Rational. software

Inside the Jazz Technology Platform

Kai-Uwe Maetzel, Scott Rich – Jazz Team

IBM Rational Software Development Conference 2007

What keeps me **Rational**?

IBM

Rational. software

Deep Inside the Jazz Technology Platform

Kai-Uwe Maetzel, Scott Rich – Jazz Team

IBM Rational Software Development Conference 2007

What keeps me **Rational**?

IBM

Agenda

- Jazz Platform architectural goals and overview
- Jazz Platform deep dive illustrated with an example
  - From the client to the server

Jazz Platform Value Proposition

- Integrated collaboration improves team effectiveness
- Designed for organizationally distributed development
- Process integration for enhanced productivity, repeatability, and best practices
- Seamless integration across the software development lifecycle
- Traceability across the life cycle
- Non-intrusive visibility into project status and health information
- Lower Total Cost of Ownership
- Scalability: process and team and project size
- Extensible on the client and server
- A growing Eclipse-based community

Your Projects, Teams, and Items in Jazz …

- Project areas reify projects and team areas reify teams
- Activities within a project are governed by the project's process
- The project's process differs from team to team

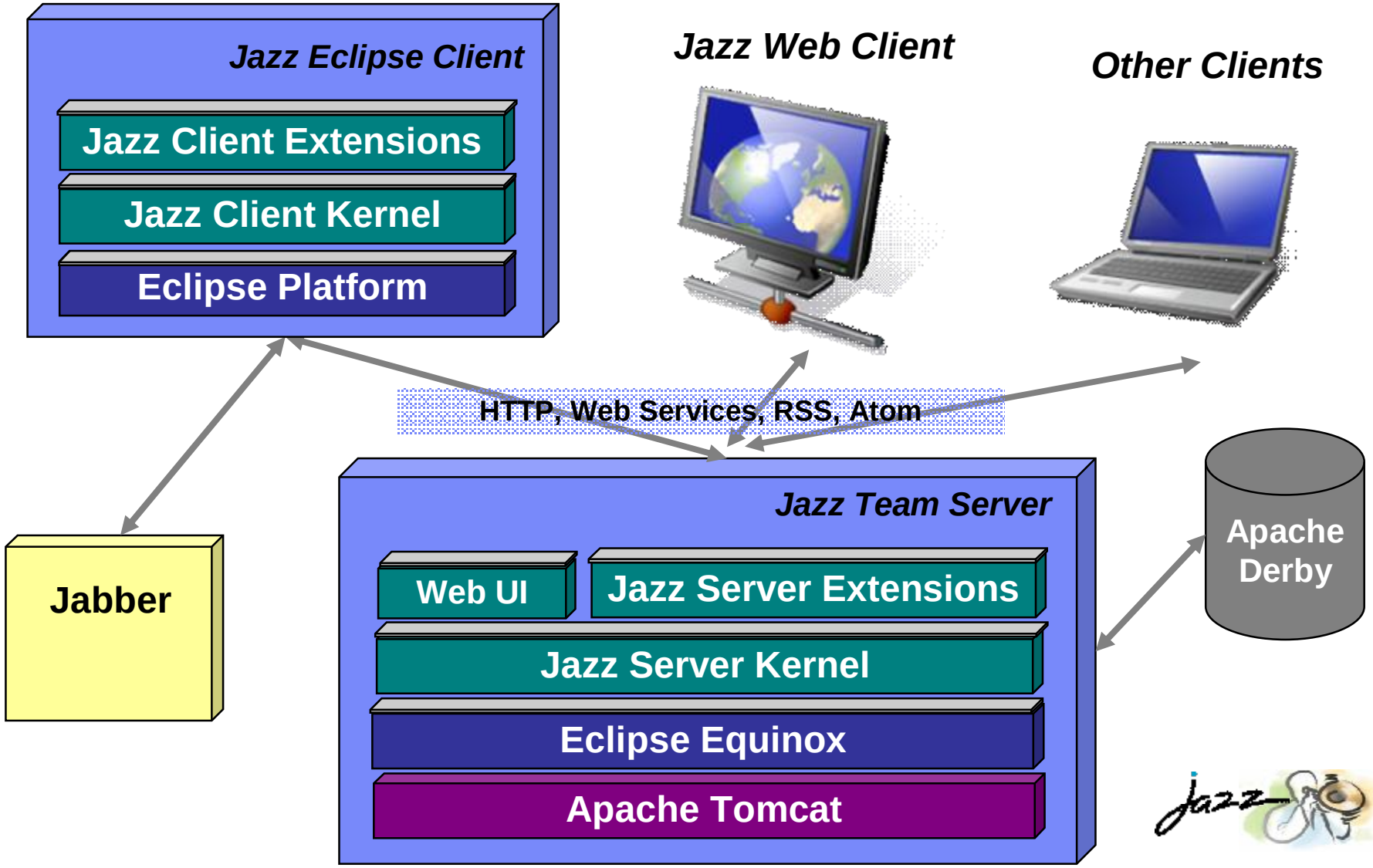
- Items belong to projects and are owned by teams
- Items are persisted on the server
- Items can be created, be updated, and be deleted
- Items have history
- Items have types
- The set of item types is extensible

Jazz Platform Architectural Goals

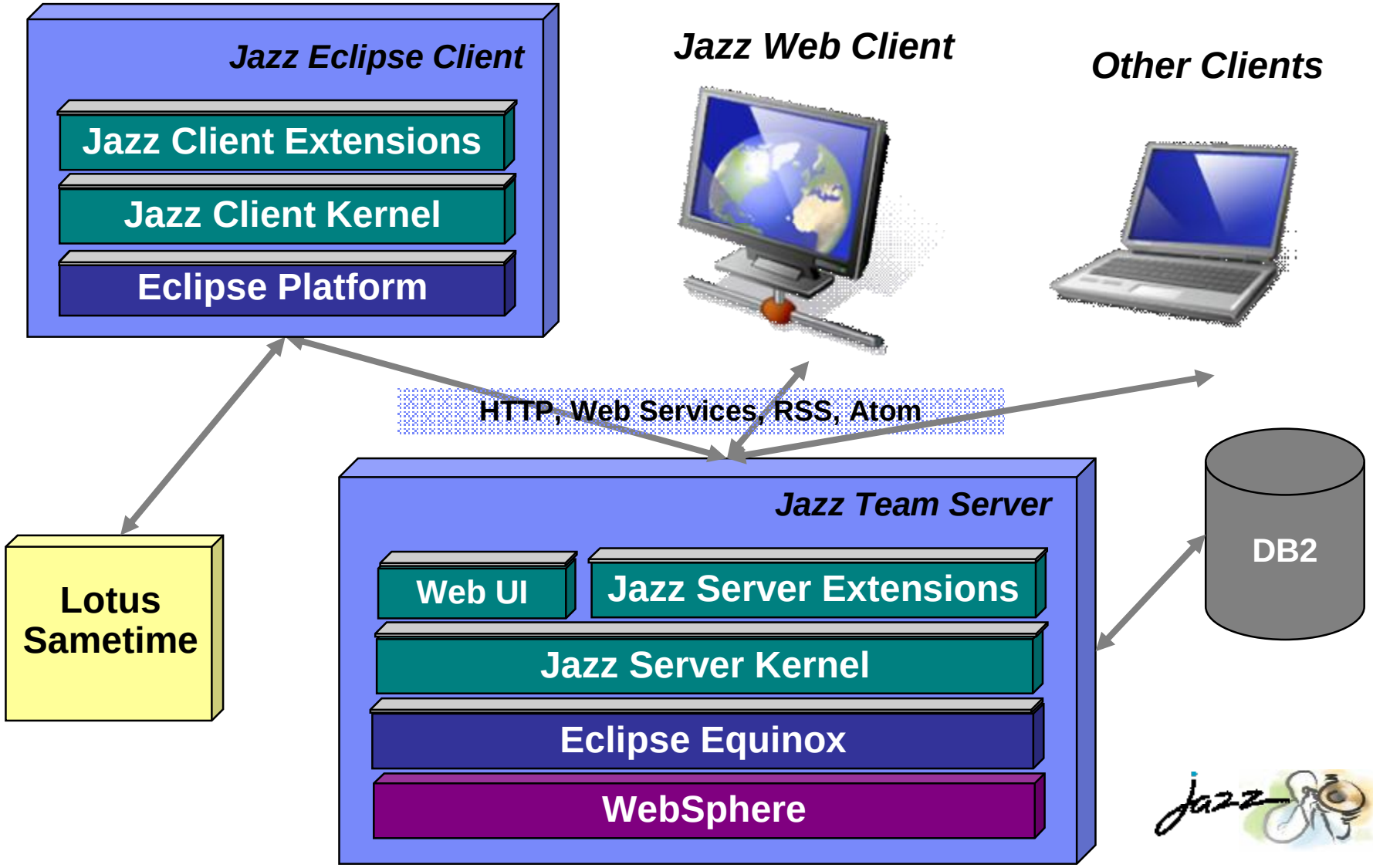
- Team first
  - Projects / teams / feeds
  - Continuous project health “for free”
- Configurable through process
- Easily extensible

- Multi-client, WAN-friendly
- Scalable
- Secure
- Standard middleware platform

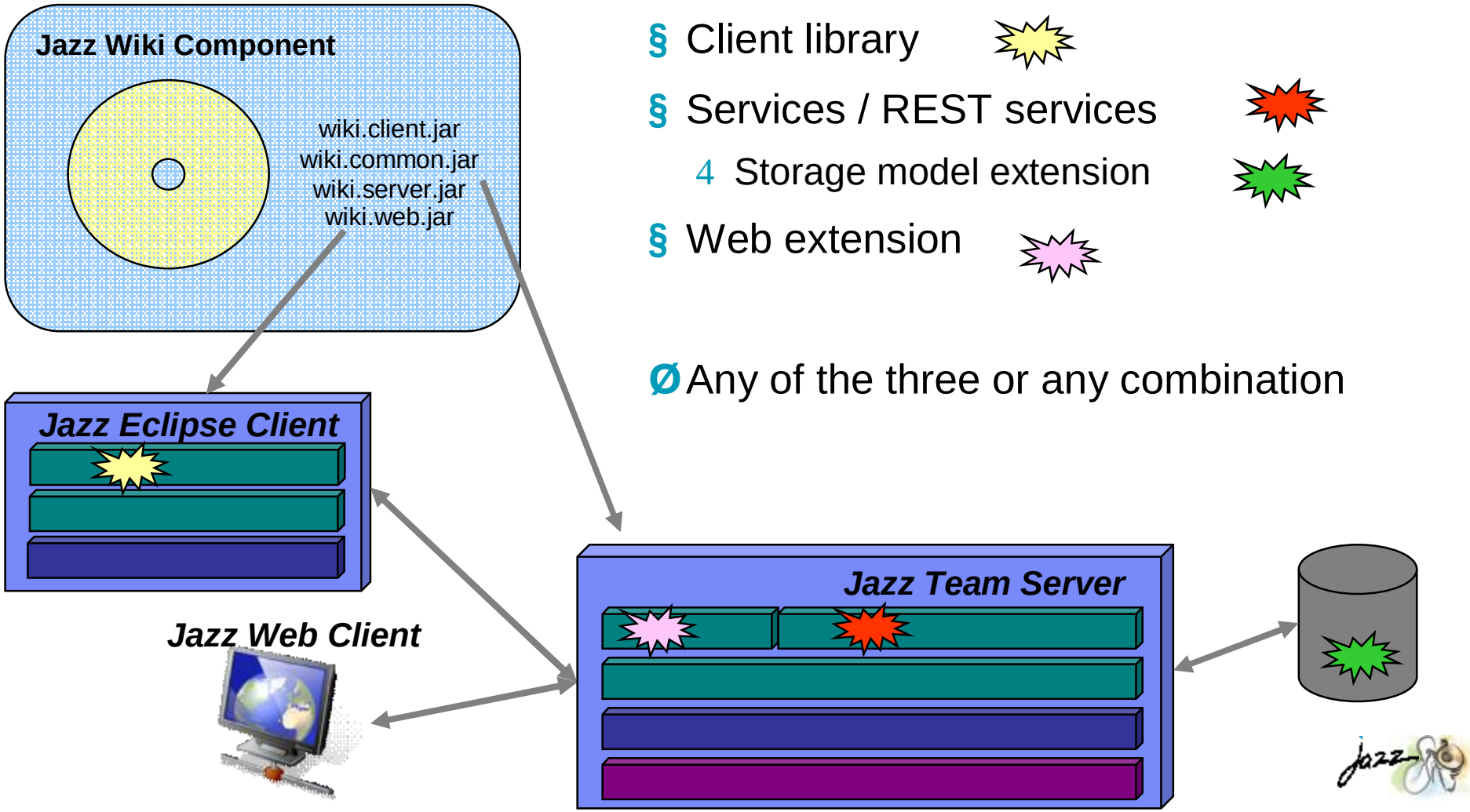
Jazz Platform Architecture – Open Source middleware



Jazz Platform Architecture – IBM middleware



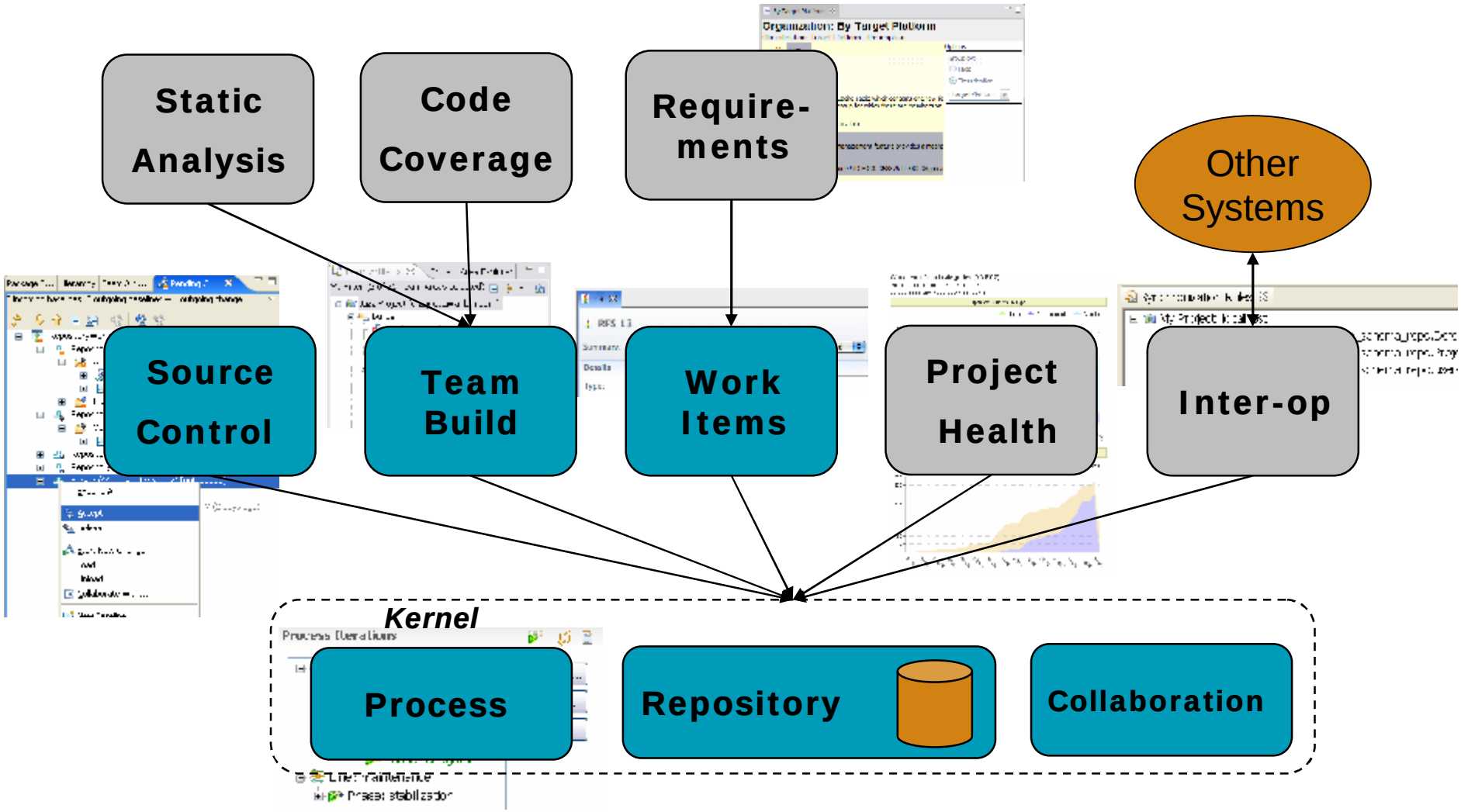
Anatomy of a Jazz Component



- Client library
- Services / REST services
  - Storage model extension
- Web extension

Ø Any of the three or any combination

Jazz Platform Components Overview



No Exceptions

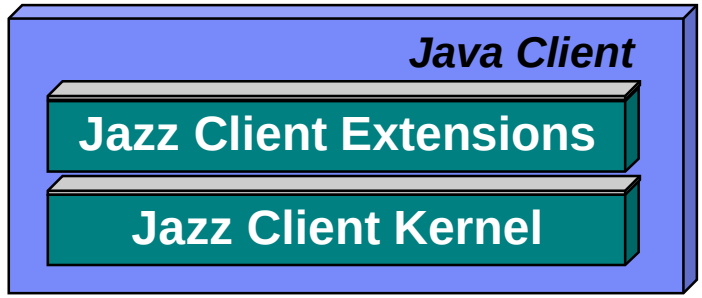
- All components are following the same pattern.

- There are no exception.
  - Not even for Jazz.

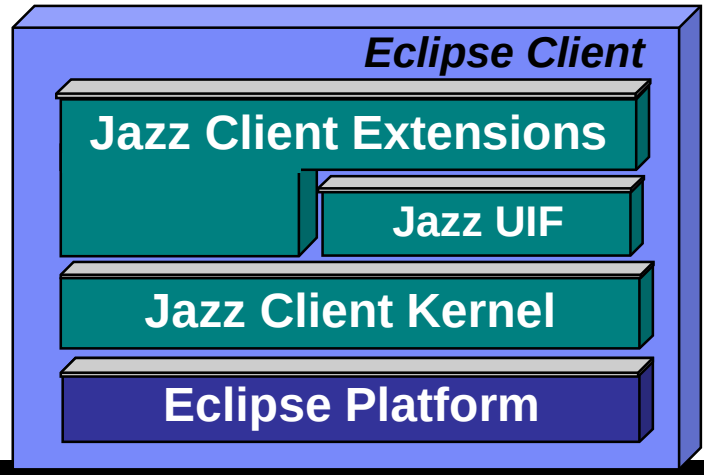
Deep Dive Roadmap

- Clients
  - Eclipse RCP & plain Java clients
  - Web UI
- Server
  - Server extensibility deep dive
  - Storage model
  - Service programming model
- Process enablement
  - Customization
  - Participation

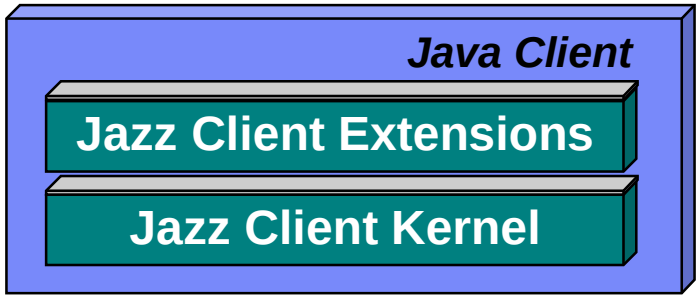
Client Applications



- Jazz Client Kernel
  - Provides access to and manages repository connections
  - Provides the basics for querying, fetching, and caching repository data
  - Provides access to and manages feed subscriptions and feed data
  - Provided access to and manages client libraries
  - Provides the notions of projects, teams, and process
  - Runs in OSGI environments such as eclipse RCP and in plain java clients
- Jazz UI Foundation
  - Basic UI building blocks for Jazz
- Jazz client extensions
  - Client libraries
  - UI or UI extensions



Plain Java



- It is a normal plain Java application
- Establishes one or more connections to Jazz repositories
- Uses (and provides) client libraries

```
TeamPlatform.startup();
ITeamRepositoryService srv = TeamPlatform.getTeamRepositoryService();
ITeamRepository repo = srv.getTeamRepository(myRepositoryURL);
repo.login(progressMonitor);
IQueryClient qryClnt = repository.getClientLibrary(IQueryClient.class);
IQueryResult result= qryClnt.findAll(myQueryExpression);
processWorkItems(result);
```

What are Client Libraries?

- Provides a sophisticated façade for a component's functionality
- Instantiated only once per repository connection
  - may have state and allow for listeners
- Talks to Jazz platform and component specific services on the server
  - Can leverage smart, optimized communication patterns
- Manages the caching of the component's repository data
- Can depend on and talk to other client libraries

- Interesting client side applications are possible by using the Jazz Client Kernel and existing client libraries

Example: The Wiki Client Library

- Registered as extensions with a Jazz Client Kernel extension point

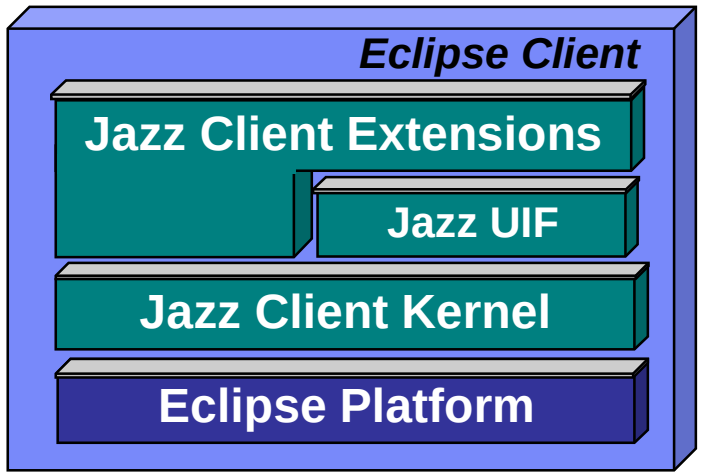
```
<extension point="com.ibm.team.repository.client.clientLibraryFactory">
  <factory
    factoryClass="com.ibm.team.wiki.internal.WikiClientLibraryFactory"
    interfaceClass="com.ibm.team.wiki.client.IWikiManager"
    name="Foundation Wiki Client Factory"/>
</extension>
```

Example: The Wiki Client Library

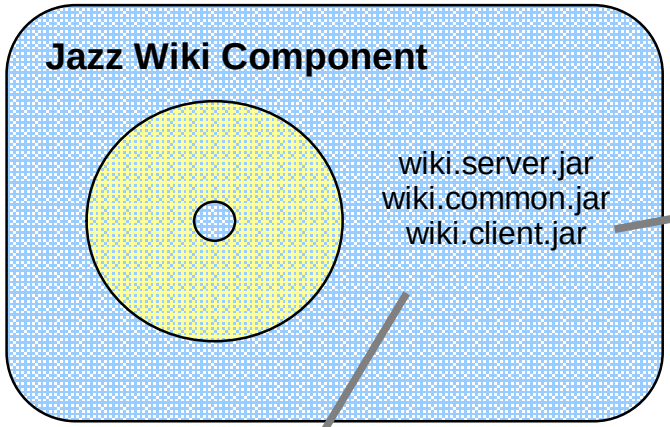
```
public interface IWikiManager {

  IWikiPage createPageUsingOwner(
      IItemHandle owner, String pageId, IProgressMonitor monitor)
    throws TeamRepositoryException;

  IWikiPage findPageUsingOwner(
      IItemHandle owner, String pageId, IProgressMonitor monitor)
    throws TeamRepositoryException;

  IWikiPage createPageUsingPath(
      WikiPath wikiPath, IProgressMonitor monitor)
    throws TeamRepositoryException ;

  IWikiPage findPageUsingPath(
      WikiPath wikiPath, IProgressMonitor monitor)
    throws TeamRepositoryException;

  IWikiPage savePage(
      IWikiPage page, XMLString content, IProgressMonitor monitor)
    throws TeamRepositoryException, IOException;
}
```

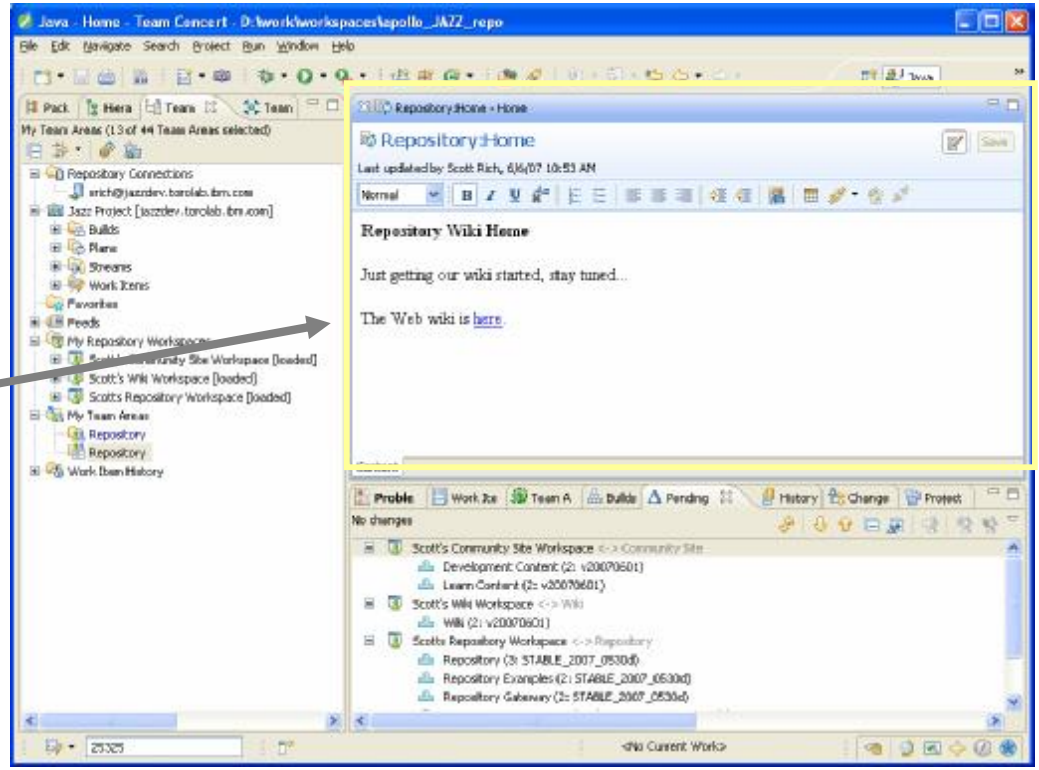
Inside a Jazz RCP Client Application

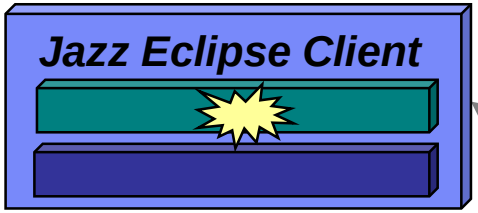
- It is a normal Eclipse RCP application
  - Plug-ins defining extensions for available Eclipse RCP extension points such as views, editors, wizards, actions, etc. …
- Uses the Jazz client kernel
- Uses (and provides) client libraries
- Extends the Jazz UI Foundation
- Extends the UI provided by other Jazz components



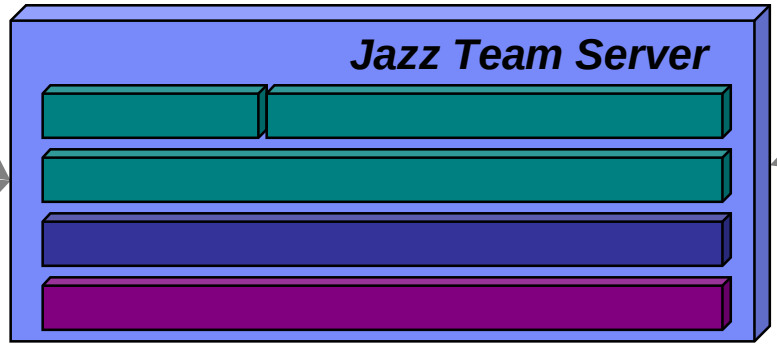
Extending the Jazz Platform Component UI

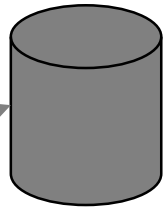




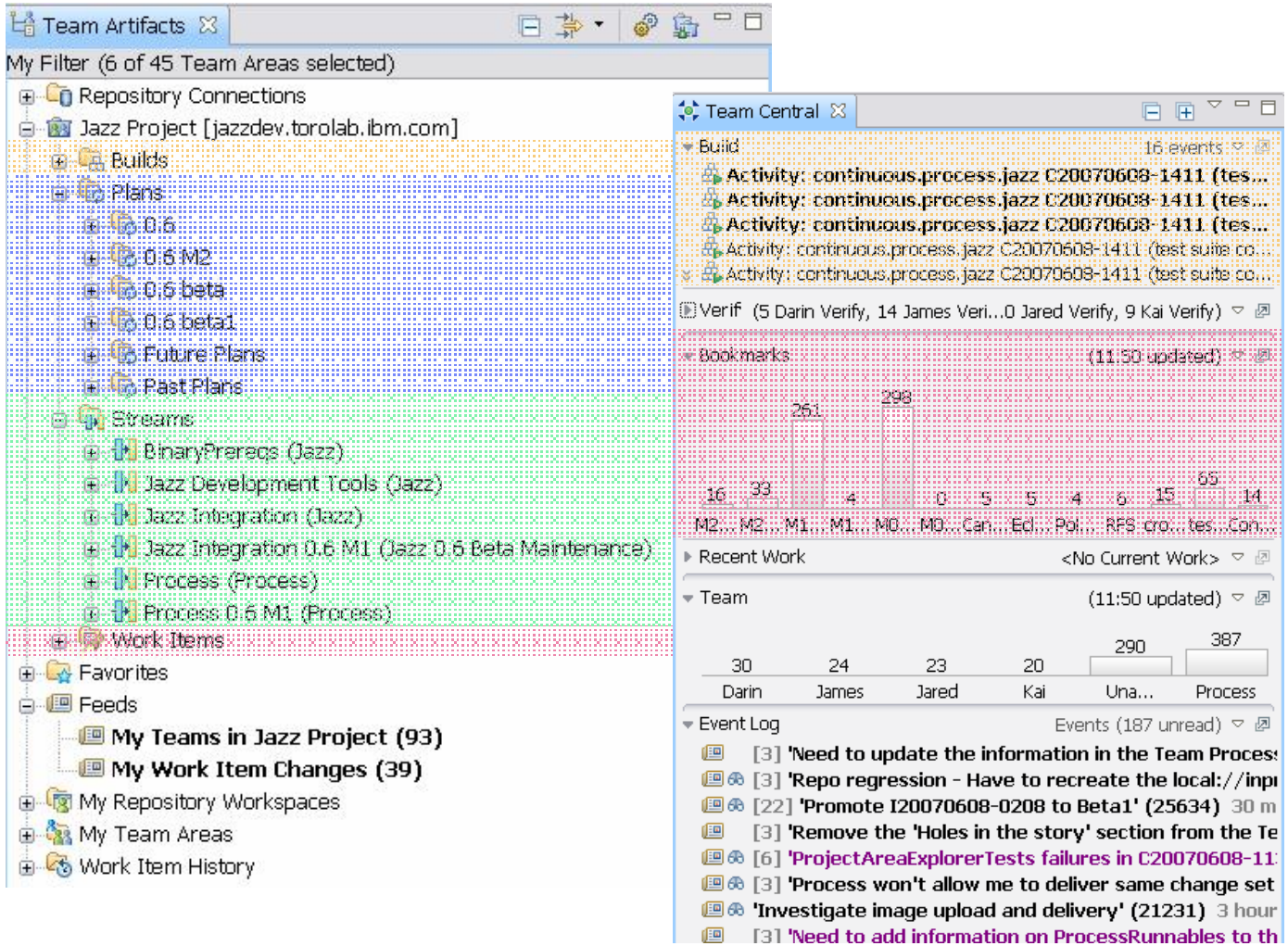


Jazz Web Client

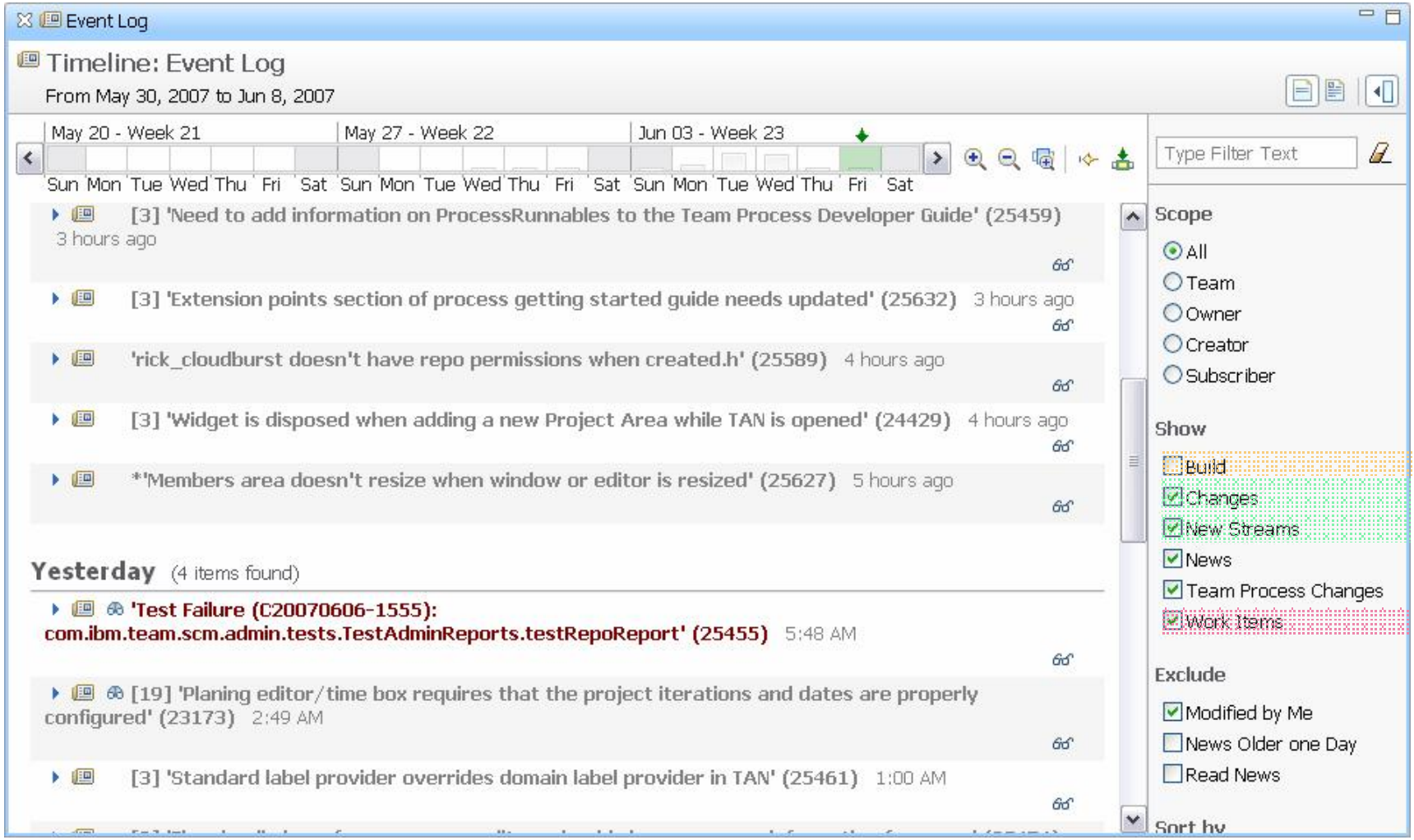




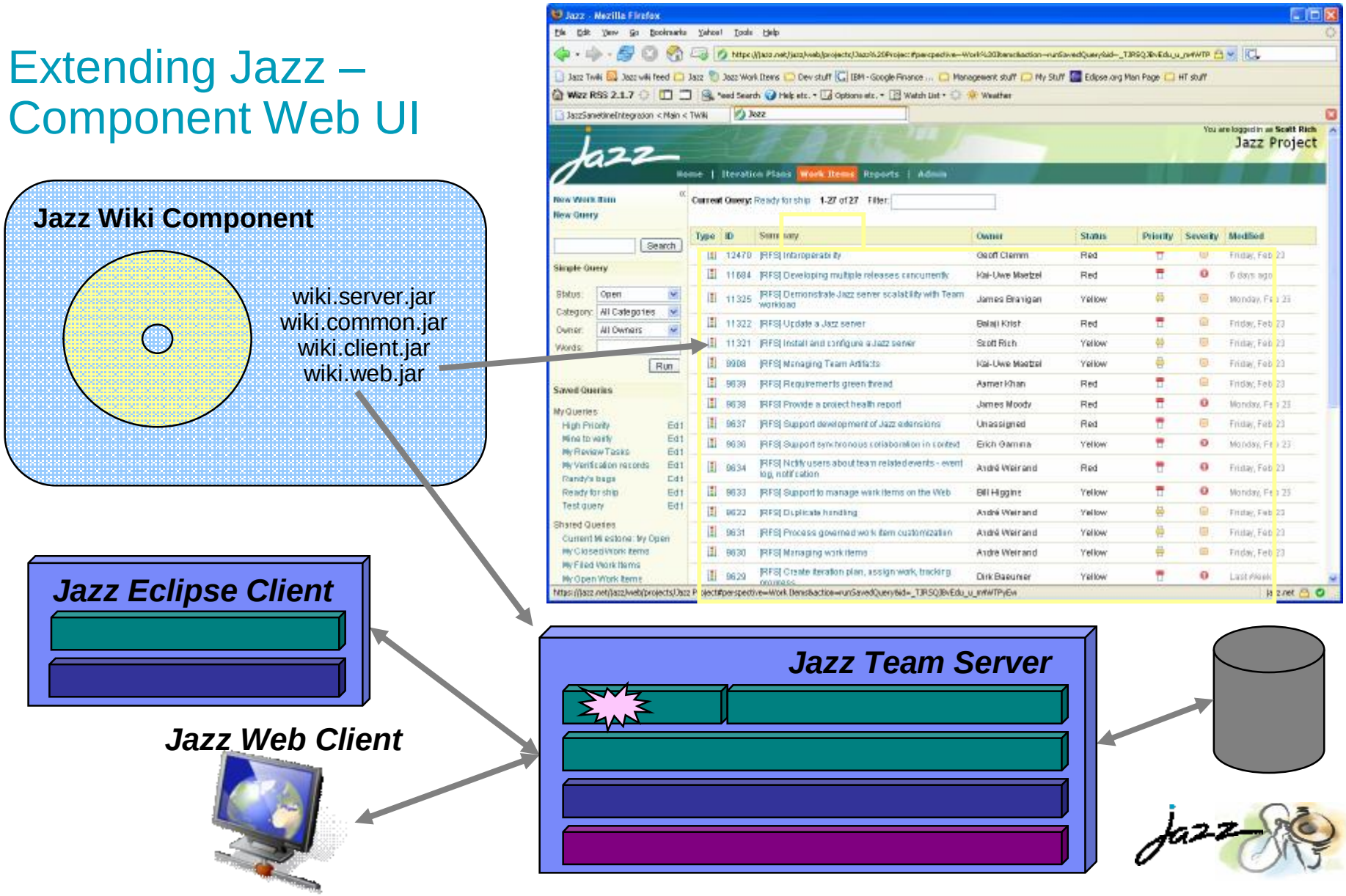
Jazz UI Foundation



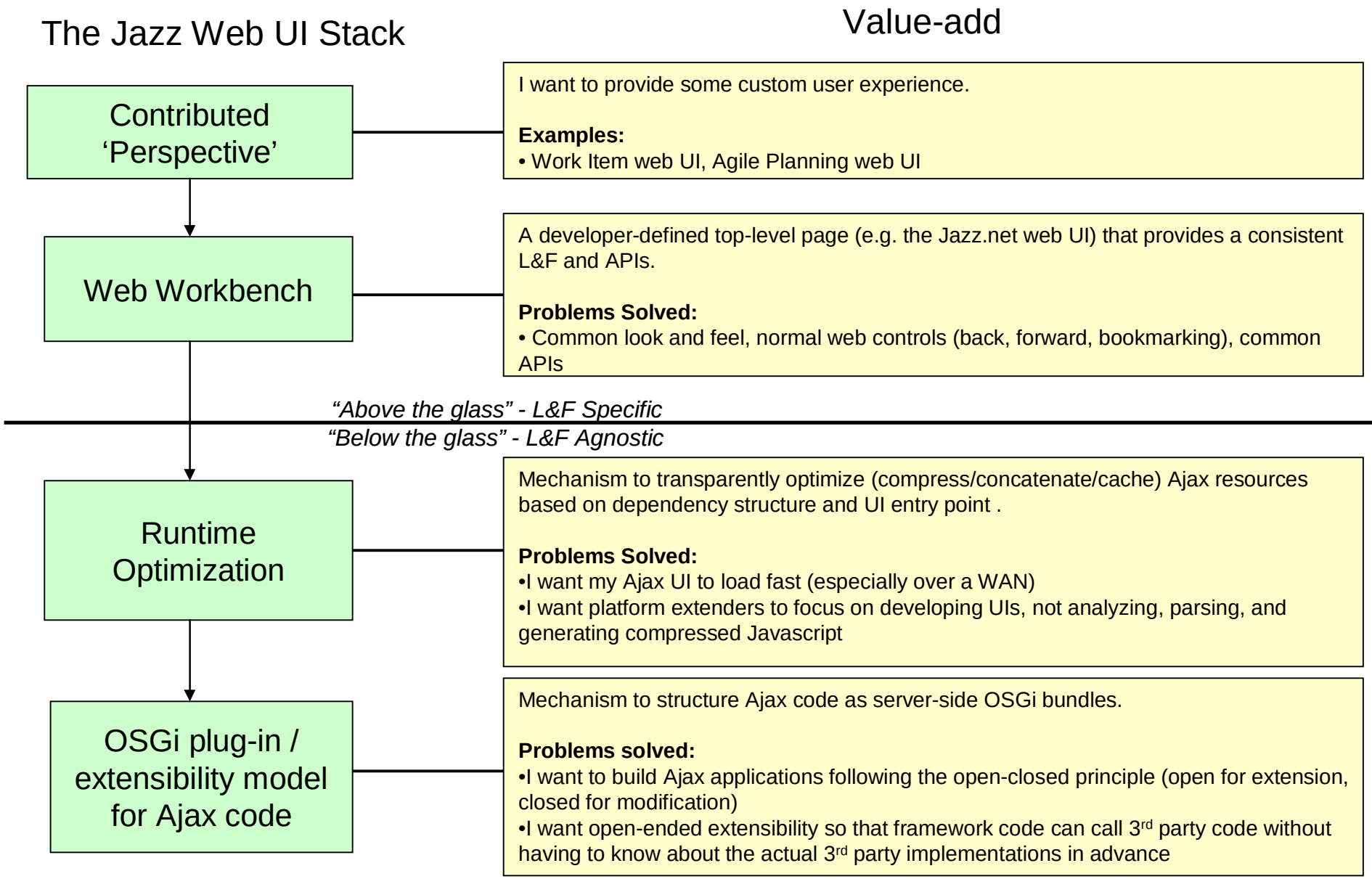
Jazz UI Foundation



Extending Jazz – Component Web UI



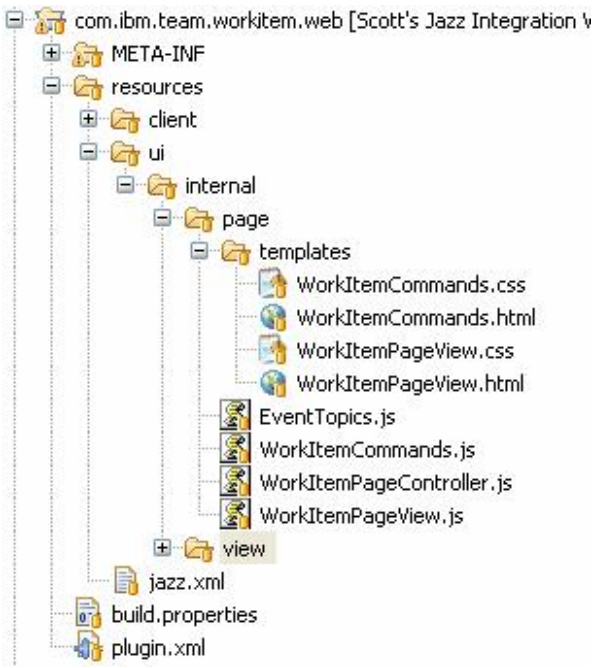
The Jazz Web UI Stack



Extending Jazz – Web UI (plugin.xml)

```
<!-- work item web UI bundle registration -->
<extension point="com.ibm.team.repository.web.webBundles"/>

<!-- work item web UI page definition -->
<extension point="com.ibm.team.repository.web.basicPerspectives">
  <basicPerspective
        id="com.ibm.team.workitem.web.WorkItemPage"
        widget="com.ibm.team.workitem.web.ui..WorkItemPageView"
        name="Work Items"/>
</extension>
```

Extending Jazz – Web UI (JavaScript)



WorkItemPageView.js:

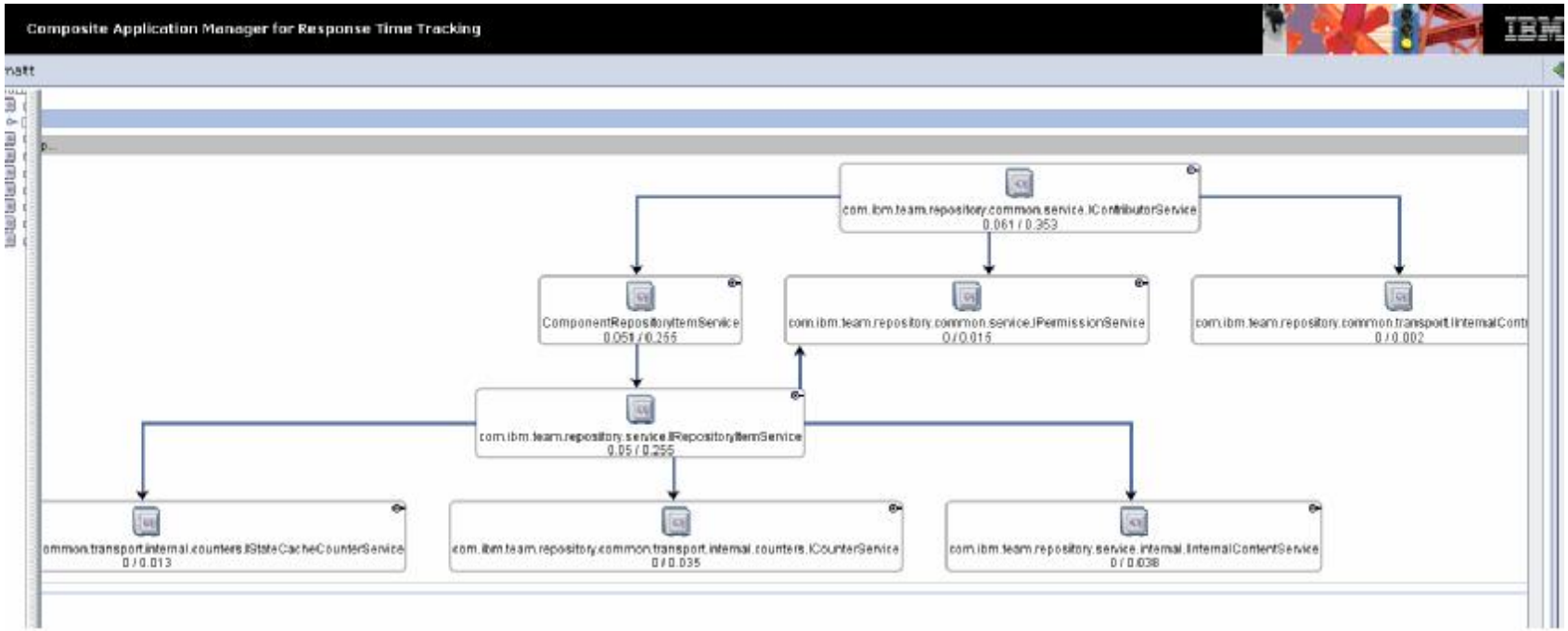
```
/*
 * The main view for the work item web UI. Responsible for displaying left-nav
 * content and for managing the main content area.
 */
dojo.widget.defineWidget(
"com.ibm.team.workitem.web.ui.internal.page.WorkItemPageView",
dojo.widget.HtmlWidget,
{
//
// Other work item code may call these properties and methods.
//

/*
 * Returns the internal id of the widget currently displayed in the
 * page's content area.
 */
getCurrentWidgetId: function() {
return this._currentMainWidgetId;
},
/*
 * Returns the work item object currently displayed in the page's
 * content area.
 */
getCurrentWorkItem: function() {
return this._multipaneContentWidget.getCurrentWidget()._workItem;
...
```

Example of extending the server

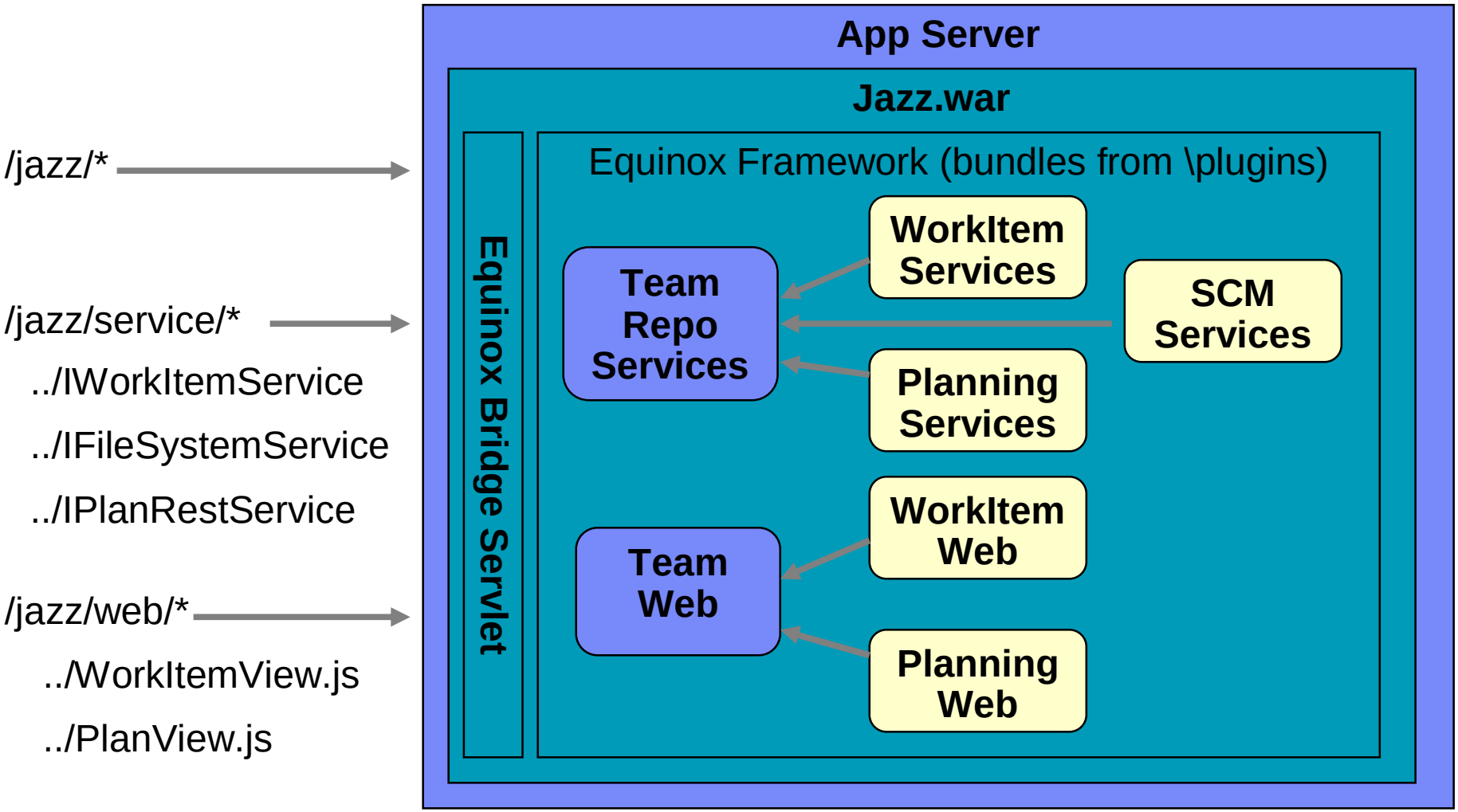
- Introduce higher-level and transactional behavior
- Teach the repository about a new item type
- Enable clients and other server extensions to use it

Extending the platform - Service programming model

- It's stateless services all the way down…
- Services are contributed by bundles
- Statelessness is key to scalability
  - All state is in the DB, all client interactions are atomic
  - Clusterable and restartable
  - No session state (clients are stateful)

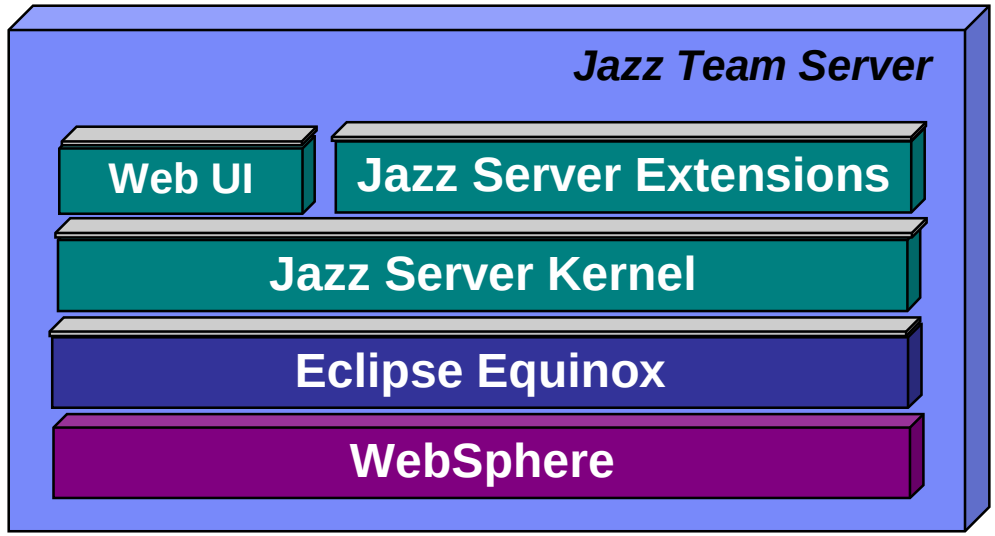


Extending the Jazz Platform – Understanding jazz.war



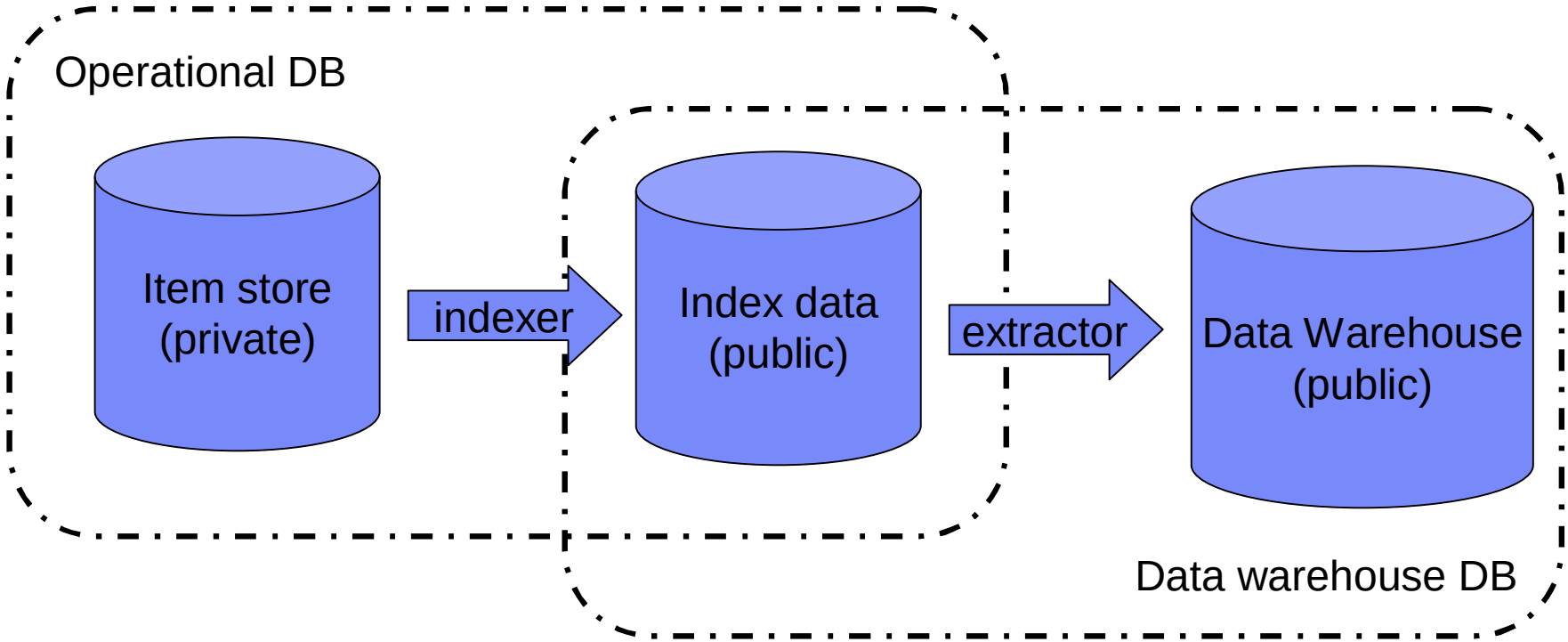
Jazz Server Kernel

- Team Repository Services
  - Provides the basics for querying, fetching, and updating repository data
  - Provides support for change events
  - Provides support to generate feeds
  - Provided access to and manages component services
  - Provides the notions of projects, teams, and process
  - Runs in OSGI environments

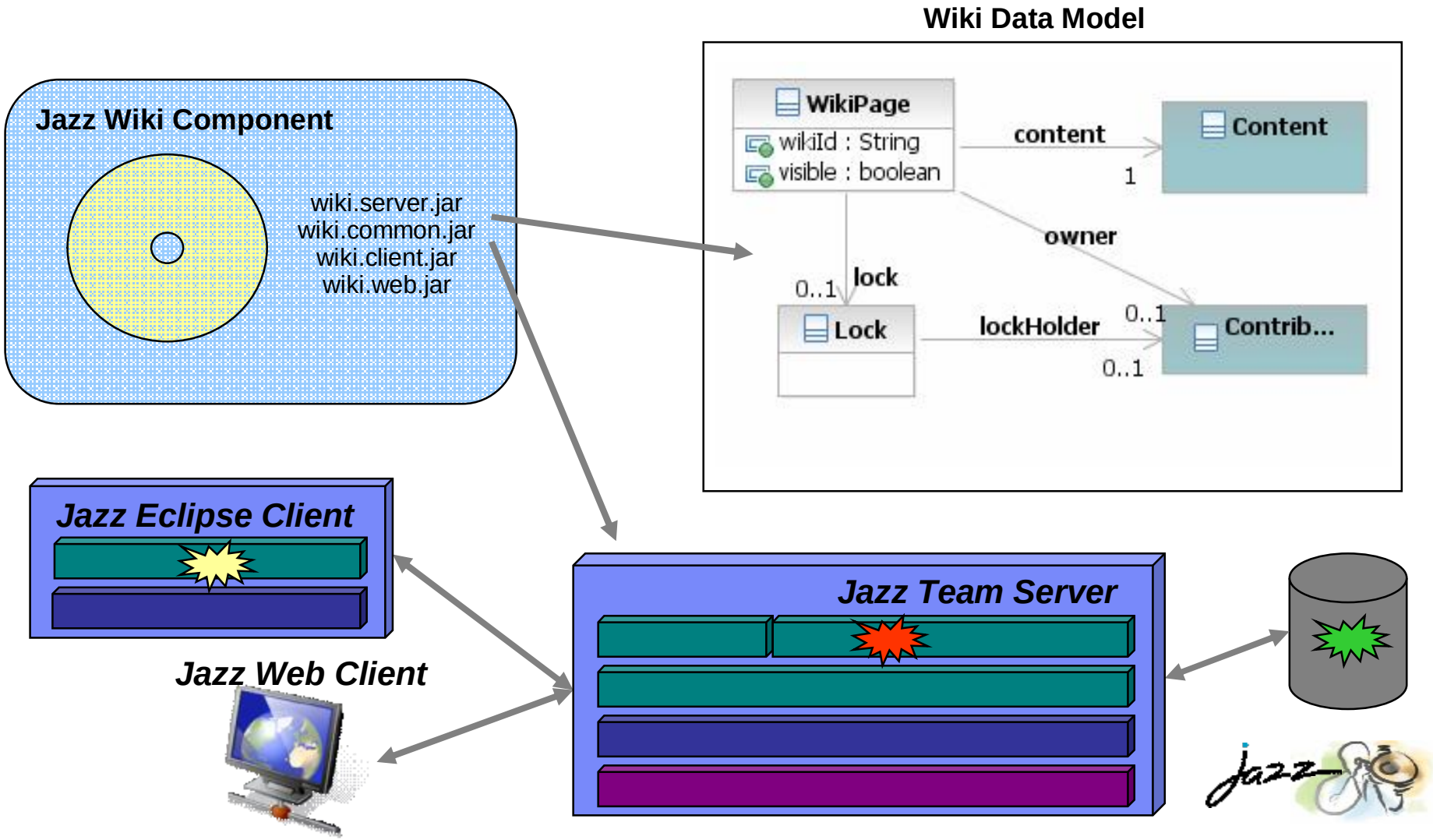


Extending the Jazz Platform – repository data model …three DBs in one

- An open-ended item store
  - Stores structured items as XML
  - Stores arbitrary content as BLOBs
- A declarative indexing story to support query
- A snapshot framework for extracting summary data for reporting



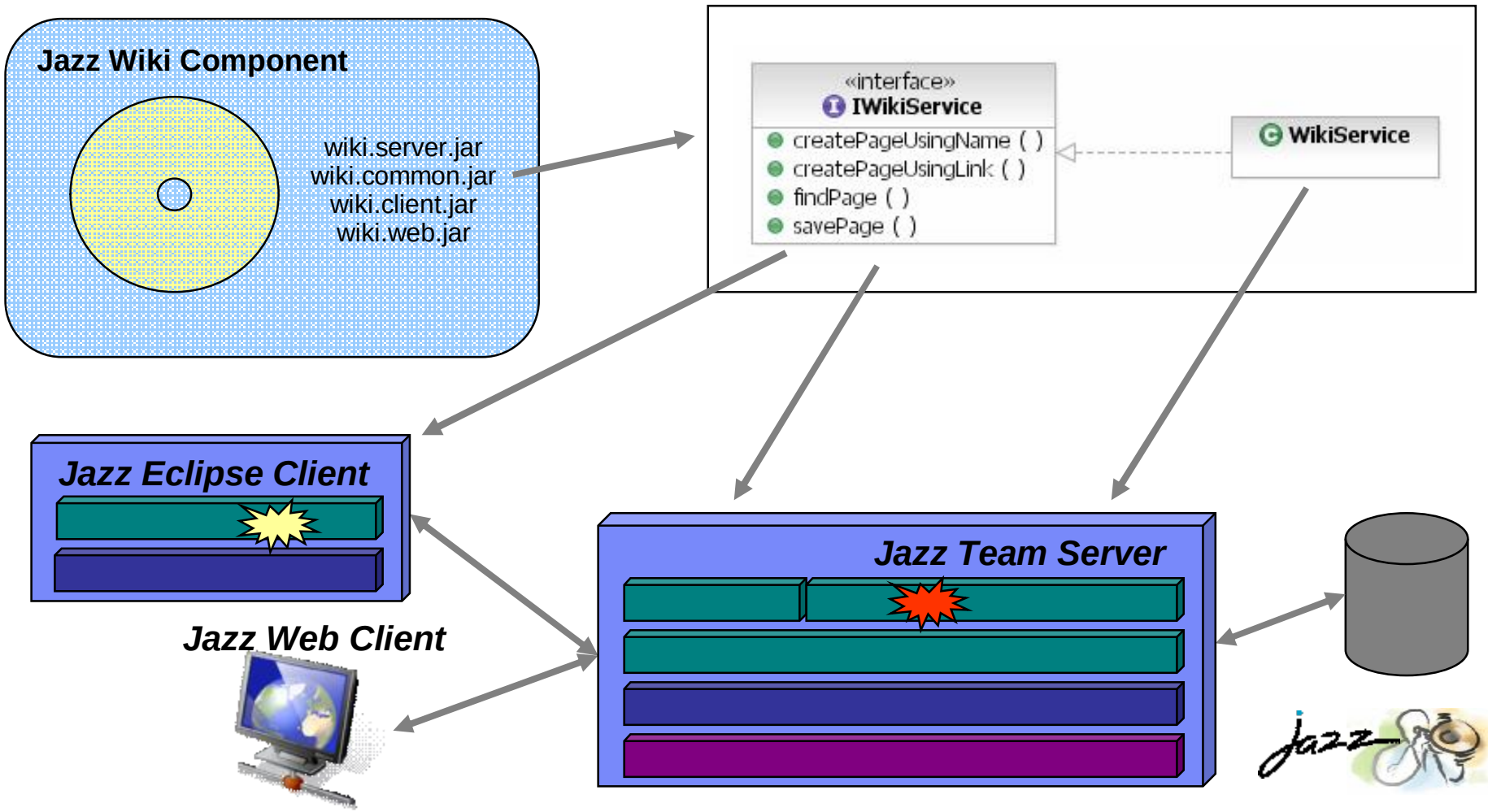
Extending Jazz – adding a component data model



Extending the Jazz Platform – Adding a component data model (plugin.xml)

```
<extension point="com.ibm.team.repository.common.components">
  <component
        id= "com.ibm.team.foundation.wiki"
        name= "Wiki"
        packageURI= "com.ibm.team.foundation.wiki">
        <service
              name="Wiki RCP Service“
              interfaceClass="com.ibm.team.wiki.IWikiService"/>
</component>
</extension>
```

Extending the Jazz Platform – Adding a service



Extending the Jazz Platform – Adding a service (plugin.xml)

```
<extension point="com.ibm.team.repository.common.components">
  <component
        id= "com.ibm.team.foundation.wiki"
        name= "Wiki"
        packageURI= "com.ibm.team.foundation.wiki">
        <service
              name="Wiki RCP Service“
              interfaceClass="com.ibm.team.wiki.IWikiService"/>
  </component>
</extension>
```

Extending the Jazz Platform – Adding a service (Java code)

Interface:

```
public interface IWikiService {

  public IWikiPage createPageUsingName(String name, IItemHandle
  defaultOwner) throws TeamRepositoryException;

  public IWikiPage createPageUsingLink(String wikiLink) throws
  TeamRepositoryException;

  public IWikiPage findPage(String wikiLink, IItemHandle
  defaultOwner) throws TeamRepositoryException;

  public IWikiPage savePage(IWikiPage page) throws
  TeamRepositoryException;

}
```

Implementation:

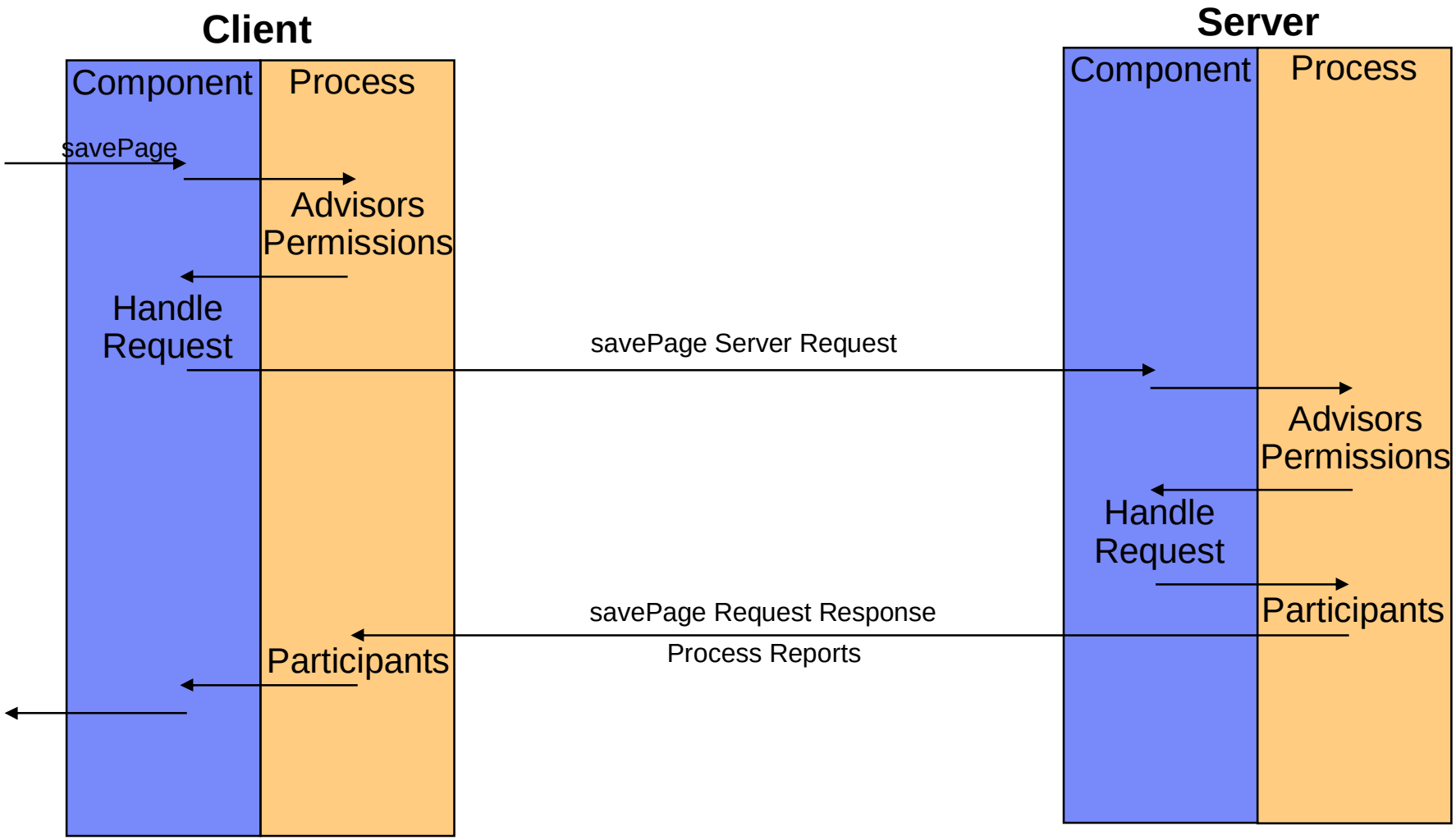
```
public class WikiService extends AbstractService implements
  IWikiService {

  …
```

Configurability through Process

- Allows team specific work products and behavior
- Allows team specific conditions under which client-side and server-side operations are allowed to proceed
- Allows team specific participants for client-side and server-side operations
- Allows team specific reactions to server-side change events

- Components explicitly design the configuration scope for process

Process Execution: Operations



Example: Server side process-enabled operation

§ Declare process-enabled save operation on the server

```
<extension
      point="com.ibm.team.process.service.configurationPoints">
  <operation
       id="com.ibm.team.wiki.server.savePage"
       name="Save Wiki Page">
  </operation>
</extension>
```

Example: Server side process-enabled operation

- When saving a page, wrap the save in an operation and pass the operation to the governing process for execution:

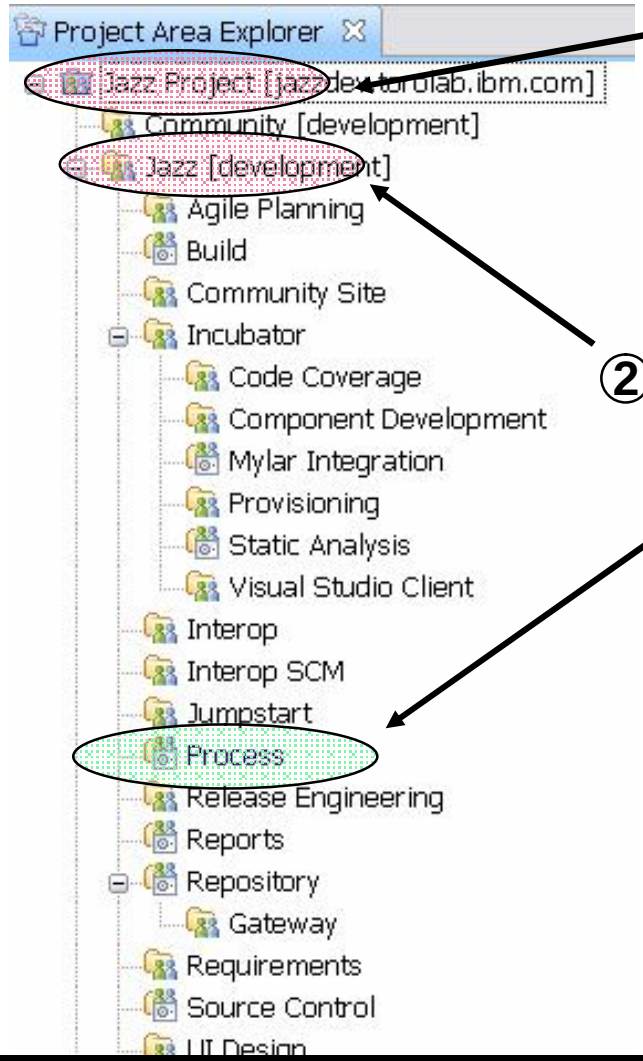
```
public IItemsResponse savePage(IWikiPage page) throws TRE {
 ITeamArea area = getTeamArea(page);
 IServerProcess serverProcess = getServerProcess(area);
 IDevelopmentLine line = serverProcess.getDevelopmentLine(area);

 WikiSaveOperation op = new WikiSaveOperation(page, area, line);
 IOperationReport report = serverProcess.adviseAndExecute(op);

 IItemsResponse response = ProcessCommon.createItemsResponse();
 response.setOperationReport(report);
 response.setClientItems(new IWikiPage[] { op.savedPage });
 return response;
}
```

Example: Server side process-enabled operation

§ The save operation defined inside the service looks like:

```
class WikiSaveOperation extends AdvisableOperation {

 public IWikiPage savedPage;

 public WikiSaveOperation(
    IWikiPage page, ITeamArea area, IDevelopmentLine line) {

  super("com.ibm.team.wiki.server.savePage", page, area, line);
 }

 public IOperationReport run(
    IProcessConfiguration conf, IProgressMonitor m)
    throws TeamOperationCanceledException {

  savedPage = transactionalSave((IWikiPage) getOperationData(), m);
  return null;
 }
}
```

Finding the Right Process



① **Jazz Project** area defines process:
No advisors for wiki savePage defined:

```
<operation id="com.ibm.team.wiki.server.savePage">
<permissions>
  <action id="any"/>
 </permissions>
</operation>
```

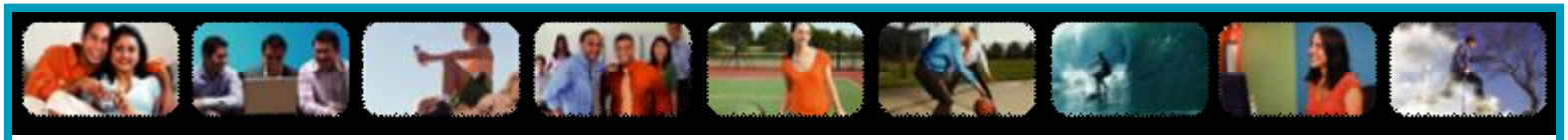
② **Jazz** team area does not customize the process

③ **Process** team area customizes the process defining advisors for wiki savePage:

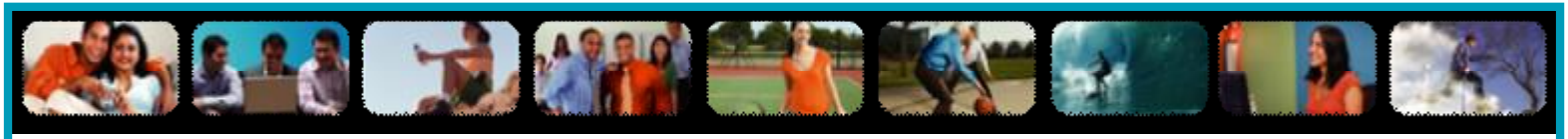
```
<operation id="com.ibm.team.wiki.server.savePage">
 <advisors>
  <advisor id="com.ibm.team.wiki.contentCheck"
    name="Check Authored Contents"
    enforcement="strict"
    description="The team believes in keeping the wiki in a s
  </advisor>
 </advisors>
 <permissions>
  <action id="any"/>
 </permissions>
</operation>
```

Programming Model Recap

- Client APIs and RCP integration
  - Lots of interesting extensions are possible with existing services and data
- Services
  - You own item saves, provide good validation and process enablement
  - REST services to support Web client and forward compatibility
    - Avoid exposing data directly
- Storage model
  - Think about migration
  - Use Content for open-ended text and unstructured content



Questions



Thank You