Ein Blick auf die Eclipse Plattform

Eine Präsentation von
Karsten Panier
und
Thomas Nawrath

Inhalt

- Motivation
- Was ist Eclipse?
- Architektur
- Eclipse Projekte
  - EMF
  - ALF
- Eclipse für Softwarehersteller
- Der Eclipse Entwicklungsprozess
- Die Plug-In Schnittstelle & RCP
- Success Storys
- Bewertung & Fazit

Motivation

- Interesse durch täglichen Umgang
- Berufliche Zukunft
- Synergieeffekte durch Recherche für Studium, Beruf und Arbeitskreis Objekt-orientierung
- Eclipse Plattfrom verbreiten
  - Mehr integrierte Anwendungen
  - Größere Community
- Eclipse ist mehr als eine IDE

Historie

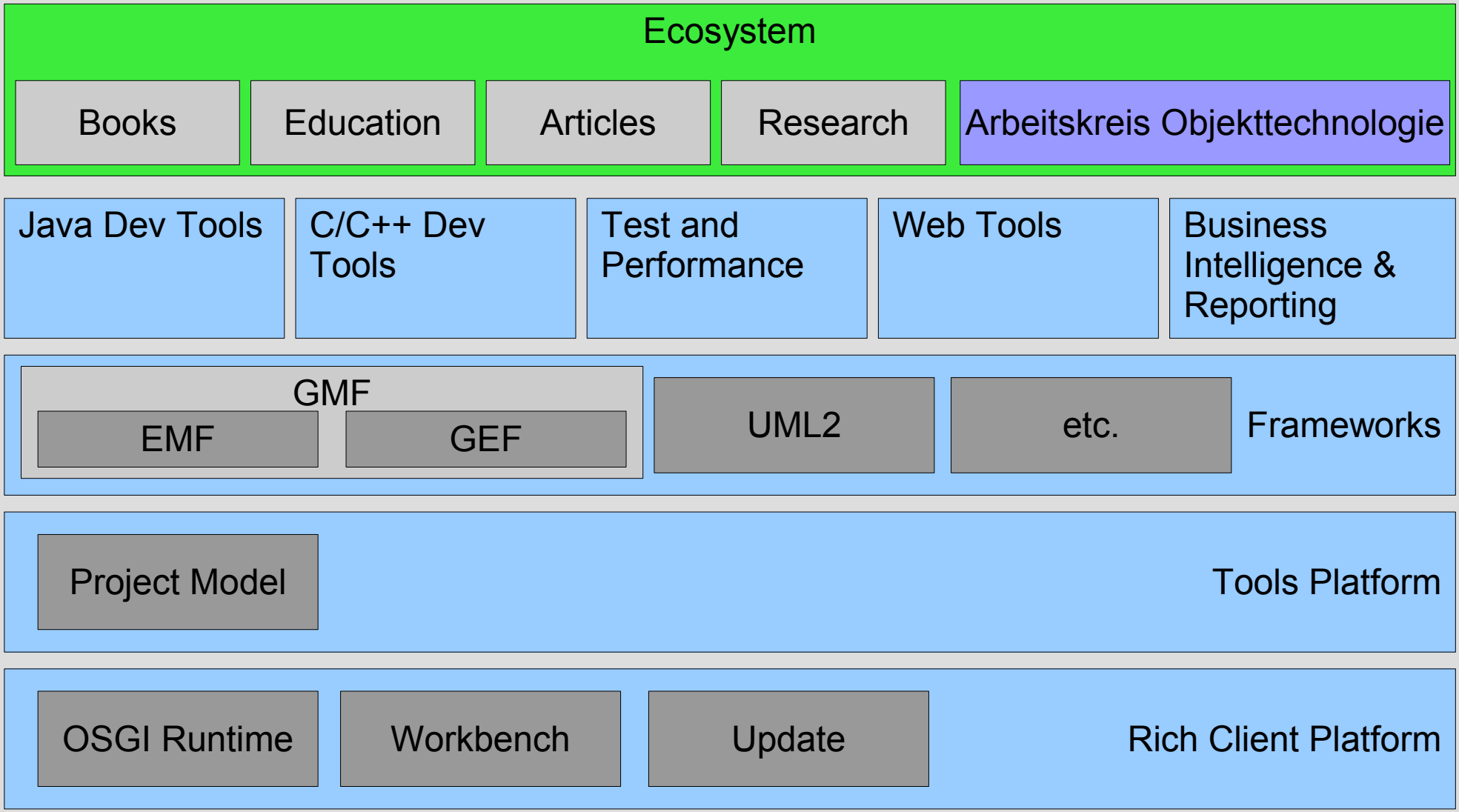
- 07.11.2001 Eclipse wird Open Source
  - IBM übergibt Source Code im Wert von 40 Millionen Dollar
- 28.06.2002 Eclipse 2.0 released
- 02.02.2004 Eclipse wird unabhängig
  - IBM überlässt der Eclipse Foundation die Leitung
- 21.06.2004 Eclipse RCP released
- Sommer 2005
  - Web Tools 0.7
  - TPTP
  - etc...

Was ist Eclipse

- Open Source Community
- Mit dem Ziel:
  - Softwareentwicklung zu verbessern
    - Lifecycle Werkzeuge
    - Frameworks
- Ist ein Ecosystem bestehend aus:
  - Technologieherstellern
  - Universitäten / Forschungsinstituten
  - Jedem Interessierten

⇨Eclipse hat die Akzeptanz um Standards zu schaffen

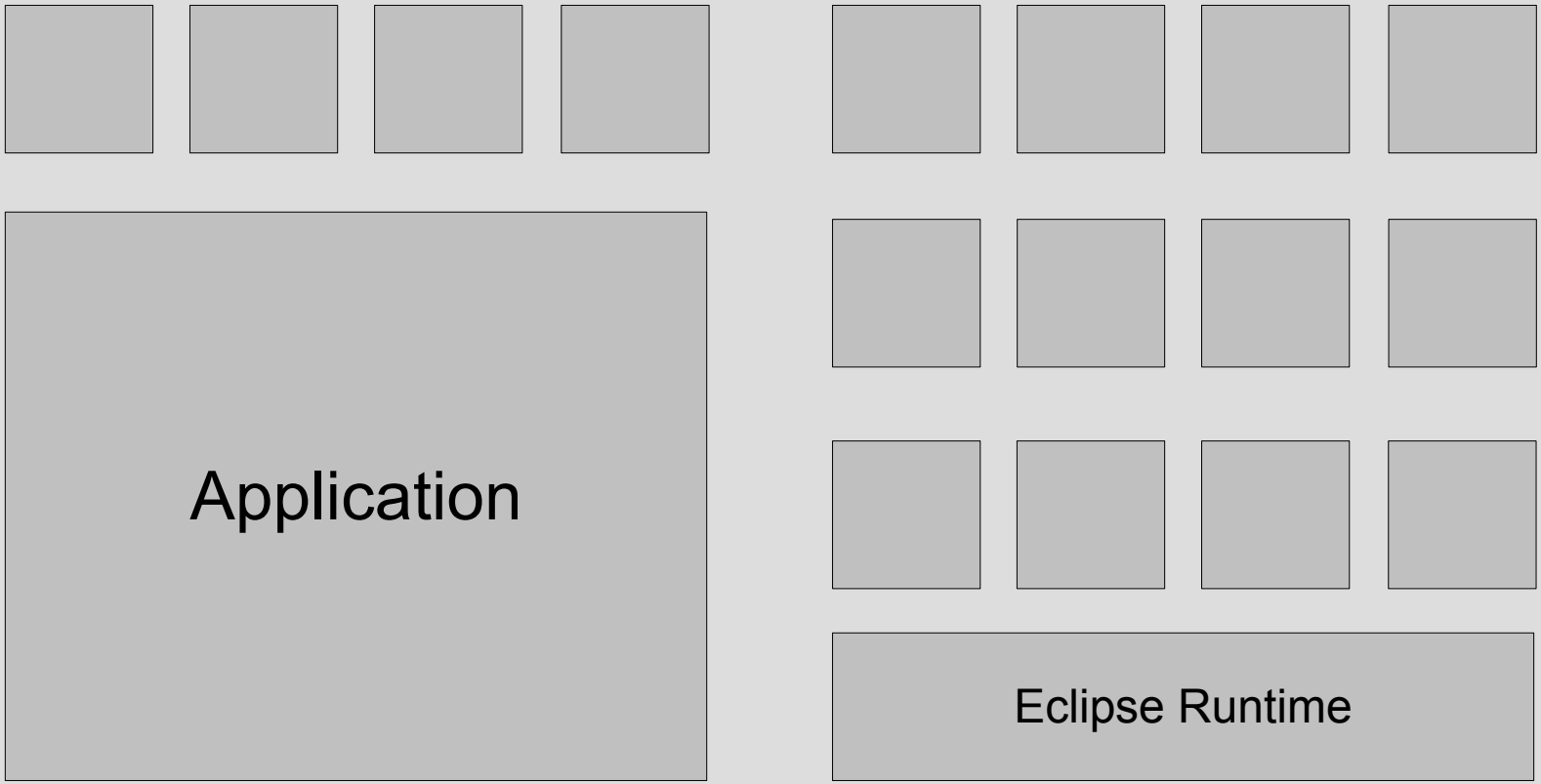
Architektur I



Architektur II

- Flexibel
- Abhängig von der Sichtweise und verwendeten Komponenten
- Komponenten Architektur
  - Plug-Ins
  - Features
- Open Service Gateway Initiative (OSGI) konform seit Eclipse 3.0
- SOA für weitere Projekte

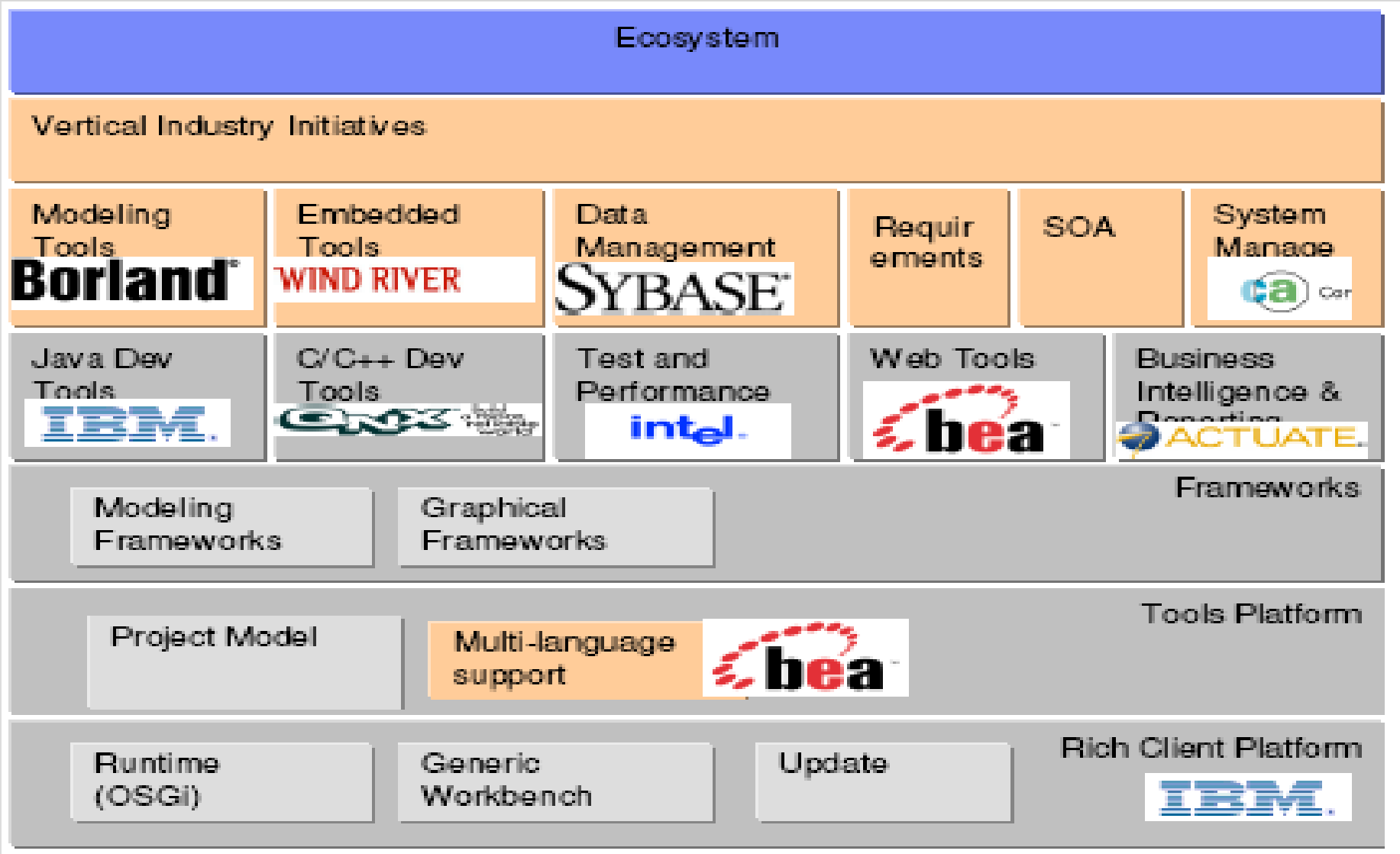
Architektur III (Plug-Ins)



Anwendung mit Plug-Ins

Eclipse

Architektur Ausblick



Quelle: IX-Konferenz 2005, Keynote Mike Milinkovich

Eclipse Projekte

- Eclipse ist in Projekten organisiert
  - Eclipse Management Organisation
    - 8 TOP Level Projekte
    - Sub Projekte
- Lebenszyklus eines Projektes
  - Initiieren ⇨ Project Proposal
  - Review / Projekt in die Projektstruktur einbinden
  - Implementation
    - Zyklisch
    - Reviews mit Abgleich der Eclipse Roadmap
- Organisation
  - Leitung durch Project Management Committee
  - Committer

Eclipse Projekte

- Eclipse Project
  - Equinox OSGI Framework
  - Platform
  - RCP
  - JDT/PDE
- Eclipse Tools Project
  - C/C++, Cobol, UML2
  - EMF, GEF, VE
- Eclipse Technology Project
  - Enthält Projekte folgender Rubriken
    - Incubator
    - Research
    - Education

Eclipse Projekte

- Eclipse Web Tools Platform Project
  - Web Standard Tools
  - J2EE Standard Tools (JavaServer Faces Tools)
- Eclipse Test and Performance Tools Platform Project
  - TPTP Platform
  - Monitoring Tools
  - Testing Tools
  - Tracing und Profiling Tools

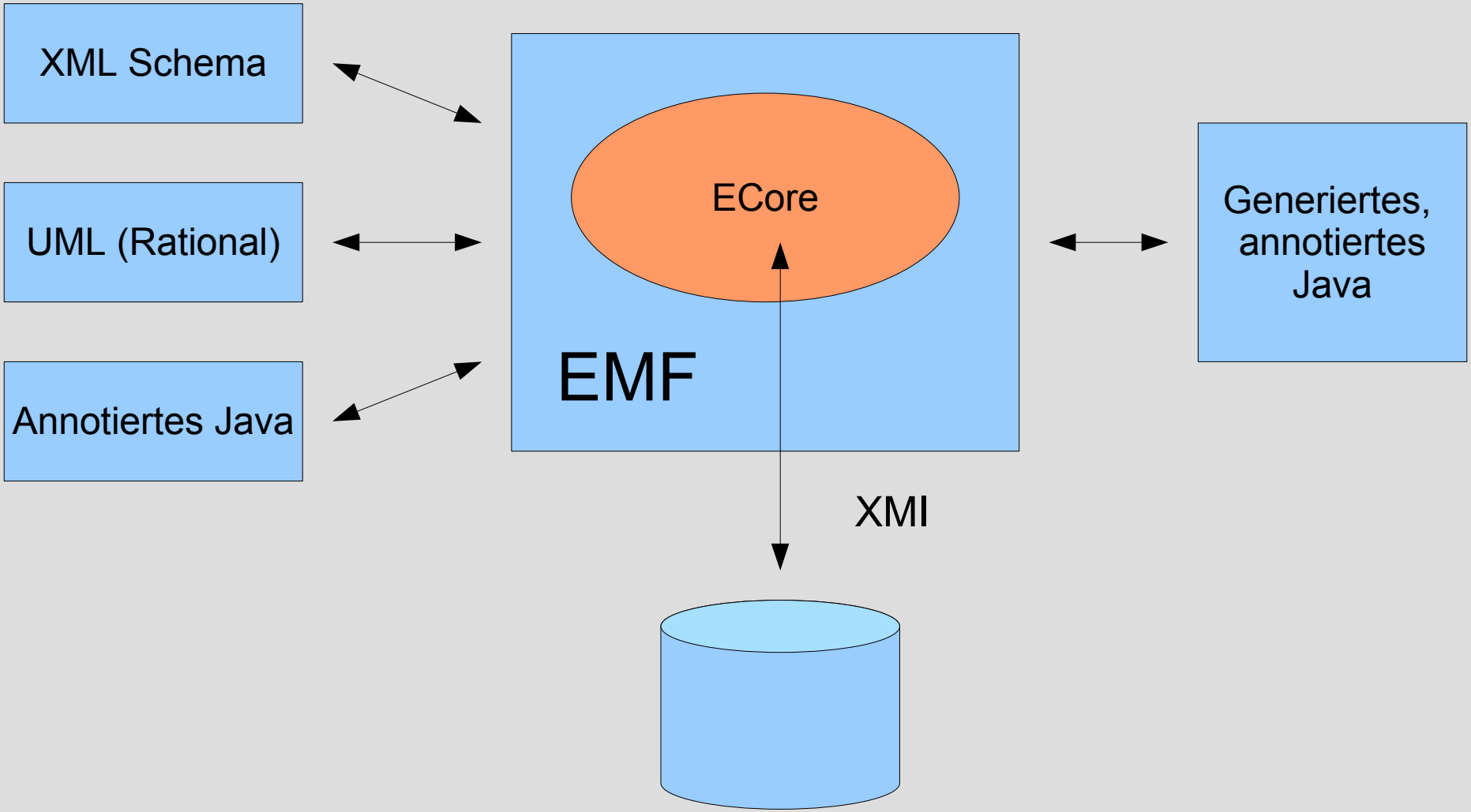
Eclipse Projekte

- Business Intelligence and Reporting Tools Project
  - Designer
  - Report Engine
- Data Tools Platform Project
  - Connectivity
  - Model Base
  - SQL Development Tools
- Device Software Development Platform
  - Target Management
  - Device Debugging

Das Eclipse Modeling Framework

- EMF ist ein Framework und eine Codegeneratoreinheit für Applikationen, die auf einem strukturierten Datenmodell basieren.
- EMF verbindet Java, XML und UML
- Ziel: „To model or program, that is not the question.“
- „You can think of EMF as MDA on training wheels.“

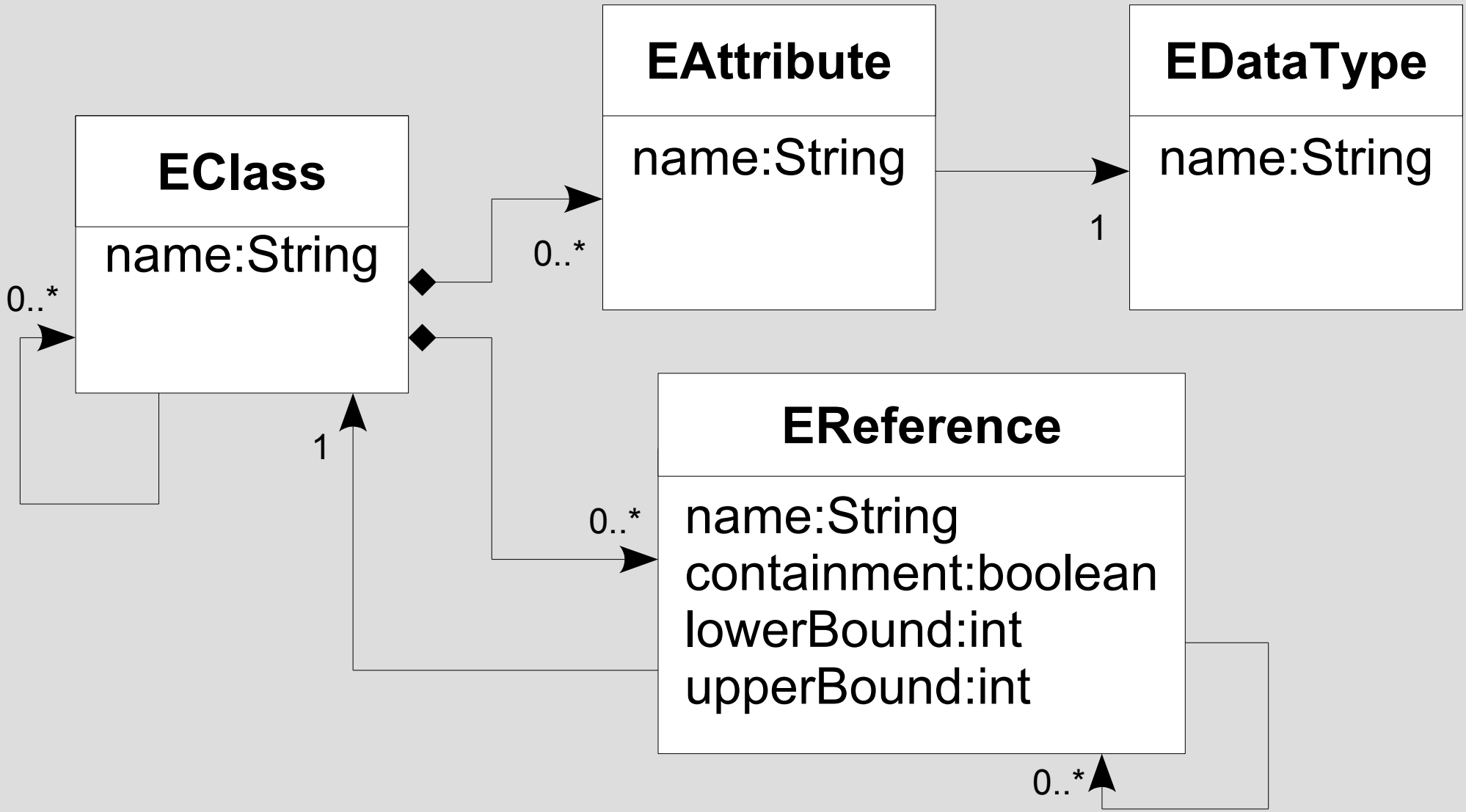
EMF Architektur



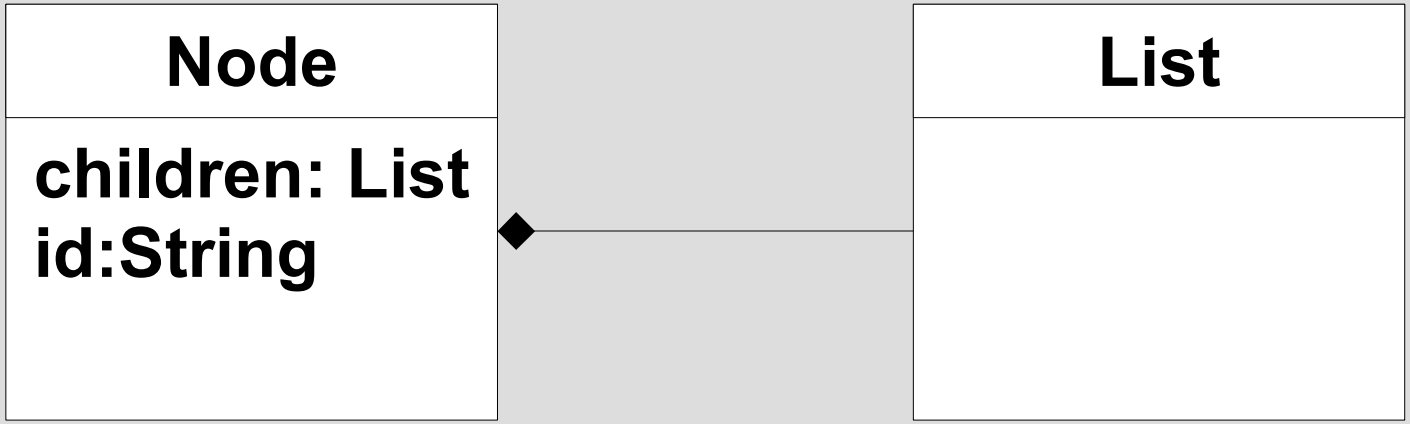
EMF Bestandteile

- EMF Core Framework
  - Metamodell (Ecore)
- EMF.Edit
  - Generische Klassen zum Bau von EMF Editoren
- EMF.Codegen
  - Codegeneratoreinheit, die das JDT verwendet. Es enthält:
    - Java Interfaces und Implementationen für die Klassen des Modells.
    - Adapter (ItemProviders) adaptieren die Klassen des Modells zur Anzeige und Bearbeitung.
    - Editor – Kommando basierter Editor für die Klassen des Modells.

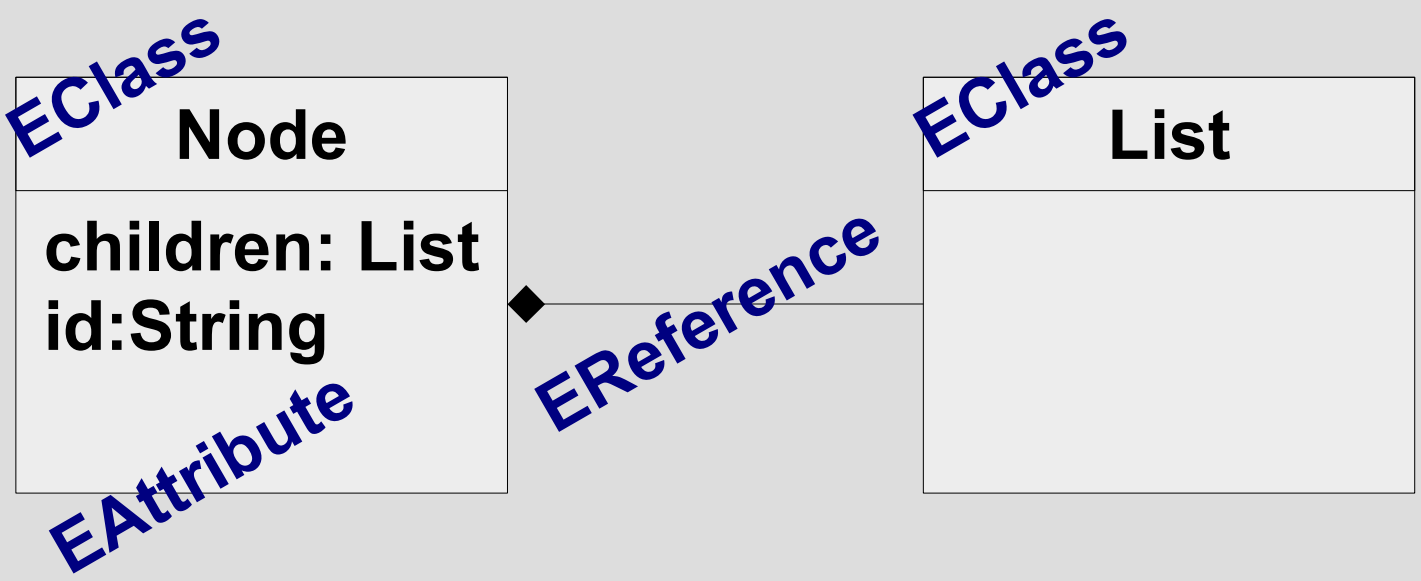
ECore Metamodell



ECore Beispiel



ECore Beispiel



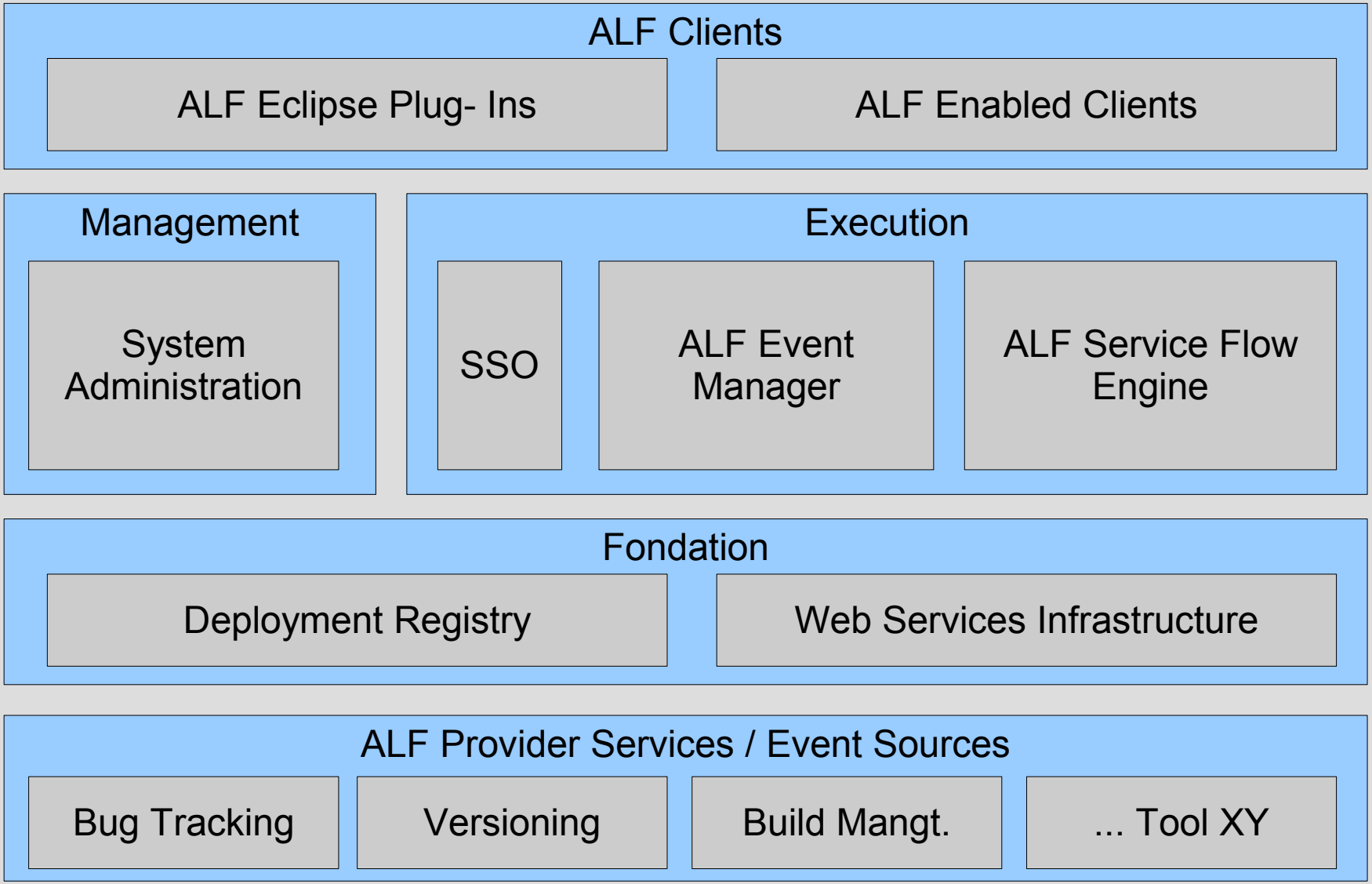
Eclipse ALF Projekt

- Application Lifecycle Management
  - Unter Eclipse Technology (Inkubator Status)
- Wer steht dahinter
  - Hersteller
    - BuildForge
    - Cognizant Technology Solutions
    - Secure Software
    - Segue
    - Serena Software
  - Kunde
    - UBS

ALF Ziele

- Integration der am Lebenszyklus beteiligten Systeme
- Infrastruktur zur Kommunikation der Systeme untereinander
  - EventManager
- Prozessdesigner um Systeme zu vernetzen
  - ALFDesigner

ALF Architektur



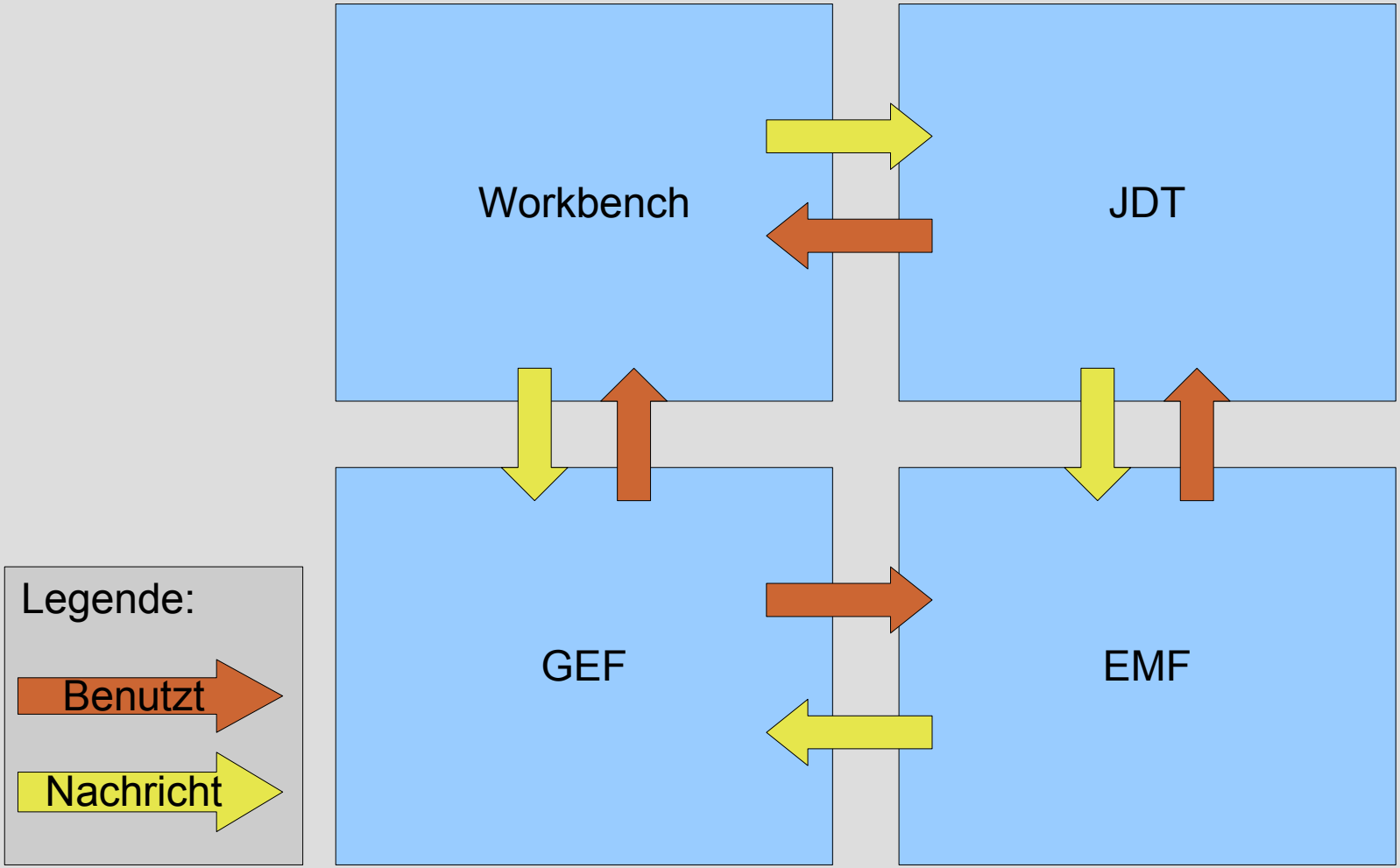
Eclipse für Softwarehersteller

- Standards etablieren
- Integrationsplattform
- Eclipse als Vertriebsweg
  - IBM (WSAD/RAD)
  - Birt Project QuellCode von der Firma Actuate
    - Support für Birt
    - Produkte basierend auf Birt
  - JSR220 Versants Open Access

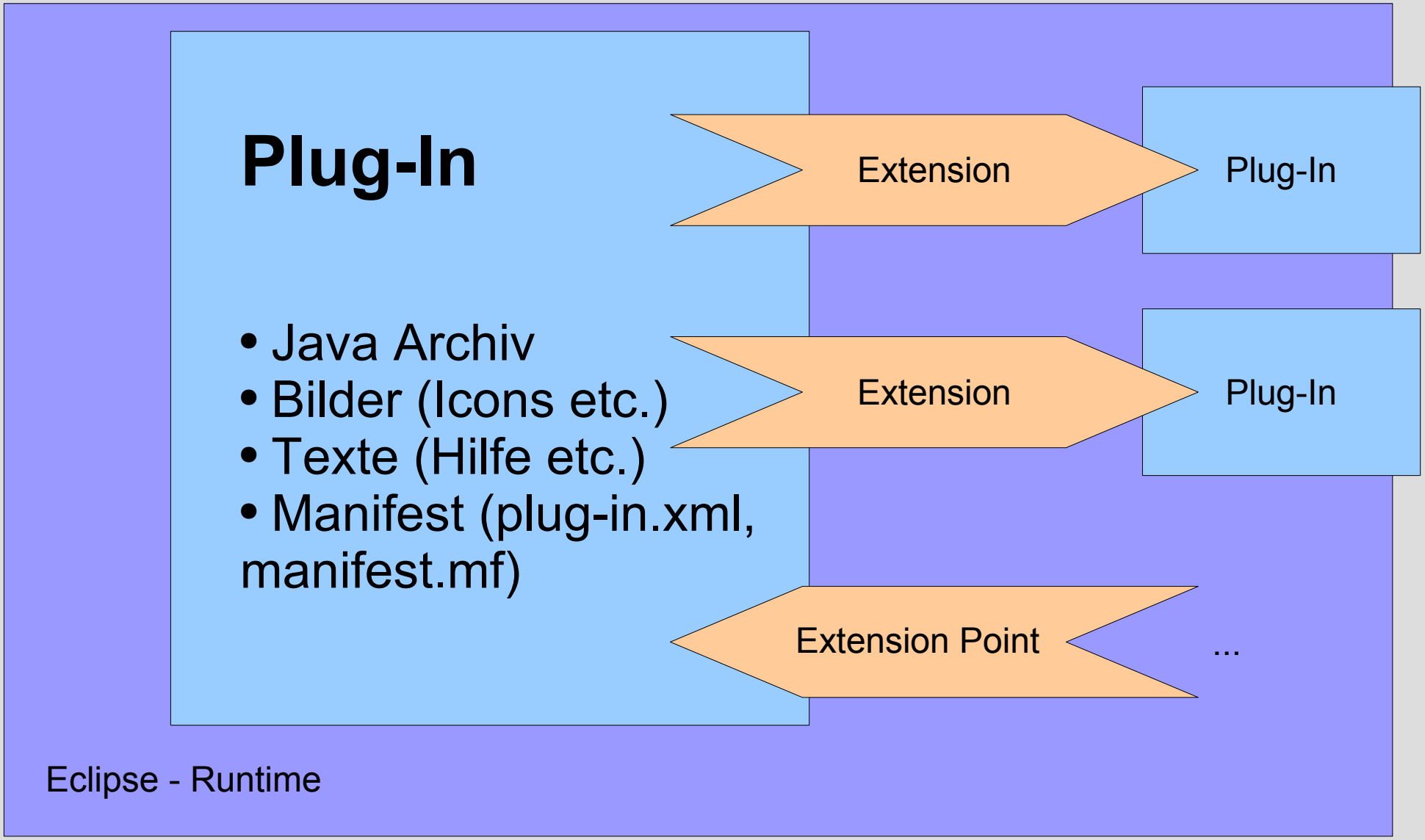
Eclipse Entwicklungsprozess

- Agil
  - Iterativ mit Milestone
  - Community beeinflusst die Planung
- Transparenz
  - Nigthly Build
    - Unit Tests
    - Performance Tests
  - Anforderungen
  - Reports über die Ergebnisse
    - Buildresults
    - Dashboard mit Bugs
- Eat your own Dog Food

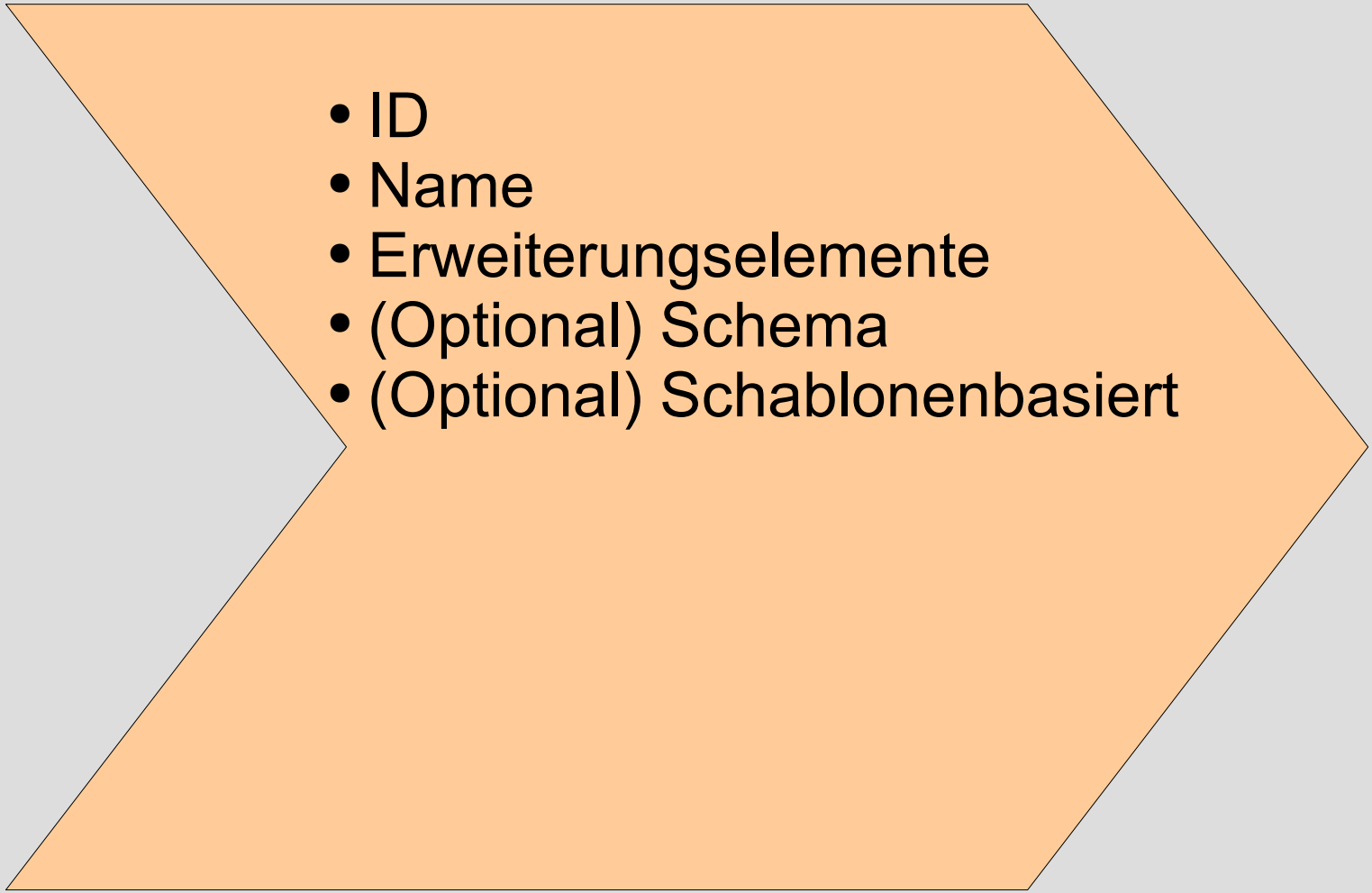
Plug-In Schnittstelle



Plug-In Schnittstelle



Extension Point



Extension Point Schema

- Definiert wie ein Extension Point zu verwenden ist.
- Attribute
  - Java
    - Klassenname
    - (Optional) BasedOn: Name einer abstrakten Klasse oder eines Interfaces um Methodenrümpfe und Klassen vor zu generieren
  - Ressource
    - Pfad auf externe Ressourcen
  - String
    - Boolscher Wert, String (Auf Enumeration beschränkbar)

Extension Point Schablonen

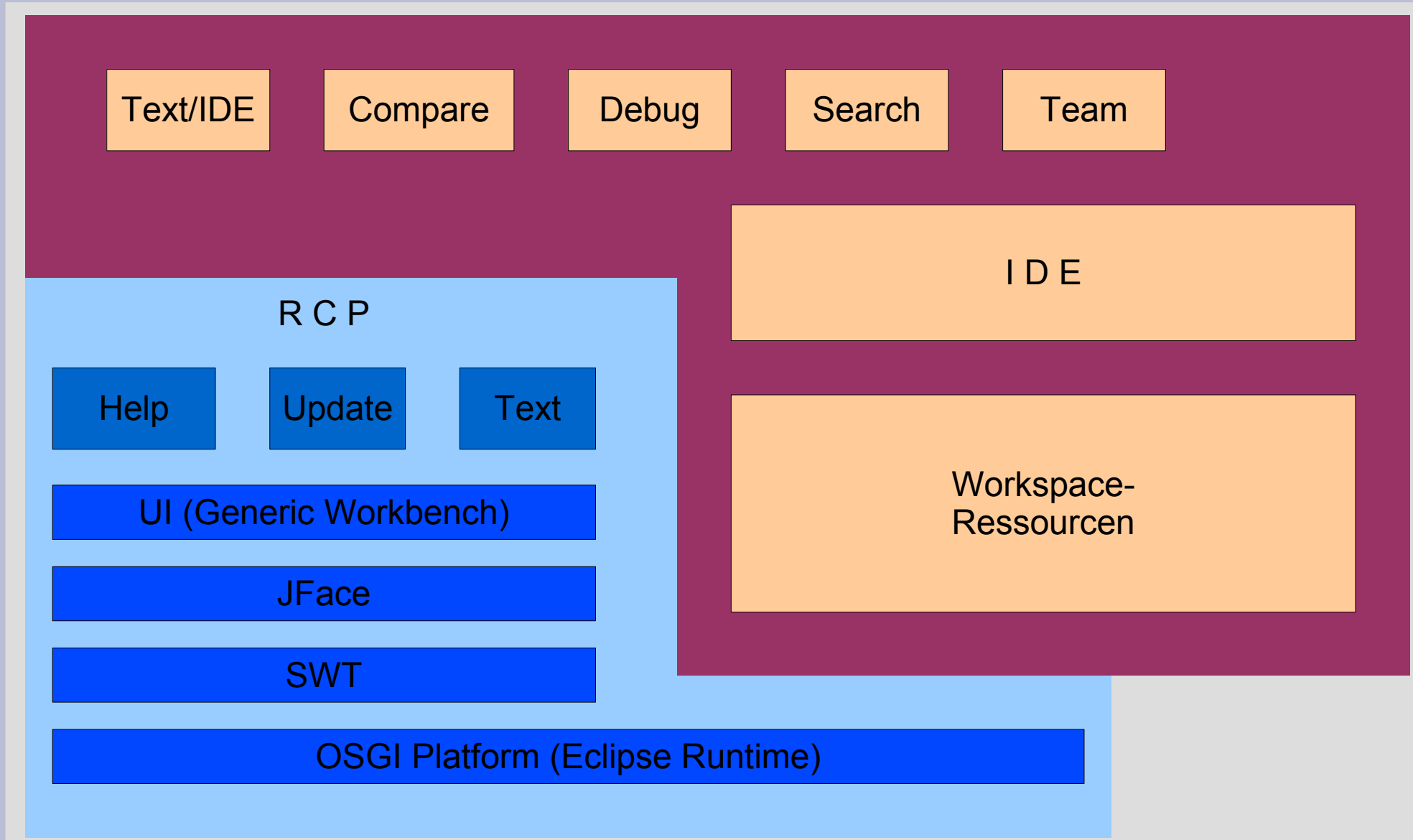
- Wizardgestütze Generatoren zur einfachen Verwendung von gängigen Extension Points.
- z.B. für
  - Editoren
  - Views
  - Wizards für neue Dateien
  - Pop-up Menus
  - Neue Perspektiven
  - Eigene Seiten für Einstellungen

Plug-In

Abstract Plug-In
<implements OSGi Interface>

- Singleton
- Lebenszyklus durch Runtime verwaltet
- Eigenen Classloader
- Access Rules

Abgrenzung RCP / IDE



Bausteine RCP

- SWT/JFace
  - Fortschrittsbalken
  - Dialoge (z.B.: Warning, Dateidialog)
- Generic Workbench
- Evtl. Textverarbeitung
- Evtl. Hilfesystem
- Application Plug-In
- Plug-In – Updatemechanismus via Internet
- Konfiguration und Zustand von Plug-Ins und Plattform

Bausteine IDE

- Workspace
  - Resource: eigene Dateisystemschicht
    - Projekte
    - Verzeichnisse
    - Dateien
  - Marker
  - Aufgaben
  - Ereignisbehandlung
- Team
  - Local History
- Debug
  - Breakpoints

Plug-In Rumpfprogramm

```
public class SamplePlugin extends AbstractUIPlugin {

   private static SamplePlugin plugin;

   public SamplePlugin() {..}

   public void start(BundleContext context)
         throws Exception {..}

   public void stop(BundleContext context)
         throws Exception {..}

   public static SamplePlugin getDefault() {..}

   public static ImageDescriptor
      getImageDescriptor(String path) {...}
}
```

Einfache Action

```
public class SampleAction implements IWorkbenchWindowActionDelegate {
    private IWorkbenchWindow window;

    public SampleAction() {
    }

    public void run(IAction action) {
        MessageDialog.openInformation(
            window.getShell(),
            "Sample Plug-in",
            "Hello, Eclipse world");
    }

    public void selectionChanged(IAction action, ISelection selection) {
    }

    public void dispose() {
    }

    public void init(IWorkbenchWindow window) {
        this.window = window;
    }
}
```

Plug-In Konfiguration

```
<?xml version="1.0" encoding="UTF-8"?>
<?eclipse version="3.0"?>
<plugin>
   <extension
         point="org.eclipse.ui.actionSets">
      <actionSet
            label="Sample Action Set"
            visible="true"
            id="Test.actionSet">
         <menu
               label="Sample &amp;amp;Menu"
               id="sampleMenu">
            <separator
                  name="sampleGroup">
            </separator>
         </menu>
         <action
               label="&amp;amp;Sample Action"
               icon="icons/sample.gif"
               class="test.actions.SampleAction"
               tooltip="Hello, Eclipse world"
               menubarPath="sampleMenu/sampleGroup"
               toolbarPath="sampleGroup"
               id="test.actions.SampleAction">
         </action>
      </actionSet>
   </extension>
</plugin>
```

Success Storys

- Erweiterungen von Eclipse Produkten
  - DB Edit
    - Erweiterung für Stored Procedures
    - Multiple Result Sets
    - Proprietäre Features
  - Eigene Informationen aufbereiten
    - CruiseControl
  - VPMS Modelmanager FTP Plug-In (1 Woche)
- RCP Anwendungen
  - DB Stage Compare (2 Wochen)
  - Viewer auf Anforderungsänderungen

Bewertung :-)

- Größte Stärken liegen im Desktopbereich
- Leicht erweiterbar
- Infrastruktur
  - Update Mechanismen
  - RCP Anwendungen
- Große Community
- Eclipses Verbreitung
- Hohe Qualität
- Flexibel und breit einsetzbar
- Musterprojekt von dem man lernen kann

Bewertung :-(

- Lernkurve
- Lifetime Learning durch Weiterentwicklung
  - Geänderte Bestpractices eines Frameworks
  - Neue Frameworks
- Verschiedene Versionen von Eclipse im Umlauf
- Redundanzen im Code
  - Von non public Code
  - Plug-In Teile kopieren

Fazit

- Eclipse ist cool
- Chance den Desktopbereich zu erobern
- Erfahrungsbereicherung um sehr gute Designbeispiele

Literatur

- www.eclipse.org
- Contributing to Eclipse Principles, Patterns and Plug-Ins (Erich Gamma, Kent Beck)
- Eclipse Modeling Framework (Frank Budinsky, David Steinberg, Ed Merks, Raymond Ellersick, Timothy J. Grose)
- Eclipse building Commercial-Quality Plug-Ins (Eric Clayberg, Dan Rubel)
- Java- Entwicklung mit Eclipse 3 (Berthold Daum)