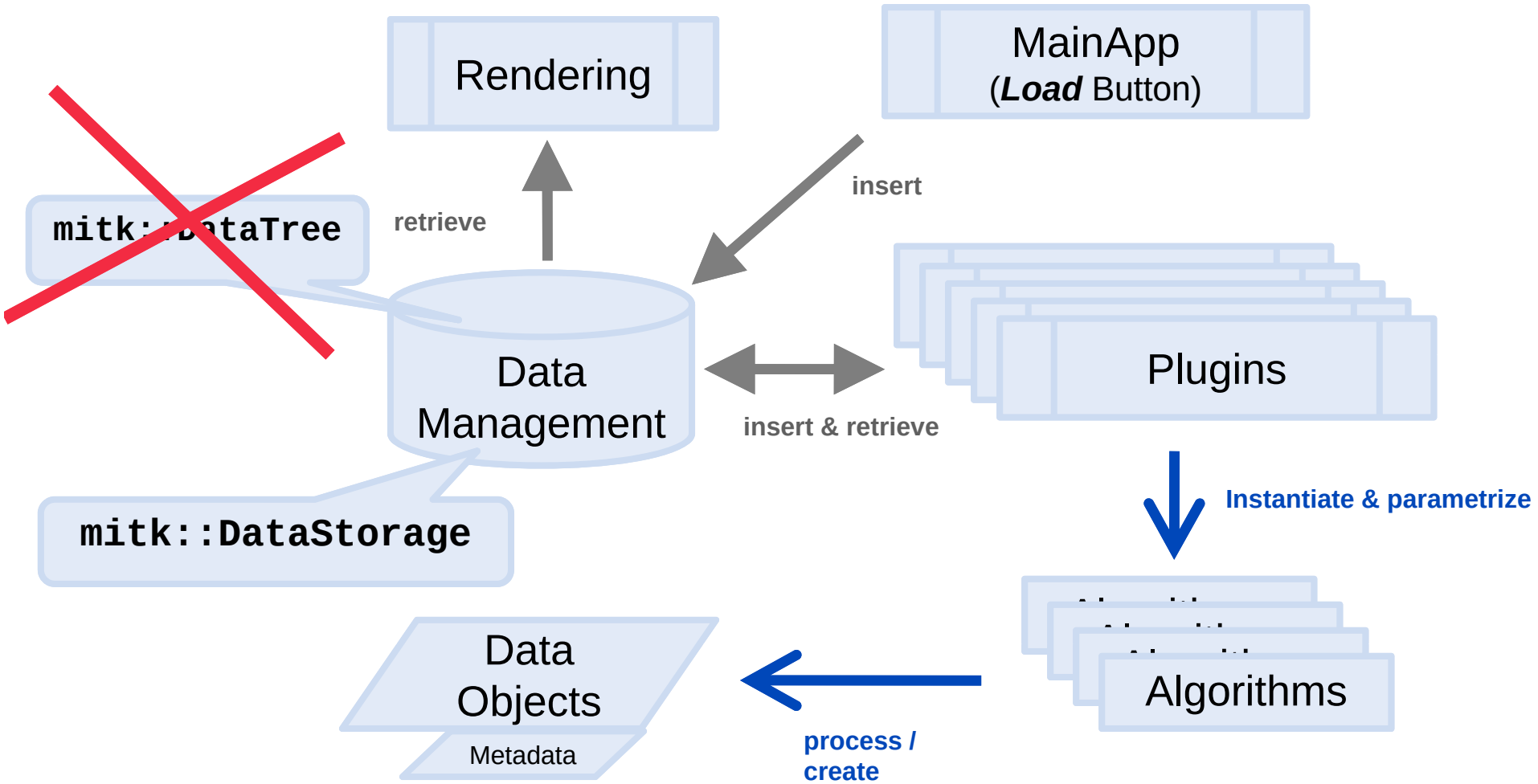
5/20/2009

MITK data management

Nach dem MITK-Retreat

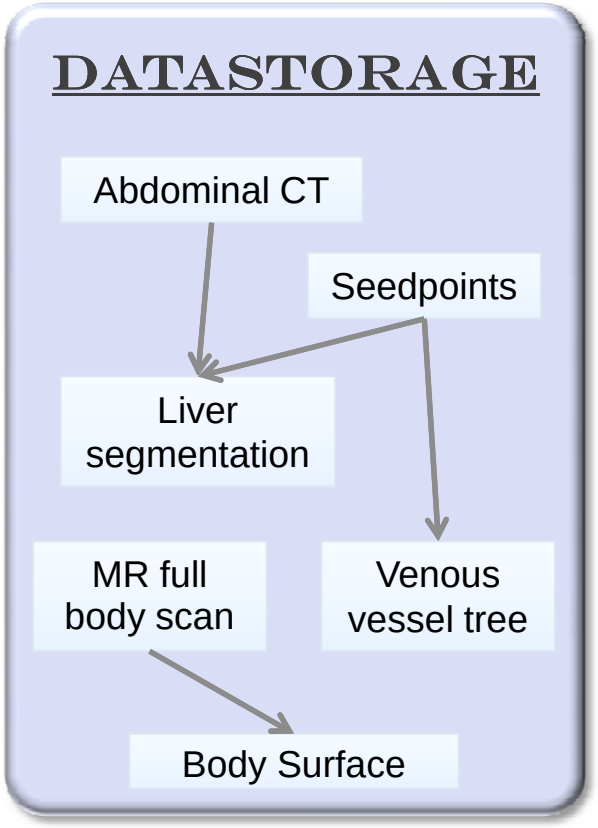
dkfz.
GERMAN
CANCER RESEARCH CENTER
IN THE HELMHOLTZ ASSOCIATION

Shared repository in MITK MainApp



DataStorage

- Database like queries for objects
  - Similar to ***`SELECT * FROM repository WHERE DataType == mitk::Image`***
  - Predicate objects to build WHERE statement
- Stores relation of objects in a ***directed acyclic graph***
  - object can be derived from multiple source objects
  - object can have multiple children objects
- Events:
  - *AddNodeEvent*
  - *RemoveNodeEvent*
  - *ChangedNodeEvent*
  - *DeleteNodeEvent*



Adding elements

```
/* create some DataTreeNodes */
mitk::DataTreeNode::Pointer n1 = mitk::DataTreeNode::New();
…
mitk::DataTreeNode::Pointer n2 =
…
mitk::DataTreeNode::Pointer n3 =
…
/* Create Data Storage */
mitk::DataStorage::Pointer ds = <get a DataStorage from
  somewhere>;
/* Fill DataStorage */
ds->Add(n1);
ds->Add(n2);
mitk::DataStorage::SetOfObjects::Pointer parents =

  mitk::DataStorage::SetOfObjects::New();
parents->InsertElement(0, n1);  // n3 is source of n1
ds->Add(n3, parents);
```

No Singleton implementation anymore!

In OpenCherry Functionality:
`this->GetDefaultDataStorage();`

Retrieving elements

```
/* retrieve all objects */
mitk::DataStorage::SetOfObjects::ConstPointer all = ds->GetAll();
for (SetOfObjects::ConstIterator it = all->Begin(); it != all->End();
  ++it)
{
  mitk::DataTreeNode::Pointer node = it.Value();
}

/* retrieve objects with specific criteria */
SetOfObjects::ConstPointer rs =
  ds->GetSubset(mitk::NodePredicateDataType::New("Image"));
```

Retrieving elements II

- Existing NodePredicates:
  - `mitk::NodePredicateData` Check for specific data object
  - `mitk::NodePredicateDataType` Check for data type (Image, Surface,…)
  - `mitk::NodePredicateDimension` Check for dimension of data object
  - `mitk::NodePredicateProperty` Check for existence of property with specific **name** or for existance of property with specific **name and value**
  - `mitk::NodePredicateAND` combine multiple predicates
  - `mitk::NodePredicateOR` combine multiple predicates
  - `mitk::NodePredicateNOT` negate predicate

Retrieving related elements

```
/* retrieve source & derived objects */
SetOfObjects::ConstPointer sources = ds->GetSources(n3);
SetOfObjects::ConstPointer child = ds->GetDerivations(n1);

/* retrieve source & derived objects with specific criteria */
mitk::NodePredicateDataType::Pointer p =
  mitk::NodePredicateDataType::New("Image");
SetOfObjects::ConstPointer sources = ds->GetSources(n3, p);
  SetOfObjects::ConstPointer child = ds->GetDerivations(n1, p);
```

Convenience DataStorage methods

```
/* Is a node already in the DataStorage? */
mitk::DataTreeNode* n = […]
bool nodeInDataStorage = ds->Exists(n);

/* Retrieve single Nodes/objects */
mitk::DataTreeNode* n = ds->GetNode(myPredicate);
mitk::DataTreeNode* n = ds->GetNamedNode("MyNode");
mitk::Image* image = ds->GetNamedObject<mitk::Image>("data");
```