What's Next: RT Text Editing

- Additional Meta-Data
  - Shared Markers (problems, tasks, etc)
  - Parse tree, generated artifacts
- Association with Other Resources
  - Inter-resource Dependencies
  - Navigation operations
- Other requests from Community
  - Full support for auto-complete, quick fix, etc
  - VCS integration

What's Next: Graphical Models and Beyond

- Graphical Operations
  - node position
  - Connect nodes
  - Disconnect node
  - etc
- Appropriate Transformations?
- EMF Integration
- Distributed Model Synchronization Framework?

ECF Project Goal

Lower Barriers to Team and Community Communication

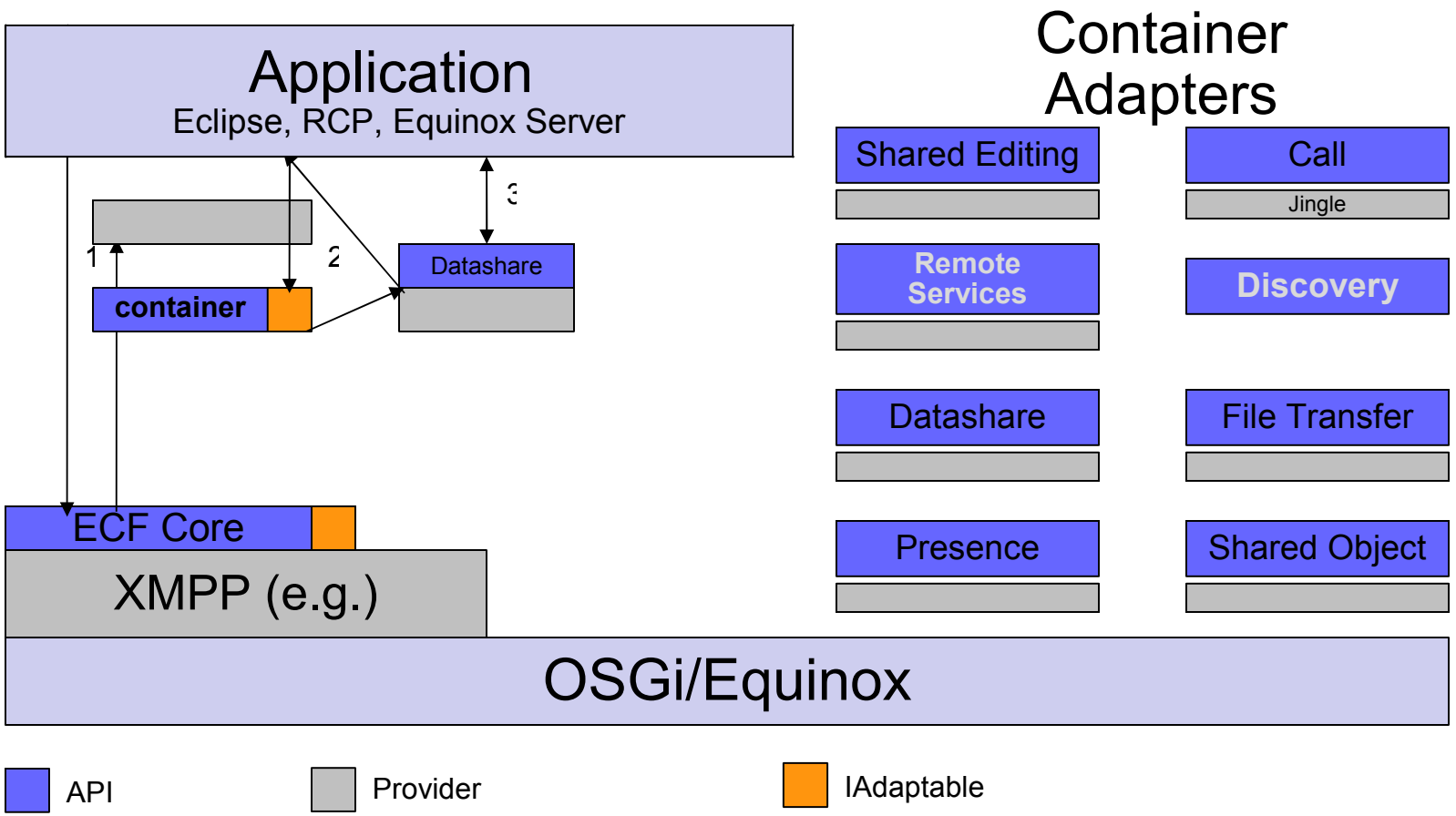
by providing

Interoperable, Integrated, Extensible Framework

Proud to Be an Open, Community-Run (non-corp controlled) Eclipse Project

A few harmless flakes working together can release an avalanche of destruction

ECF Architecture



Code

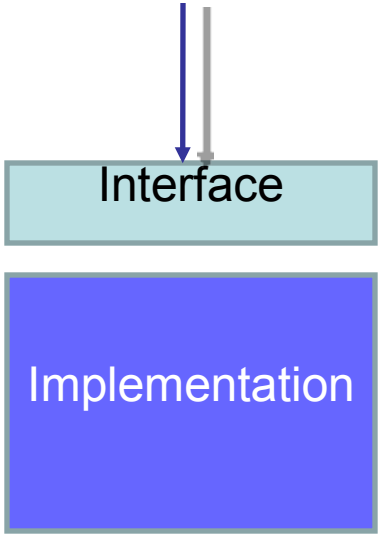
```
IContainer container =
  ContainerFactory.getDefault().createContainer();

IChannelContainerAdapter adapter = (IChannelContainerAdapter)
  container.getAdapter(IChannelContainerAdapter.class);

IChannel channel = adapter.createChannel(...)
```

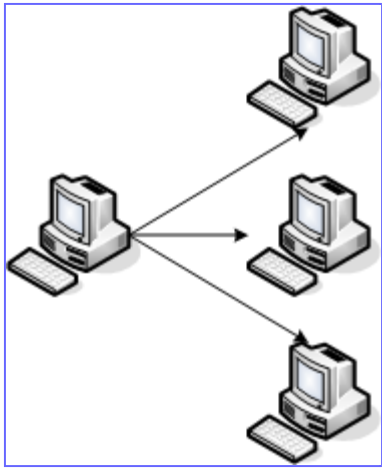
OSGi Services

- OSGi services provide
  - Encapsulation at a larger granularity
  - Loose coupling of functionality
  - Extensibility
  - Abstraction
- Remote services
  - Take this existing boundary to turn an application into a distributed application
  - Provide an abstraction to design distributed apps



OSGi services in the network

- Locate a service
  - Implementation for a given interface
  - Service discovery
  - Common knowledge
- Making use of a service
  - Providing service access via ECF API
  - “importing” the service into the local service registry
  - Providing a local service proxy

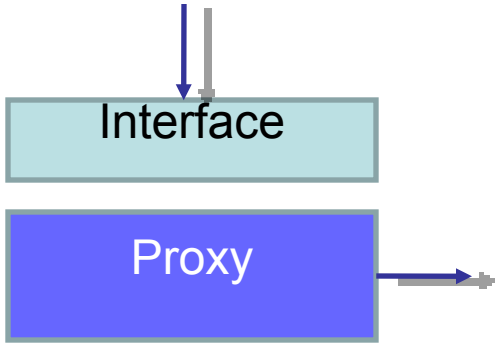


ECF service discovery – Overview

- Query for known/available services
  - Synchronous
  - Asynchronous: add/remove a service listener and get notified about service discovery/”undiscovery”
  - Query by filter/example (TODO)
- Manual and automatic service announcement

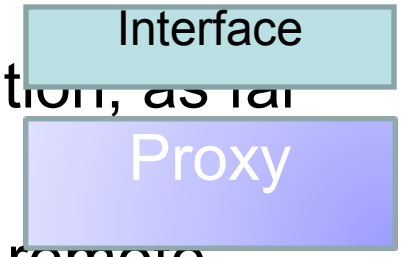
Remote Services

- OSGi services which cross address spaces
- Same ideas:
  - Ask for a service (-reference)
    - Can trigger service discovery
  - Get the service
    - Get a proxy for the service
    - Proxy generation can be proactive or reactive
  - Use the service
    - Method invocations become remote invocations



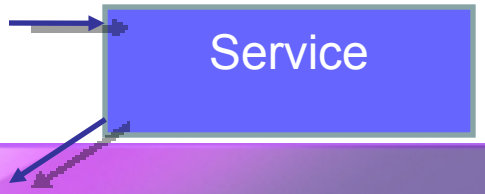
Transparent API

- Service and client remain untouched
- Some entity (not the client) states the demand
- Proxy is already present when the client asks for the service
- The service remains agnostic against distribution, as far as possible
- Seamless and flexible transition from local to remote services



Non-Transparent API

- Client is aware of distribution
  - Retrieve an IRemoteService object
  - Explicit app-level failure handling
- Explicitly call remote invocations
- Call semantics can differ from local service calls
  - One-shot invocation (non-blocking)
  - Asynchronous invocation
    - E.g., with listener callback
    - Futures



Google Can Contribute

- Committer and Code Contributions
  - Jump In and Work With Community
- Usage
  - Try It
  - Fix it/extend it to your liking
  - Integrate with Google APIs: Calendaring, Google Groups, Social Networking APIs, Others
- Other
  - Support/Donate to Project (e.g. hw)
  - Hire people and let them contribute to ECF

Eclipse ECF Project

- http://www.eclipse.org/ecf
- http://wiki.eclipse.org/ECF
- ECF 2.0 will ship with Eclipse Ganymede
- The work on ECF 2.1 has just started ☺