An Introduction to ECF

Chris Aniszczyk <zx@us.ibm.com>
Software Engineer
IBM Lotus
Agenda

- Background
  - ~15 minutes
- Demos
  - ~10 minutes
- API Introduction
  - ~5 minutes
- Conclusion
  - ~15 minutes
- Q&A
Agenda

- Background
- Demos
- API Introduction
- Conclusion
- Q&A
Background

- Eclipse does many things well…
  - World-class Java IDE (JDT)
  - Fantastic Web Development tooling (WTP)
  - Database tooling (DTP)
  - and so on…

- …but lacks support for true collaboration
  - Messaging
  - Shared Editing
  - Shared Debugging
  - etc…

- Enhanced support for collaboration within the context of Eclipse can increase productivity
Background

- Competition is good 😊
- Netbeans
  - [http://collab.netbeans.org](http://collab.netbeans.org)
Background

- **Subthaedit**
  - [http://www.codingmonkeys.de/subthaedit](http://www.codingmonkeys.de/subthaedit)
Background

- Solution?
- The Eclipse Communications Framework (ECF) to the rescue!
  - Yet Another #$!!@! Framework
    - A cross-protocol communications framework for Eclipse/RCP
    - ECF provides a set of high-level abstractions, rather than yet another messaging API to support various communications components.
    - With ECF, development can be expedited over all the communication code for each of those components, allowing you to focus on business logic and UIs.
Background

- ECF creates value for 4 groups
  - **Communications providers** (Yahoo, GoogleTalk/XMPP, etc…)
    - Adoption & Interoperability
  - **Component developers** (file-sharing, screen-sharing, etc…)
    - Reusability
      - Developers can reuse components
  - **Tool integrators**
    - Feature enrichment
      - Developers can breathe new life into their existing applications
  - **UI developers**
    - Usability
      - UIs can be improved/customized independent of underlying implementation
Demos
Demos

- Eye candy is **important 😊**
  - XMPP (GoogleTalk) IM, Yahoo IM
    - Who wants to chat with me ;)?
  - IRC
    - irc://irc.freenode.net/#eclipse-dev
- Shared Workspaces
  - URL Sharing
  - Shared Editing
Agenda

- Background
- Demos
- API Introduction
- Conclusion
- Q&A
API Introduction

- Interoperability through protocol
  - org.eclipse.ecf.core.IContainer
  - Goal
    - Simple API / Extensibility via OSGi model / getAdapter(...) 

- Clients use the IContainer API
  - IContainer container = ContainerFactory.getDefault().createContainer("ecf.xmpp.smack");
  - Container.connect(...)

- Semantics
  - Connection/Disconnection/LifeCycle
    c.connect(ID, IConnectContext)
    ...
    c.disconnect()
  - Protocol Adapters – getAdapter(...) abuse...
    - container.getAdapter(<interface>);
    - IFileshareContainer fsc = (IFileshareContainer) c.getAdapter(IFileshareContainer.class)
API Introduction

- IAdaptable abuse (we love the adapter pattern)
  - Presence/IM/Chat
  - Dynamic Service Discovery (zeroconf, etc…)
  - Datashare (channels)
  - File sharing
  - Call (SIP…)

Software. IN CONCERT.
API Introduction

▪ Two Extension Points
  ▪ `org.eclipse.ecf.containerFactory`
    ▪ ECF providers can implement their own `IContainer`
      - Current
        - XMPP/Jabber, IRC, JMS, Yahoo
      - Future
        - SIP, JXTA, Jingle, Sametime, AIM, etc…
  ▪ `org.eclipse.ecf.namespace`
    ▪ ECF providers can implement their own addressing
      - e.g., xmpp://zx@ecf.eclipse.org
Agenda

- Background
- Demos
- API Introduction
- Conclusion
- Q&A
Conclusion

- Future Direction
  - Corona Integration (http://www.eclipse.org/corona/)
    - OSGi-based SOA component framework for server-side Eclipse plug-in deployment
      - ECF usage for event reporting / resource sharing
  - New “sub-projects”
    - Shared Editing
    - Call API/VOIP/Asterisk/GoogleTalk
      - RCP Soft-phone
    - Application Sharing/VNC
    - ECF+OSGI for Servers
      - RSP (Rich Server Platform) / Equinox Servlet Incubator…
Conclusion

- Google’s Summer of Code Projects ([http://code.google.com/soc](http://code.google.com/soc))
  - **ECF BitTorrent Provider**
    - Create an EPL’d BT implementation that will be used by the file-sharing API
    - Investigate integration with Eclipse’s Update Manager
  - **Real-time Shared Editing Support**
    - Enable pair-programming sessions in Eclipse, really ;)
    - Possibly use the SubEthaEdit protocol
  - **SharedCode Plugin (SCP)**
    - Provide an easy to use UI to search and share source code amongst developers
Conclusion
Conclusion

- Support from the community **welcomed** and **appreciated**!
  - Ways to help
    - Improve / Extend existing applications
    - Build provider implementations
    - Contribute to API Design (we need feedback!)
    - Join ECF Community! (commercial or not!)

- Website
  - http://www.eclipse.org/ecf

- Mailing List
  - http://dev.eclipse.org/mailman/listinfo/ecf-dev

- Newsgroup
  - news://news.eclipse.org/eclipse.technology.ecf
Agenda

- Background
- Demos
- API Introduction
- Conclusion
- Q&A
Questions
Thank You

Chris Aniszczyk <zx@us.ibm.com>
http://mea-bloga.blogspot.com
http://www.eclipse.org/ecf