MITK in the context of NA-MIC

The Medical Imaging Interaction Toolkit

Div. of Medical and Biological Informatics, DKFZ Heidelberg

Open-source Toolkits

Powerful toolkits for

- Visualization: VTK (www.vtk.org)
- Segmentation/registration: ITK (www.itk.org)

But:

insufficient support for
interactive, multi-view software

MITK …

- uses parts of NA-MIC: **ITK** & **VTK**
- adds features outside the scope of boths

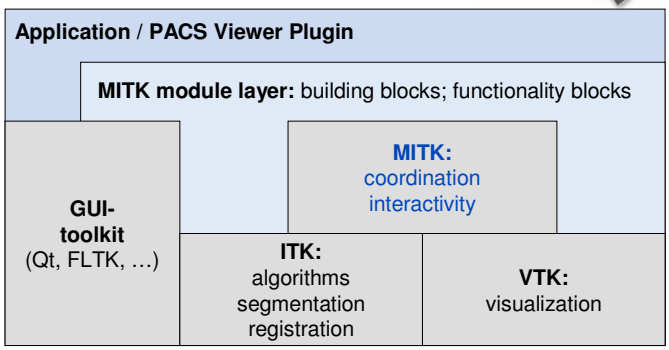
➔ is not at all a competitor to VTK or ITK

MITK

dkfz.

Medical Imaging Interaction Toolkit (MITK)

- open-source C++ toolkit based on ITK/VTK
- coordination of visualizations and interactions
- combine modules developed independently from each other



Ivo Wolf
Medizinische und Biologische Informatik
Heidelberg

MITK

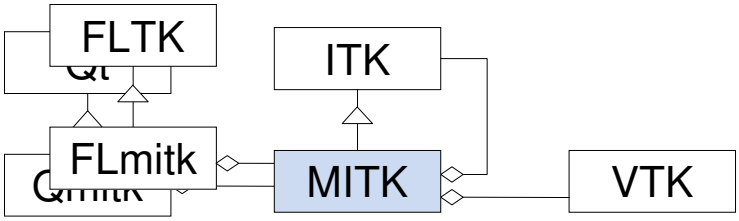
dkfz.

- Object-oriented C++ framework / toolkit
- BSD-style license, almost identical to VTK / ITK
- Supports
  - Linux and Windows
  - gcc 3.3, 4.2, VC7.1, VC8, VC9
  - Latest VTK release
  - Latest two ITK releases
- MITK-core does not depend on a GUI toolkit
- MITK-application-level provides
  - Qt3 base application
  - Many Qt3 widgets
  - FLTK example
  - Qt4 is work in progress

Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

Re-use of design and concepts

dkfz.

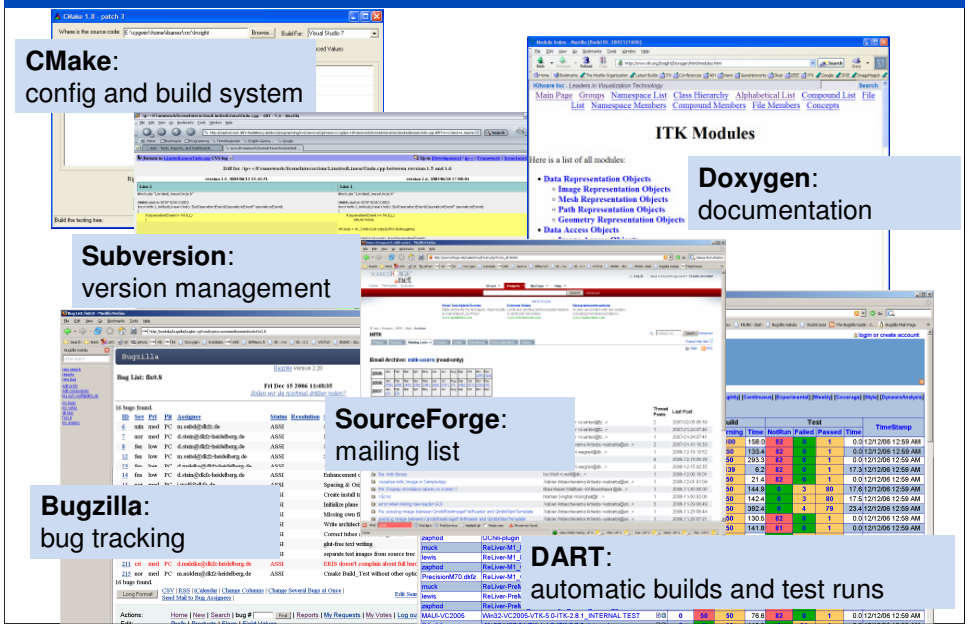


- MITK's core is GUI independent

Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

MITK uses NA-MIC tools and software process

dkfz.



CMake: config and build system

Doxygen: documentation

Subversion: version management

SourceForge: mailing list

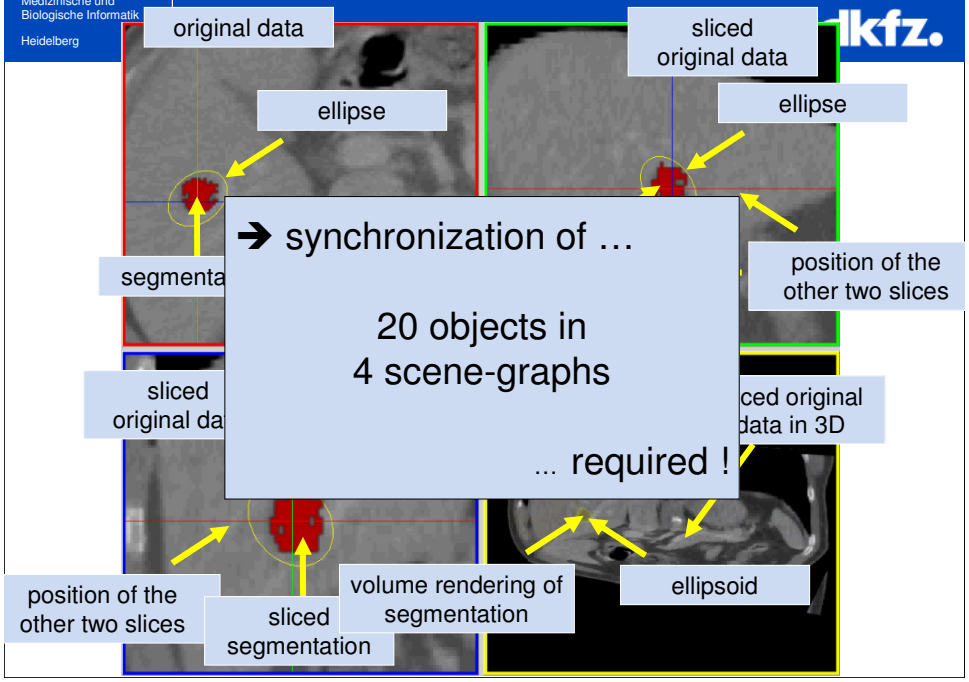
Bugzilla: bug tracking

DART: automatic builds and test runs

What MITK does – a quick overview

DEUTSCHES KREBSFORSCHUNGSZENTRUM IN DER HELMHOLTZ-GEMEINSCHAFT

Ivo Wolf
Medizinische und Biologische Informatik
Heidelberg



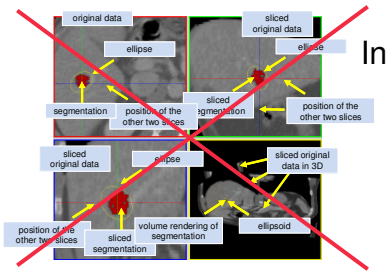
➔ synchronization of …

20 objects in
4 scene-graphs

… required !

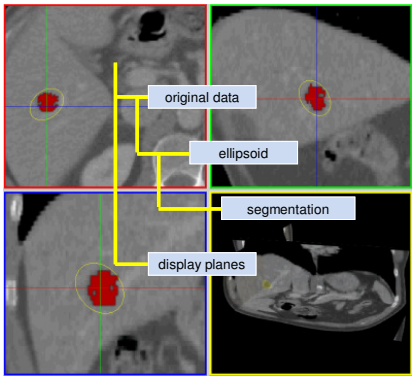
Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

Getting out of the maze …



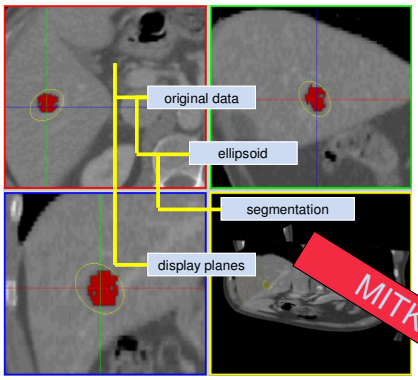
Instead of creating **many** scene-graphs with **even more** elements …

… create a **single data repository** with a **few data-objects**!



Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

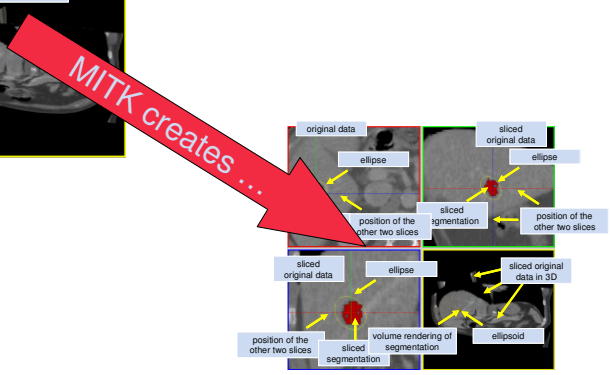
MITK: Data repository instead of scene-graphs



MITK takes the data repository …

and builds …

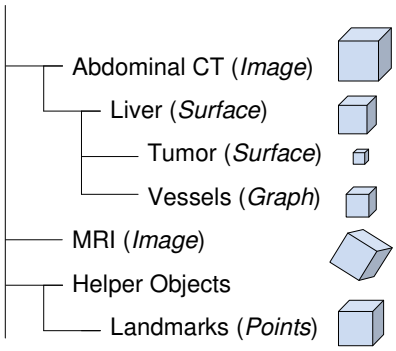
➔ VTK scene graphs



Data repository

dkfz.

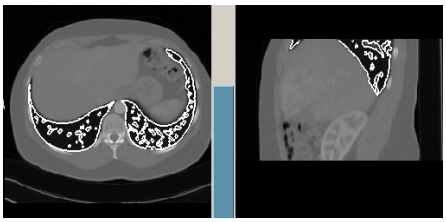
- Repositories for sharing data objects between modules
- Any number of data objects
- Any kind of data objects
- Data objects with geometry frame (bounding-box, transform, etc.)



Rendering the data repository

dkfz.

RenderWindows:

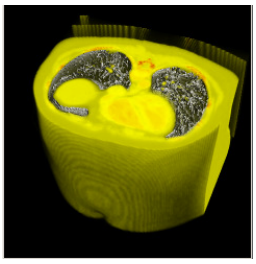


- **single** RenderWindow class
- **different types** of views
  - ➔ 2D/3D
  - ➔ special views definable (e.g., for AR)

```
renderer->SetMapperID(mitk::BaseRenderer::Standard3D);
```

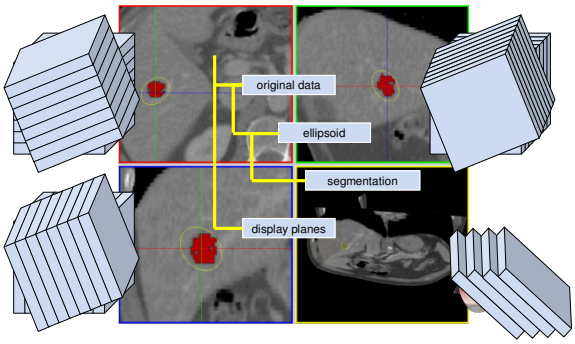
- **point** to the **data repository**
  - ➔ **any number of views** on the data:

```
renderer1->SetData(repository);
renderer2->SetData(repository);
...
```



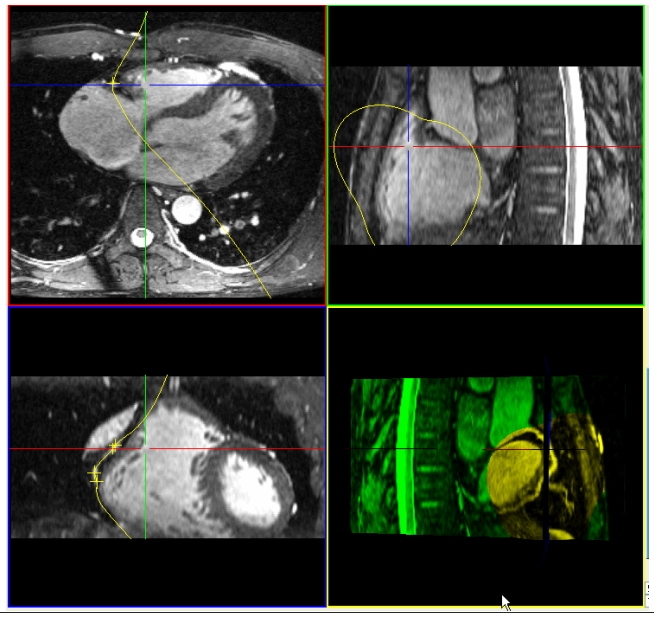
Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

Defining *how* we want to see the data …



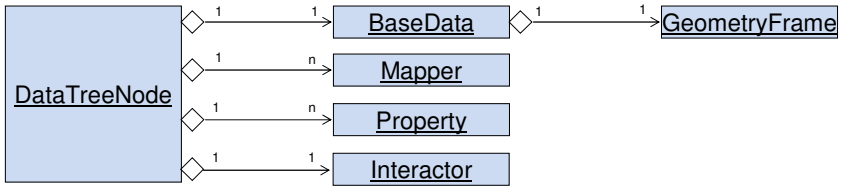
Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

Render and interact on curved planes



Ivo Wolf
Medizinische und Biologische Informatik
Heidelberg

The nodes in the data repository



BaseData: the actual data: images, surfaces, etc.
GeometryFrame: position and orientation in space
Mappers: render the data into a renderwindow
Properties: define how to draw the data
Interactor: defines user interaction with the data

Ivo Wolf
Medizinische und Biologische Informatik
Heidelberg

Controlling rendering with properties

- Rendering specific properties

Generic	Image	PointSet	Surface
• visible • layer • name	• opacity • color • use color • binary • outline binary • texture interpolation • reslice interpolation • volumerendering • levelwindow • LookupTable • TransferFunction	• line width • pointsize • selectedcolor • color • contour • contourcolor • close • show points • show distances • distance decimal digits • show angles • show distant lines	• line width • scalar mode • wireframe line width • material • scalar visibility • color mode • representation • interpolation

How to add a new data type

Extension for new data types:

- ➔ derive data class
- ➔ derive mapper
- ➔ create file I/O
- ➔ Register mapper / I/O handler at factory

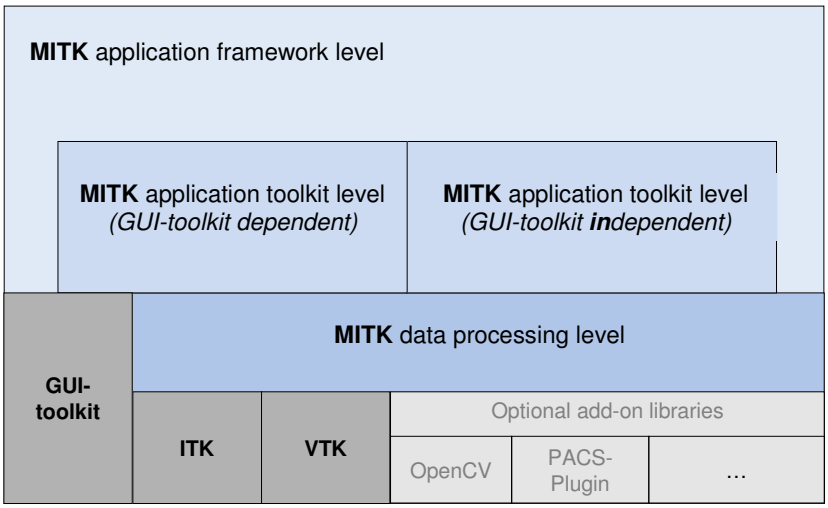
Example:

- attributed vessel graphs

[DKFZ and University of Tübingen]

MITK Architecture

MITK Architecture



Data Processing level

- access to ITK and VTK data structures and algorithms
- Access to other libraries (OpenCV, ANN, TinyXML,…)
- Tree / Graph data structures and algorithms
- Spatial object location (Geometries)
- Time steps for data objects
- Loading / saving of different file formats
- Interface to tracking systems

Application toolkit level

- Rendering
  - Mappers, Update Management, Render Properties
- Data Management
  - Object Container, Object Properties, Scene Management
- Interaction
  - Statemachine based
- Undo/Redo
- Processing of tracking data
- Qt Widgets
  - TreeNodeSelector, StandardViews, PropertyEditor, LevelWindow, Renderwindow, SlicerControls, Navigationviews,…

Application framework

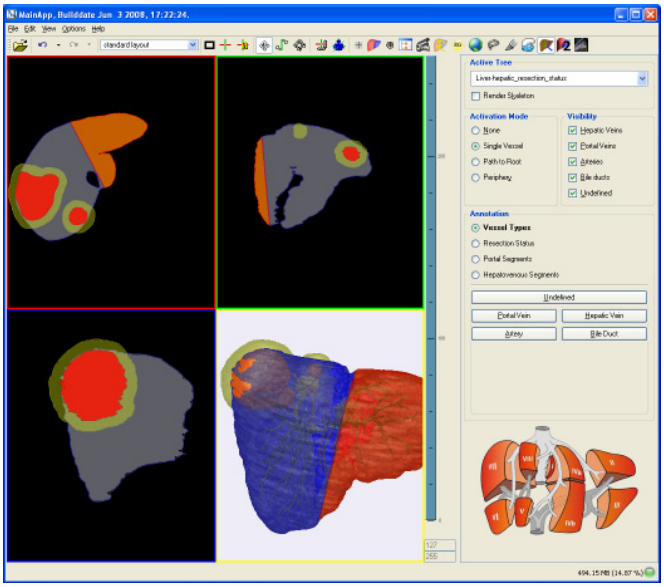
Base application (*MITK-MainApp*):

- Container for functionalities
  - independent „Plug-Ins“ for specific problems
- Shared repository for data objects
- Persistence:
  - Application state can be saved and restored on next startup
- Interface to CHILI-PACS Workstation

Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

MITK - MainApp

dkfz.



Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

MITK functionality modules

dkfz.

Functionality = a module with …

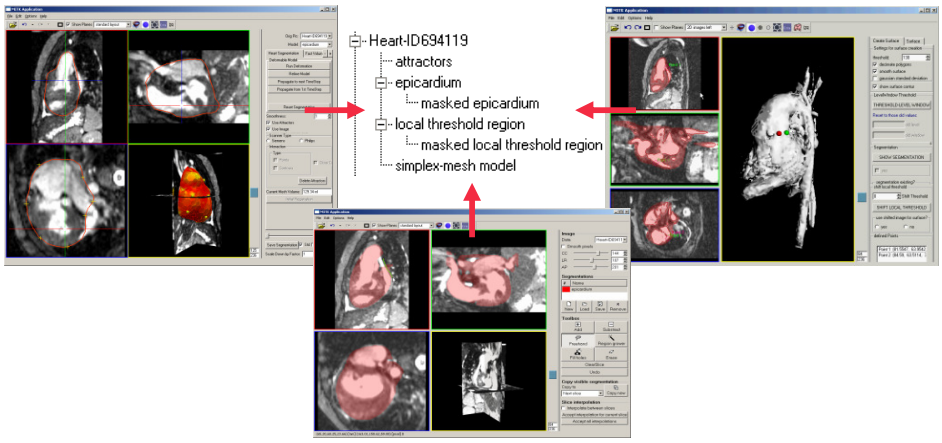
- an identification (icon/tooltip/…)
- a workspace area
- a control area
- a option dialog
- a help page (manual)
- the algorithmic implementation

Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

Combining functionality blocks

dkfz.

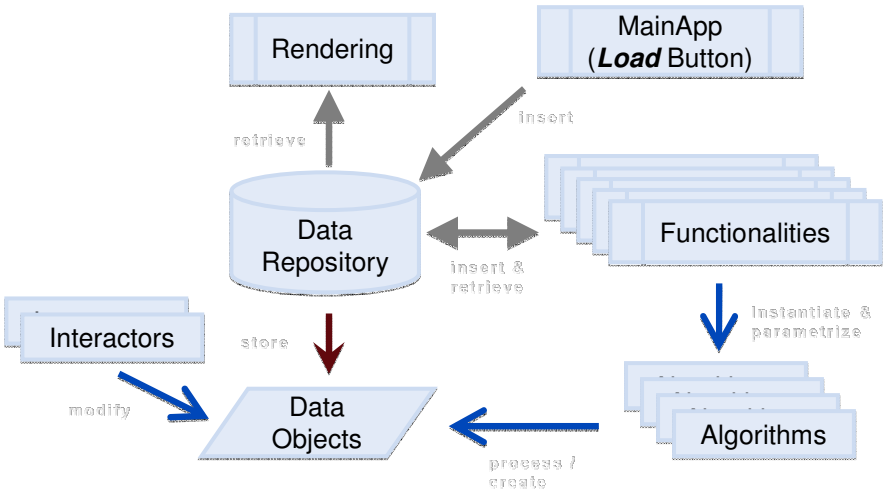
- Functionalities are independent from each other
- They communicate via the data repository



Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

Architecture of MITK-MainApp

dkfz.



Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

Navigation / IGT with MITK

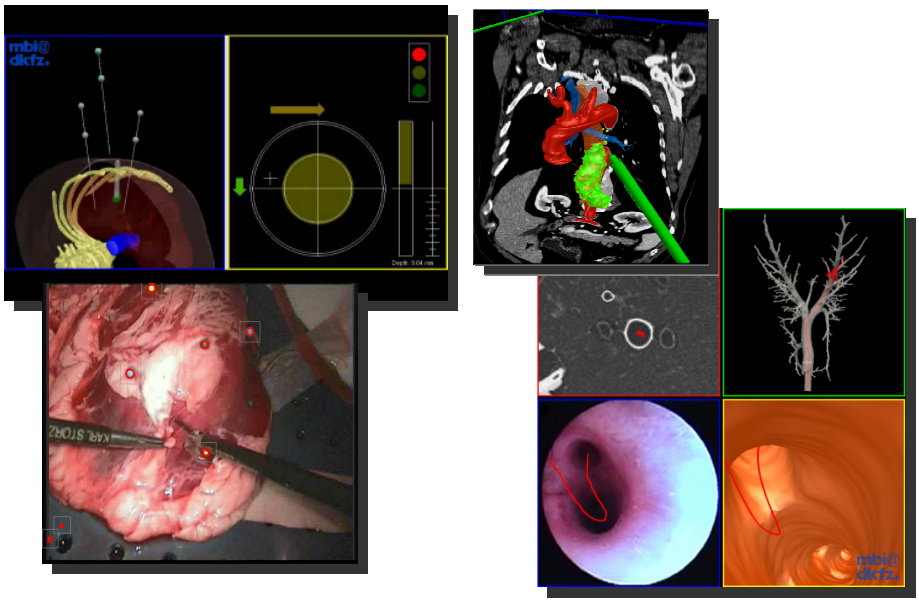
dkfz.

- Tracking component allows access to different tracking systems:
  - NDI Polaris/Aurora
  - Microntracker
  - Our own video based Inside-Out-Tracking algorithm
- Filter pipelines for tracking coordinates (Kalman-filter,…)
- Logging & replay of tracking data
- Geometry classes to manage different coordinate systems
- (not yet open source)

Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

Examples of IGT applications with MITK

dkfz.



Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

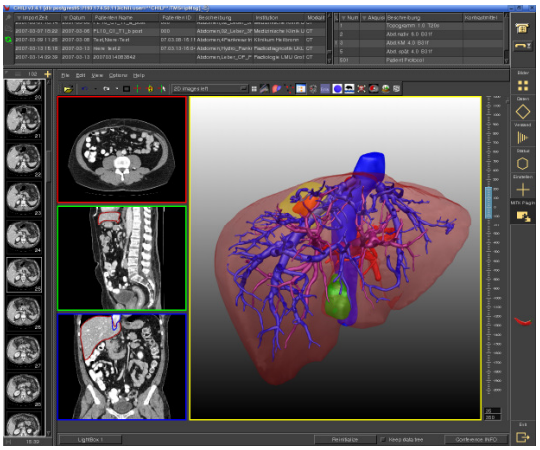
Embedding in PACS/tele-conferencing system

dkfz.

Integration in PACS/telemedicine system CHILI® as a PlugIn:

- PACS
- Connection to modalities
- DICOM import/export
- DICOM “unification”
- Data transfer
- Tele-radiology
- Management of results from image processing

➔ facilitates clinical integration



How to get started

Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

www.mitk.org

dkfz.

Download options:

- anonymous svn:

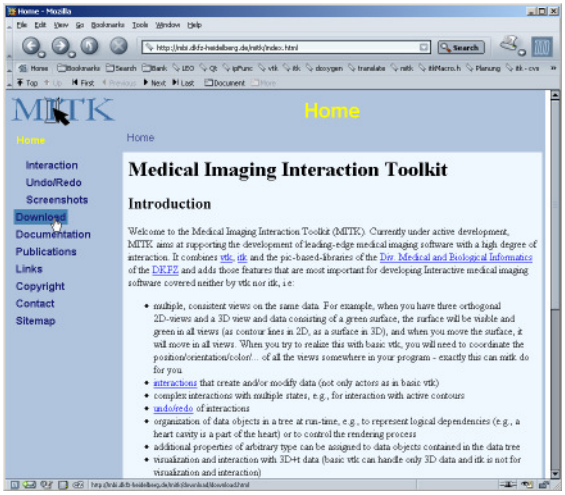
svn co http://svn.mitk.org/trunk/mitk/

- zipped archive (v 0.8)

https://sourceforge.net/projects/mitk/

Tutorial:

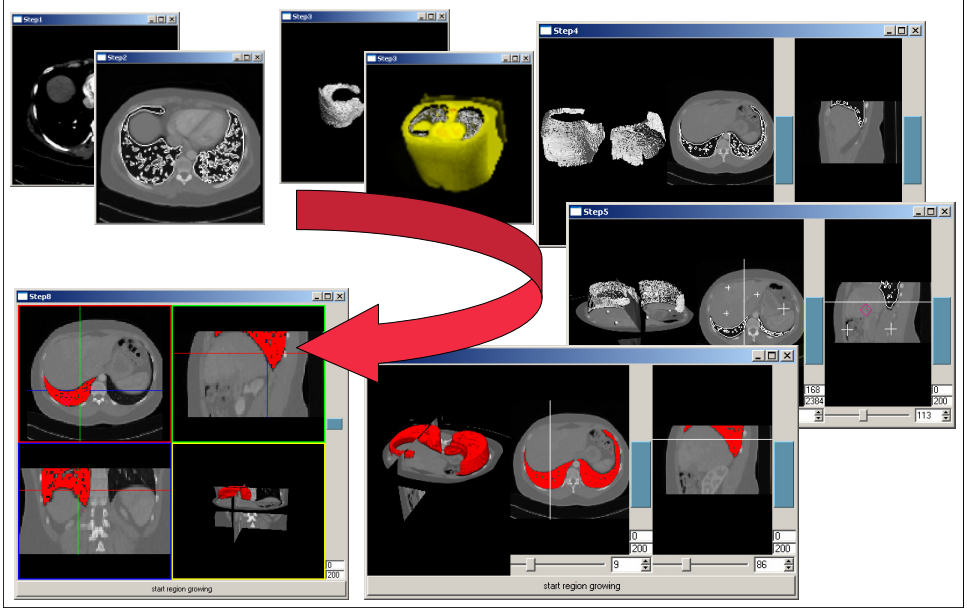
http://mitk.org/documentation/



Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

9-step tutorial

dkfz.



Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

A small functionality

dkfz.

We'll have a look at a very simple functionality for region growing:

0. (create a functionality)
1. select an image
2. set some seed points
3. react, when a GUI button is pressed
4. run a region grower from ITK
5. display the result in MITK

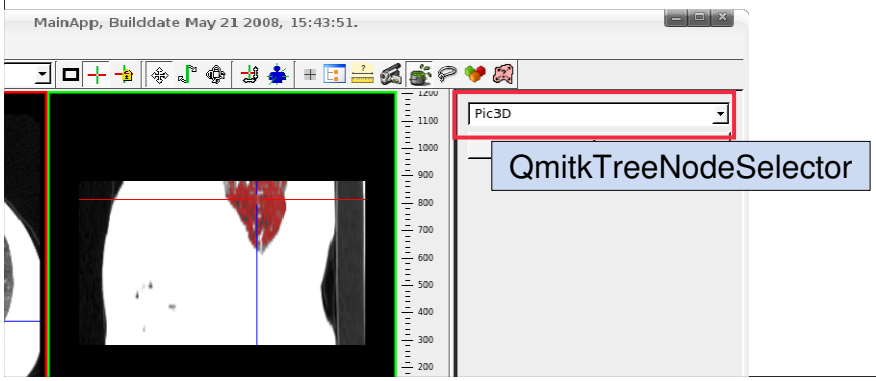
(can be downloaded at mitk.org)

Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

A small functionality

dkfz.

1. **selection of an image**
2. set some seed points
3. react, when a GUI button is pressed
4. run a region grower from ITK
5. display the result in MITK



Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

A small functionality

dkfz.

1. selection of an image
2. **set some seed points**
3. react, when a GUI button is pressed
4. run a region grower from ITK

PointSetInteractor

```
QmitkRegionGrowing.cpp (~/mitk/extern/src/QFunctional
File  Edit  View  Terminal  Tabs  Help
 92
 93 void QmitkRegionGrowing::Activated()
 94 {
 95   QmitkFunctionality::Activated();
 96
 97   if (m_PointSetNode.IsNull())
 98   // only once create a new DataTreeNode containing a PointSet with some interaction
 99   {
100     // new node and data item
101     m_PointSetNode = mitk::DataTreeNode::New();
102     m_PointSetNode->GetPropertyList()->SetProperty("name", mitk::StringProperty::New("Seedpoints for region growing"));
103     m_PointSet = mitk::PointSet::New();
104     m_PointSetNode->SetData( m_PointSet );
105
106     // new behaviour/interaction for the pointset node
107     m_Interactor = mitk::PointSetInteractor::New("pointsetinteractor", m_PointSetNode);
108     mitk::GlobalInteraction::GetInstance()->AddInteractor( m_Interactor );
109
110     // add the pointset to the data tree (for rendering)
111     GetDataTreeIterator()->Add( m_PointSetNode );
112   }
113 }
114
"~/mitk/extern/src/QFunctionalities/QmitkRegionGrowing/QmitkRegionGrowing.cpp" 238 lines --45%--          109,0-1       42%
```

Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

A small functionality

dkfz.

1. selection of an image
2. set some seed points
3. **react, when a GUI button is pressed**
4. run a region grower from ITK
5. display the result in MITK

Qt “connections”

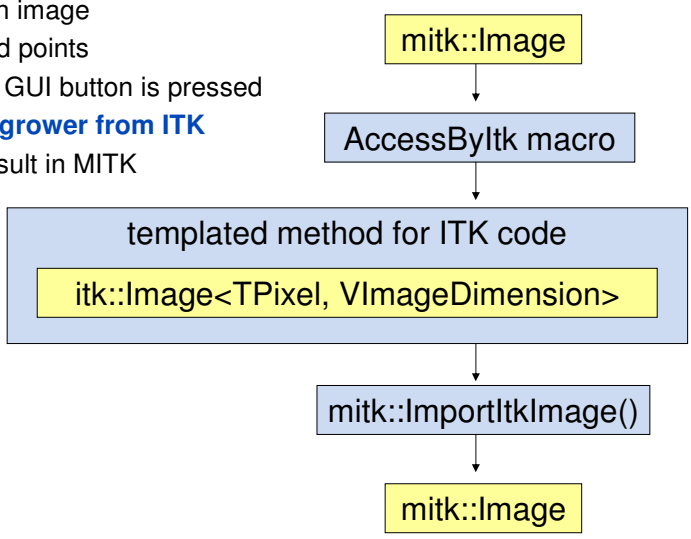
```
QmitkRegionGrowing.cpp (~/mitk/ext...ionalities/QmitkRegionGrowing) - VIM
File  Edit  View  Terminal  Tabs  Help
71
72 void QmitkRegionGrowing::CreateConnections()
73 {
74   if ( m_Controls )
75   {
76     connect( (QObject*)(m_Controls->btnRegionGrow), SIGNAL(clicked()),
77                                                this, SLOT(DoRegionGrowing()) );
78   }
79 }
80
                                                        80,0-1        30%
```

Ivo Wolf
Medizinische und Biologische Informatik
Heidelberg

A small functionality

dkfz.

1. selection of an image
2. set some seed points
3. react, when a GUI button is pressed
4. **run a region grower from ITK**
5. display the result in MITK



Ivo Wolf
Medizinische und Biologische Informatik
Heidelberg

A small functionality

dkfz.

3. react, when a GUI button is pressed
4. run a region grower from ITK
5. **display the result in MITK**

QmitkRegionGrowing.cpp (~/mitk/extern/...Functionalities/QmitkRegionGrowing) - VIM

File Edit View Terminal Tabs Help

```
217
218   regionGrower->Update();
219
220   mitk::Image::Pointer resultImage = mitk::ImportItkImage( regionGrower->GetOutput() );
221   mitk::DataTreeNode::Pointer newNode = mitk::DataTreeNode::New();
222   newNode->SetData( resultImage );
223
224   // set some properties
225   mitk::DataTreeNodeFactory::SetDefaultImageProperties( newNode );
226   newNode->SetProperty("binary", mitk::BoolProperty::New(true));
227   newNode->SetProperty("name", mitk::StringProperty::New("dumb segmentation"));
228   newNode->SetProperty("color", mitk::ColorProperty::New(1.0,0.0,0.0));
229   //newNode->SetProperty("volumerendering", mitk::BoolProperty::New(true));
230   newNode->SetProperty("layer", mitk::IntProperty::New(1));
231   newNode->SetProperty("opacity", mitk::FloatProperty::New(0.5));
232
233   // add result to data tree
234   mitk::DataStorage::GetInstance()->Add( newNode );
235
236   mitk::RenderingManager::GetInstance()->RequestUpdateAll();
237 }
238 
                                                        238,0-1       Bot
```

The future

dkfz. DEUTSCHES KREBSFORSCHUNGSZENTRUM IN DER HELMHOLTZ-GEMEINSCHAFT

Ivo Wolf
Medizinische und Biologische Informatik
Heidelberg

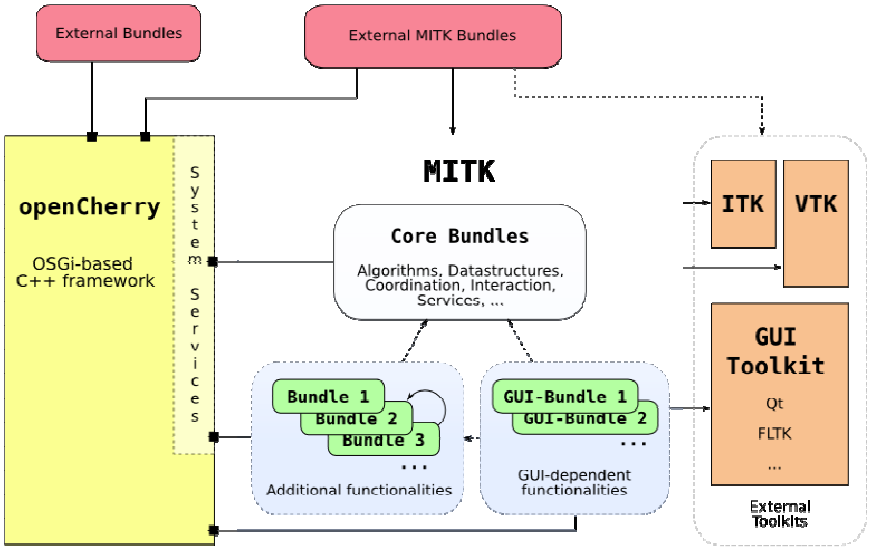
OSGi-based Extensibility

dkfz.

OSGi-based extensibility for MITK-applications:

- OSGi: component model originally designed for Java
- Basic building blocks are *bundles* (aka plugins) and *services*
- Easy extensibility through loose coupling
- Every plugin can define its own *extension points*
  ➔ general concept for extensions
  ➔ plugins within plugins at no additional costs
- MITK will provide a set of core *bundles* and *services* for complex imaging tasks and interactions

OSGi-based Extensibility



Other enhancements

Hot topics:

- Release of two functionalities for registration
- Transition of the Qt3 MITK code to Qt4
- OSGi-based application platform providing views/editors, perspectives and GUI-services (openCherry plugins)
- Python scripting

Ivo Wolf
Medizinische und
Biologische Informatik
Heidelberg

dkfz.

Thank you !

MITK

www.mitk.org