BVM-Tutorial 2009: openCherry A modular, cross-platform, C++ application framework

Daniel Maleike, Michael Müller, Jochen Neuhaus,
Marco Nolden, Sascha Zelzer

DEUTSCHES
KREBSFORSCHUNGSZENTRUM
IN DER HELMHOLTZ-GEMEINSCHAFT

Motivation

MITK is a *toolkit*, but provided an application layer based on Qt3 (*MainApp*)

Issues with the Qt3 MainApp:

- Qt3 has been superseded by Qt4 a long time ago
- Fixed application layout
- The modul concept (Functionalities) allows only coarse modularity
- Not possible to add modules in binary form

→ Build a new, component-oriented application framework with a Qt4 frontend

Goals

- Provide a plug-in system based on OSGi
- Allow loose coupling of modules via „Extension-Points“ (lazy-loading)
- Enable binary distribution of plug-ins

- Provide a highly customizable (GUI)-application framework
- Note: Plug-ins can contain arbitrary code and are not only meant for GUI components

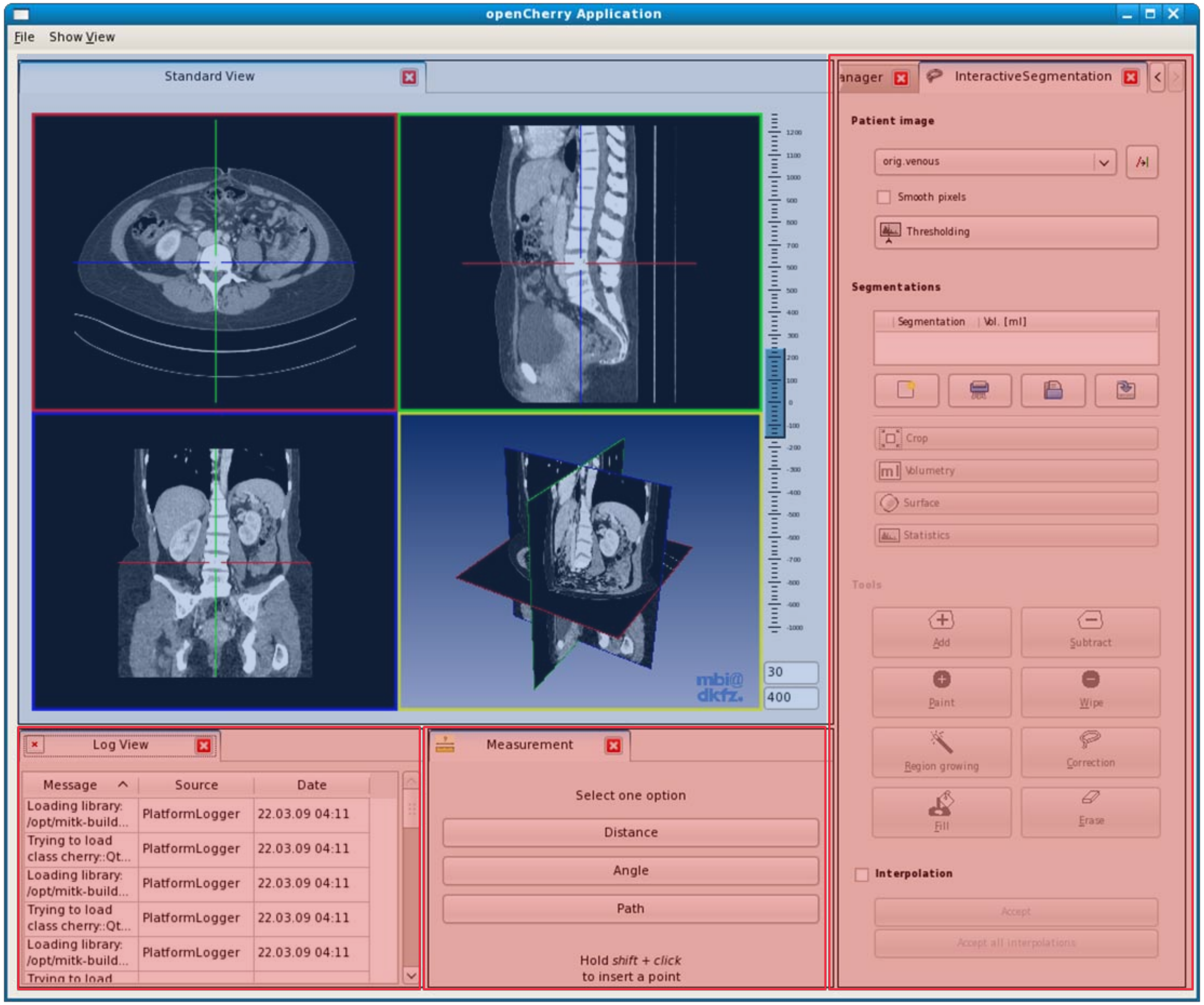
Sascha Zelzer
Medizinische und Biologische Informatik

The Workbench - Overview

Menu contributions

Editors

Views



Message	Source	Date
Loading library: /opt/mitk-build...	PlatformLogger	22.03.09 04:11
Trying to load class cherry::Qt...	PlatformLogger	22.03.09 04:11
Loading library: /opt/mitk-build...	PlatformLogger	22.03.09 04:11
Trying to load class cherry::Qt...	PlatformLogger	22.03.09 04:11
Loading library: /opt/mitk-build...	PlatformLogger	22.03.09 04:11
Trying to load		

Select one option

Distance

Angle

Path

Hold *shift + click* to insert a point

The Workbench

- You can add arbitrary views and editors to your (or others) application
- Define *perspectives*, a layout of views and editors designed for specific tasks
- Use the command framework (to be finished soon) to add menus and toolbar items to the application

The Plug-In System

- A plug-in can contain resources and/or code
- Need to supply meta information about a plug-in:

META-INF/MANIFEST.MF

```
Manifest-Version: 1.0
Bundle-Name: openCherry User Interface Plugin
Bundle-SymbolicName: org.opencherry.ui
Bundle-Version: 1.0.0
Bundle-Vendor: DKFZ, Medical and Biological
  Informatics
Require-Bundle: org.opencherry.osgi, …
Bundle-Activator: cherry::WorkbenchPlugin
```

- The bundle activator is a class for plug-in lifecycle management
- What about *loose* coupling?

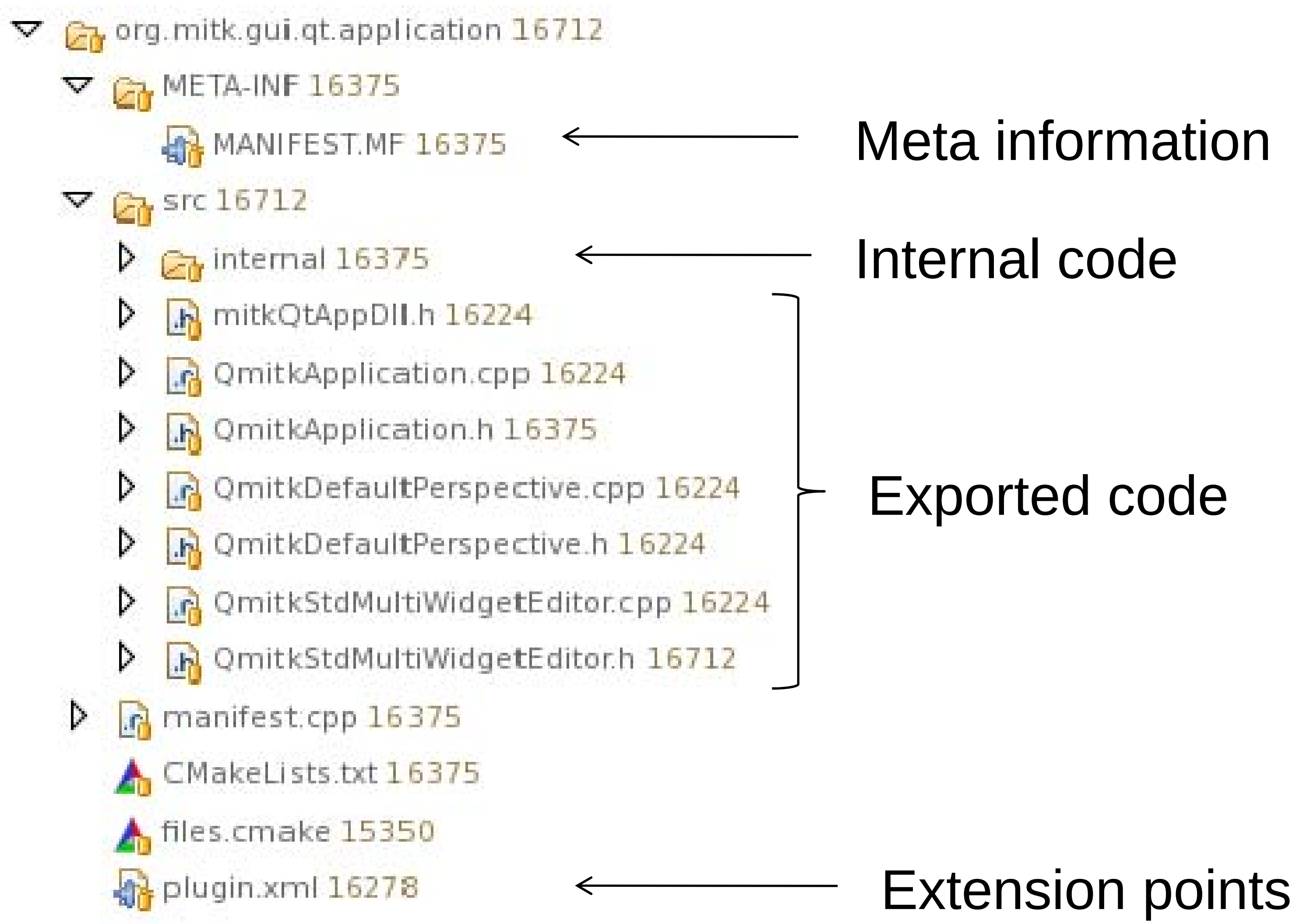
The Plug-In System

- Extension points can be used to provide or collect information without loading the plug-in.

plugin.xml

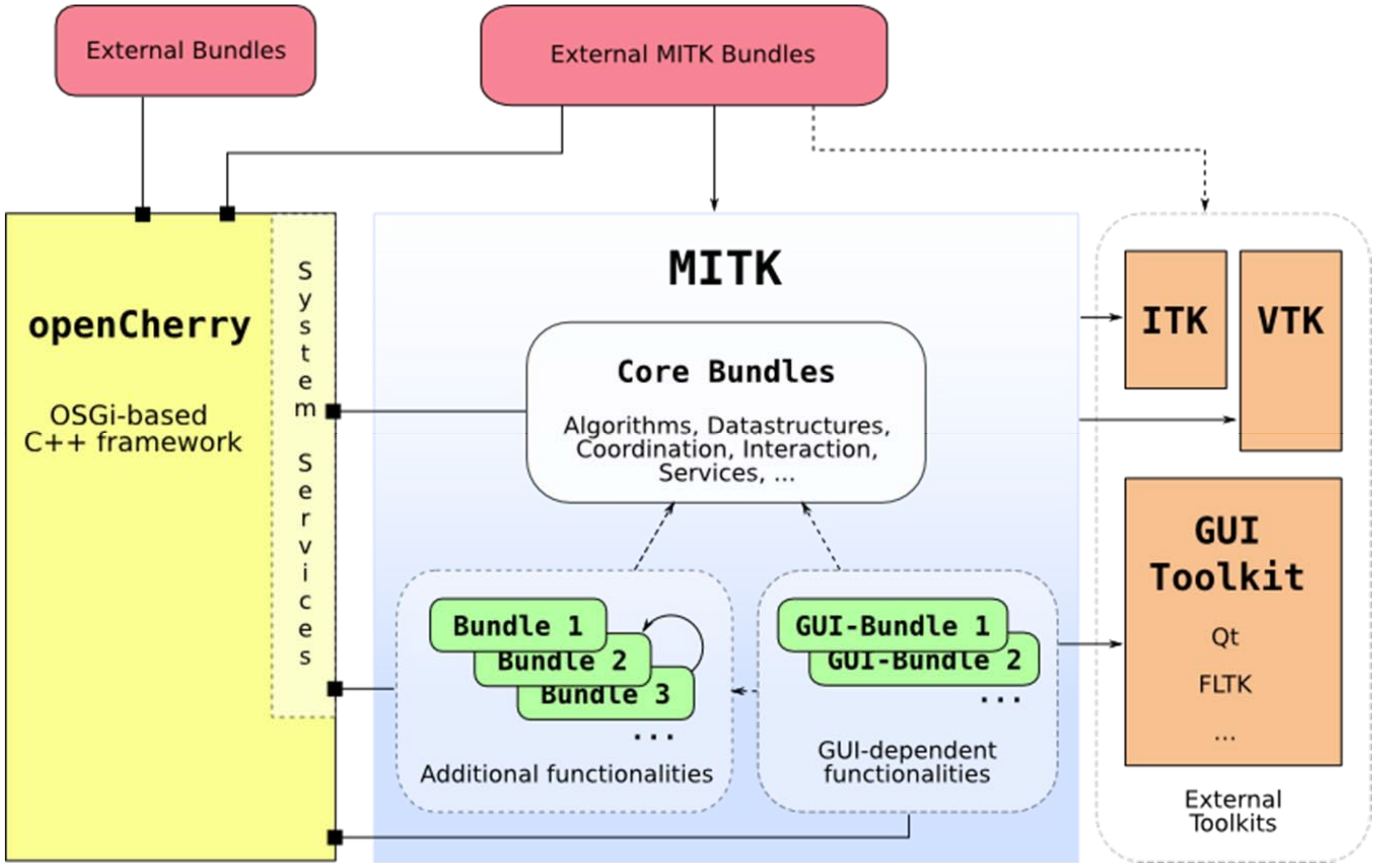
```
<extension point="org.opencherry.ui.views">
  <category
    id="org.mitk.views.general"
    name="MITK General"/>
  <view
    id="org.mitk.views.datamanager"
    name="Datamanager"
    category="org.mitk.views.general"
    icon="resources/datamanager.xpm"
    class="QmitkDataManagerView" />
</extension>
```

The Plug-In System



```
org.mitk.gui.qt.application 16712
  META-INF 16375
    MANIFEST.MF 16375
  src 16712
    internal 16375
    mitkQtAppDll.h 16224
    QmitkApplication.cpp 16224
    QmitkApplication.h 16375
    QmitkDefaultPerspective.cpp 16224
    QmitkDefaultPerspective.h 16224
    QmitkStdMultiWidgetEditor.cpp 16224
    QmitkStdMultiWidgetEditor.h 16712
  manifest.cpp 16375
  CMakeLists.txt 16375
  files.cmake 15350
  plugin.xml 16278
```

Plug-in Architecture



Benefits

- Lazy loading through extension points
- Plug-ins can extend the Platform‘s capabilities
- Plug-ins can also extend the capabilities of other plug-ins

- You can customize your application by changing the set of plug-ins and defining perspectives
- Your plug-ins can be reused in any other openCherry application
- You can give away your code/algorithm in binary form

Thank you!

Any questions?

Coffee break