

Eclipse Rich Client Platform Tutorial

Bernd Kolb
Martin Lippert

b.kolb@kolbware.de

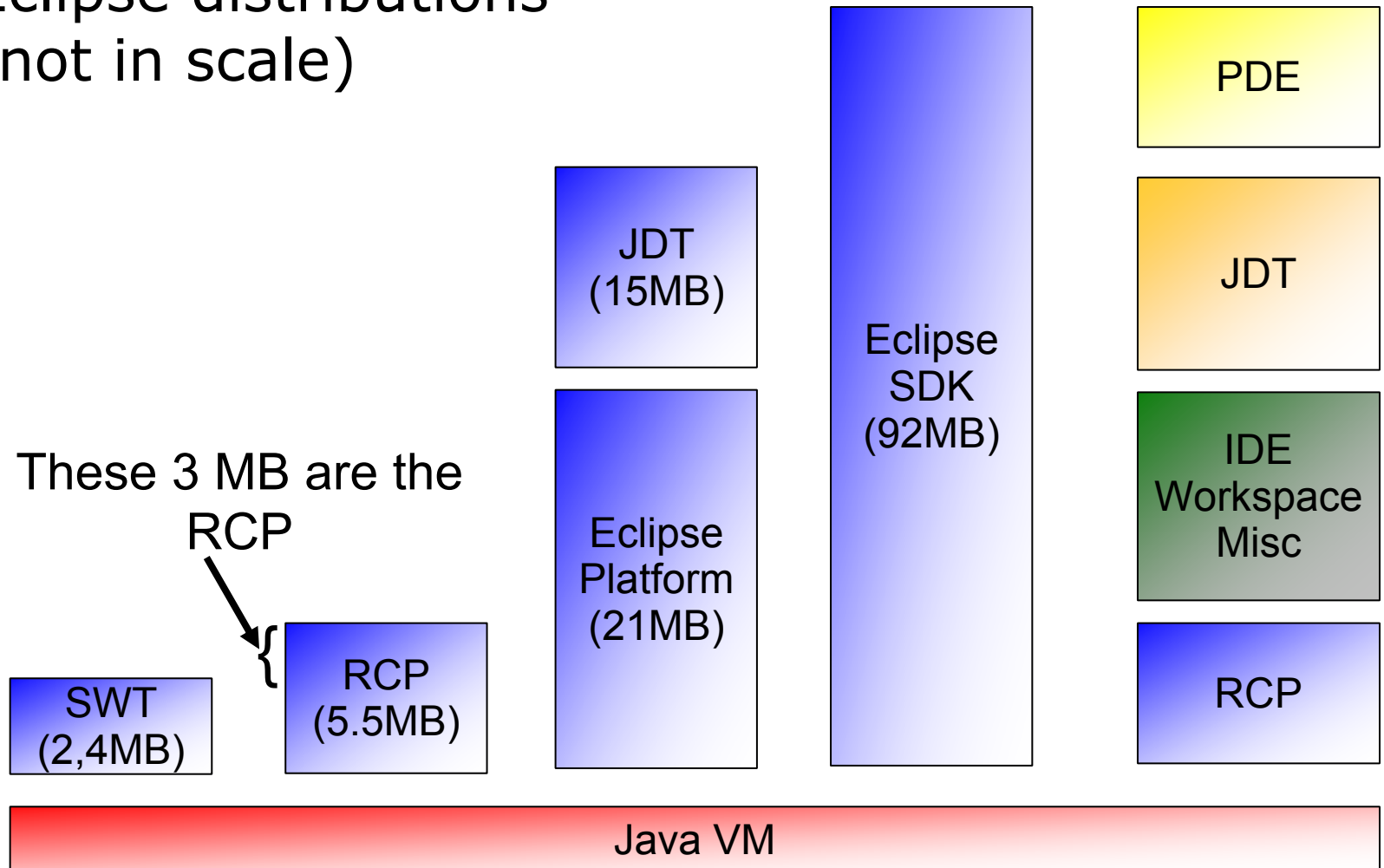
lippert@acm.org

Outline

- Architectural overview
- Building RCP App
 - Textual “Hello-World”
 - App with UI-Elements
 - Deployment
 - Using the update manager
 - Creating a feature
 - Creating a update site
 - Creating an extension point
 - Include help
 - Branding
 - Q&A

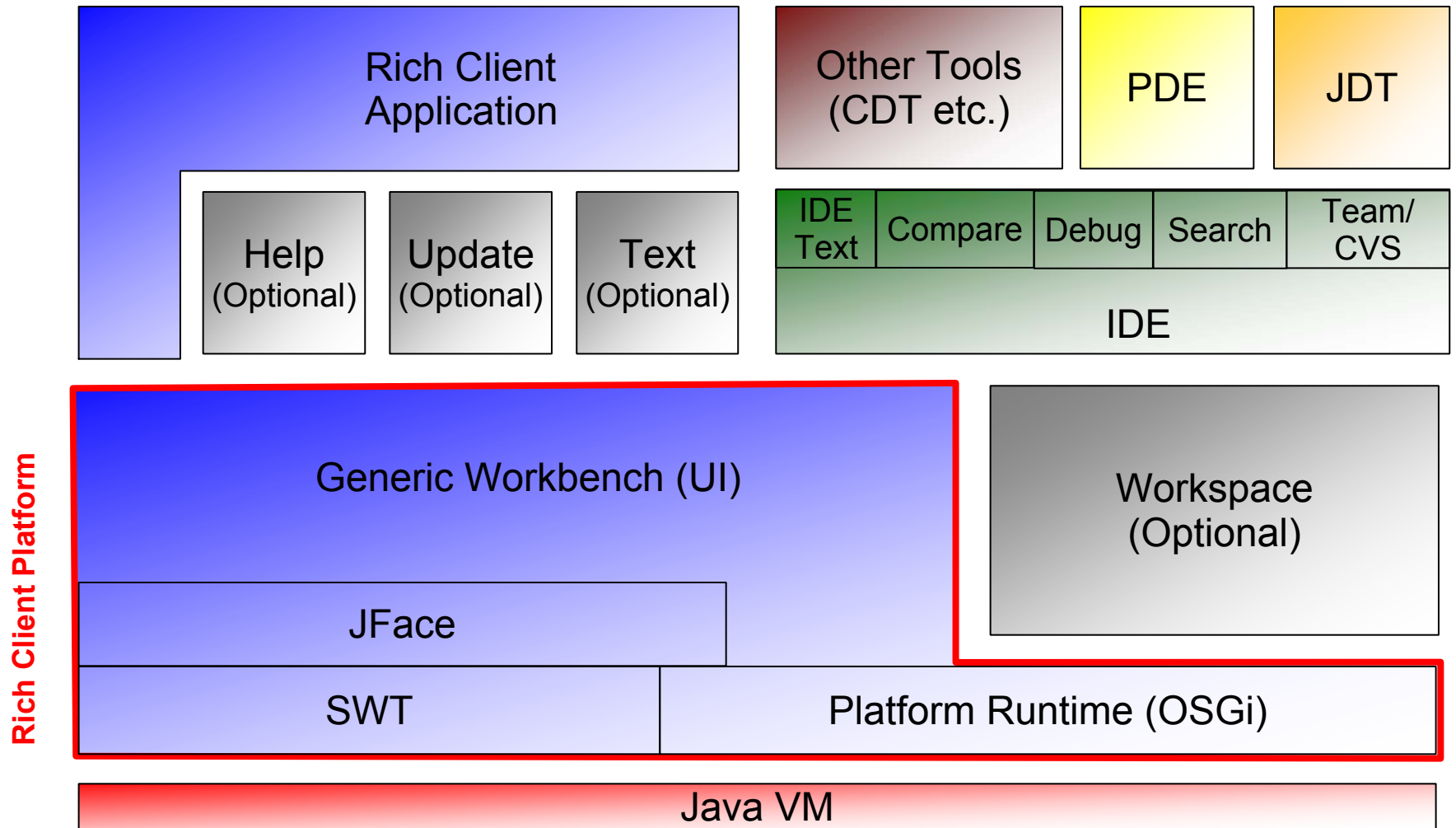
Architectural overview

- Eclipse distributions (not in scale)



Architectural overview

- Eclipse architecture

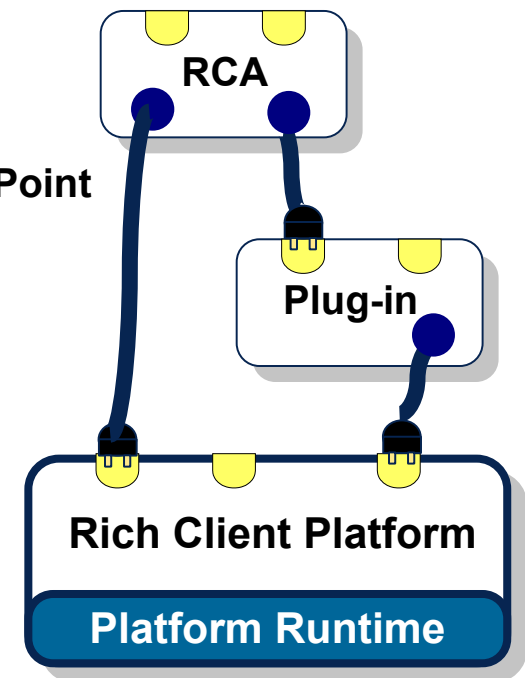


Architectural overview

- OSGi
 - Open Services Gateway Initiative
 - Is responsible for
 - Bundle administration
 - Classloading
 - Several platform services (e.g. dependency management)
 - Is NOT responsible for
 - Extensions / Extensionpoints

Architectural overview

- Plug-ins
 - The Eclipse synonym to a OSGi bundle
 - Declares the component model
 - Extensions
 - Extension points
- Platform runtime
 - Contains OSGi and the plug-in runtime
 - Is the runtime environment for bundles / plug-ins
 - Responsible for jobs and preferences

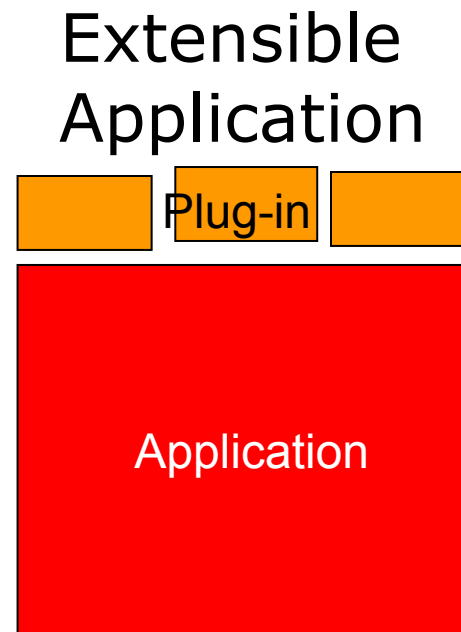
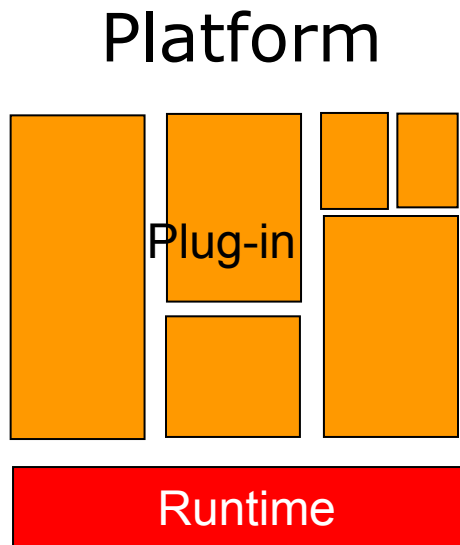


Architectural overview

- Buzzwords
 - Plug-in
 - A plug-in is a component. It has dependencies to other plug-ins, declares extension points and extensions
 - Fragment
 - Is a addition to a plug-in which adds e.g. I18N support or adds OS specific parts
 - Feature
 - Bundles a set of plug-ins with a specific version. Only a feature can be updated, not a single plug-in or fragment
 - Update-site
 - A location where features can be found and installed from

Architectural overview

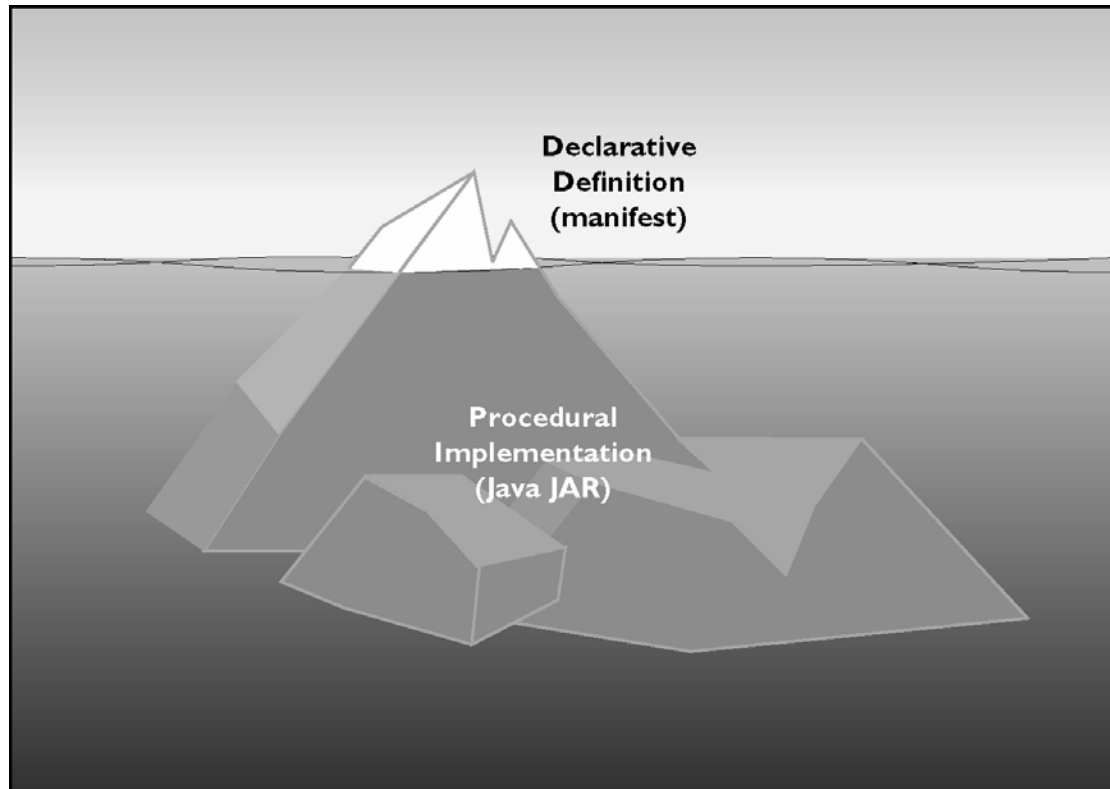
- Platform vs. extensible application
 - Eclipse is a platform. Thus it has a very small kernel



- Eclipse plug-in architecture
 - Due to the separation of declared and implemented parts, a lazy loading is possible
 - Declaration describes
 - The functionality of the plug-in
 - The UI contributions
 - Which classes implement the contributions
 - Implementation
 - Is done in Java and added as JAR to the plug-in
 - Will be loaded lazily if
 - An extension point is used which is used by this plug-in
 - Code of this plug-in is needed by an other plug-in

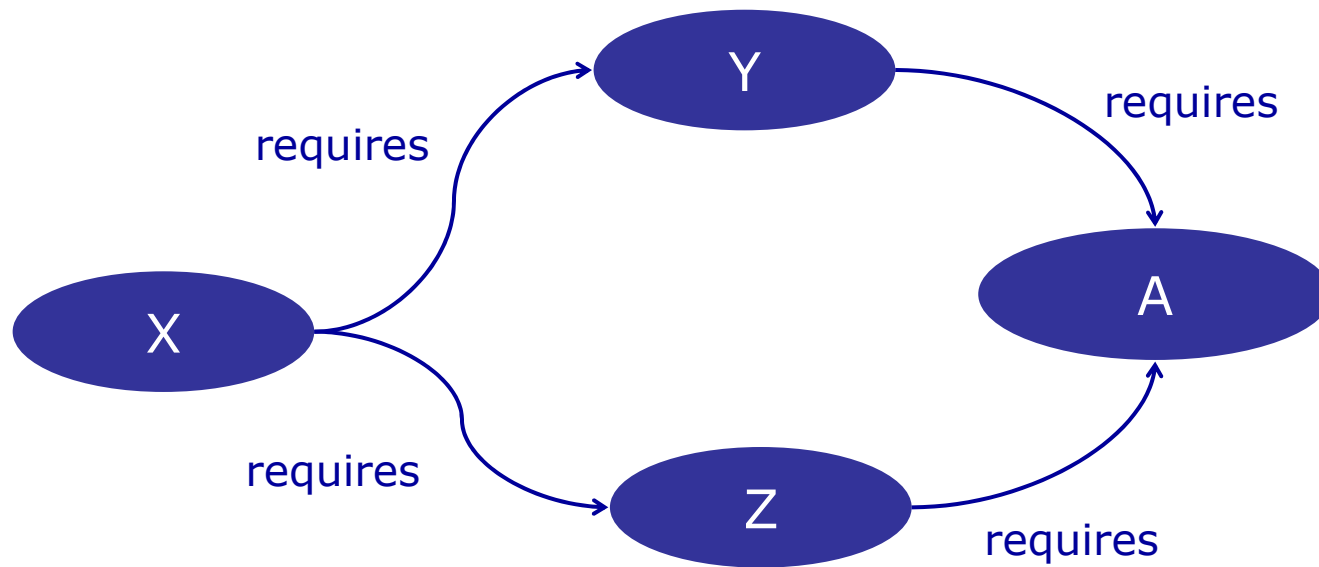
Architectural overview

- Tip of the iceberg
 - Startup time: $O(\# \text{used plug-ins})$, not $O(\# \text{installed plug-ins})$



Architectural overview

- Plug-in namespaces
 - Each plug-in has its own classloader
 - Requests are delegated to the responsible classloader



Architectural overview

- SWT
 - Native GUI-Widgets
 - Not OO (e.g. API-compliance between different OS only due to conventions)
- JFace
 - Abstraction over SWT
 - Viewer
 - Forms-API
 - Wizards / Dialogs / Actions
 - MVC / Command Pattern

Architectural overview

- Workbench
 - Contributes the empty window
 - Adds support for
 - Menu bars
 - Tool bars
 - Perspectives
 - Views / Editors
 - Preferences
 - And many more extension points

