Pocket PC

<medula> Hey where are you?
<codaca2> wandering
<medula> but we are still chatting?!
<codaca2> Hyperbola and OSGi go anywhere
<medula> slick
Linux/GTK

<medula> Hey there!
<codaca> Hi, this Hyperbola chatting is really cool!
<medula> Yeah, it was so easy to put together using Eclipse and OSGi
GTK Standalone

Hyperbola

Friends

alize

medula

medula@jabber.org - Chat

<codaca> Hey, you remember that app you showed me on the Pocket PC?
<medula> Yeah. What have you done now?
<codaca> Running it here on Linux/GTK.
<medula> You sure are getting around with that example!
Integrated
Observations

• Each is running the same application code
• Each is running on OSGi
• Each uses third party libraries
  – not written for OSGi or Eclipse
• Each looks and feels native to its environment

The realities of a converged world
Eclipse

- Eclipse 3.0 based on OSGi R3++
- Platform for tooling and generic applications
- Plug-in model based on OSGi bundles
- Extension mechanism
- GUI framework and native widget toolkit
- 40+ mirrors: 100,000s of downloads
OSGi meets Eclipse

• OSGi is essential but not enough
  – Services
  – Dynamic bundle management
• Component model that scales up to the desktop
  – 1000s of bundles
  – Declarative extensibility
  – Lazy installation and activation
• UI model that scales down to the handheld
  – Common programming model
  – Native widget support

Complementary Technologies
Technical Challenge: Scaling

- Desktop apps today have 1000s of bundles
- Embedded applications increasingly componentized
- Being lazy
  - Activate on use (Eclipse)
  - Extension Registry (Eclipse)
  - Declarative Services (RFC 80)

*What is needed, when it is needed*
Technical Challenge: Dynamics

• Dynamic function
  – On-demand acquisition
  – Dynamic bundle install/uninstall

• Dynamic data
  – Data migration between devices as needed
  – General purpose mechanism hard (years of research)
  – Domain solutions possible/available
Technical Challenge: Going Native

- Look and feel must be natural
- Increase consumer acceptance
- Use real window system widgets
- Branding show-thru on custom UI libraries

That’s running Java?!
Eclipse RCP

• Rich Client Platform
  – Runtime (OSGi + extension mechanism)
  – SWT (widget library)
  – JFace/UI (GUI framework libraries)
  – Update (bundle management agent)

• Addresses the challenges
• Production quality
• Open-source
• Shipping today
embedded RCP (eRCP)

- Eclipse Technology project
- Nokia, Motorola, IBM et al
- Eclipse on devices (i.e., embedded versions of)
  - Runtime (OSGi + infrastructure)
  - SWT (widget library)
  - JFace/UI (GUI framework libraries)
  - Update (bundle management agent)
- Address embedded-specific issues
  - Speed, space, mobile function, …
Summary

• Keys to convergence
  – Scalability \(\Rightarrow\) Laziness
  – Deep integration
  – Dynamic behaviour

• Eclipse and OSGi deliver applications across the enterprise from mobile devices to the desktop
Other information

• eRCP talk Wednesday
• OSGi and Open source panel Thursday
• eclipse.org