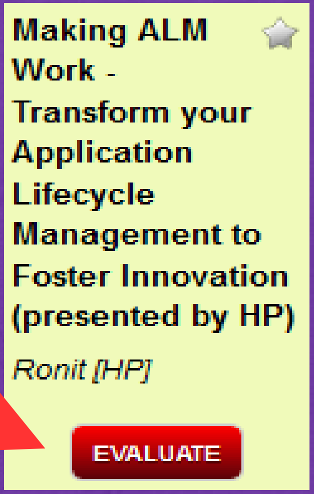
- JDT delivers powerful program manipulation services
  - Java Model, Search engine and DOM AST
    - Use them to add your own tool to the Eclipse Java IDE
  - but also in headless mode (can be used programmatically)
    - E.g. EMF, metrics tools, …
  - Full J2SE 5.0/6.0/7.0 support
  - Full-fledged batch compiler

- Community feedback is essential
  - bug reports: https://bugs.eclipse.org/bugs/enter_bug.cgi?product=JDT
  - Forum: http://www.eclipse.org/forums/index.php/f/13

Give Feedback on the Sessions

1 Sign In: www.eclipsecon.org

2 Select Session Evaluate


Making ALM Work - Transform your Application Lifecycle Management to Foster Innovation (presented by HP)

Ronit [HP]

3 Vote +1 0 -1

Legal Notice