Polska Infrastruktura Informatycznego Wspomagania Nauki
w Europejskiej Przestrzeni Badawczej

Grid Resource Registry – Abstract Layer to Computational Resources

Marek Kasztelnik and Marian Bubak

Institute of Computer Science AGH
ACC Cyfronet AGH

INNOWACYJNA GOSPODARKA
NARODOWA STRATEGIA SPÓJNOŚCI

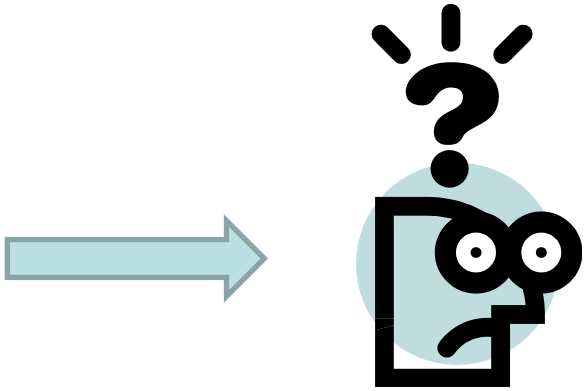
UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ
ROZWOJU REGIONALNEGO

Outline

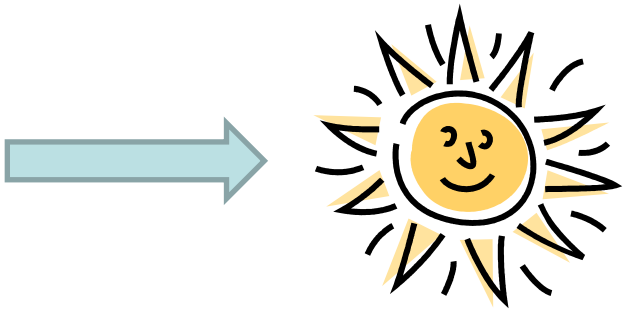
- Motivation
- Scenario of creating e-science application
- Grid Resource Registry (GRR) architecture and functionalities
- GRR usage scenario
- Conclusions

Motivation

- Many technologies, e.g.:
  - Web services
  - Rest services
  - WSRF
  - CCA components

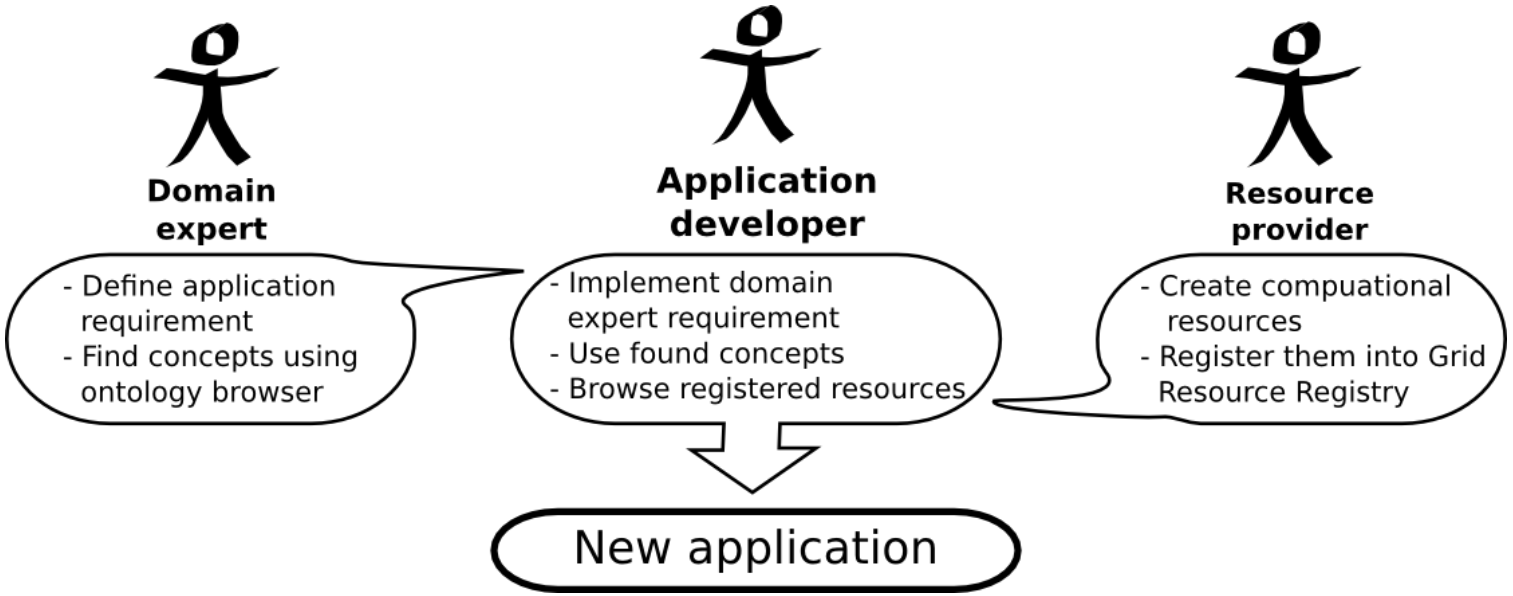


- High level service description, e.g:
  - Service X delivers:
    - A
    - B
    - C
    - …



E-science application development team

- The process of collaborative e-science application creation requires involvement on many people from different groups of interest
- Three groups of users identified:
  - **domain expert** – defines application requirements.
  - **application developer** - implements application using resources delivered by third user group
  - **resource providers**



INNOWACYJNA GOSPODARKA
NARODOWA STRATEGIA SPÓJNOŚCI
CYFRONET
AGH
UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ ROZWOJU REGIONALNEGO

Algorithm for creating e-science application

- Domain expert defines functionality of the e-science application
- Application developer takes e-science application description and search for computational blocks, which can be re-used
- If required library is not available than developer creates description of the new required resource and send it into resource provider
- Resource provider implements missing library taking into account received description (API)
- Application developer create e-science application (glue code)
- Domain expert tests created application

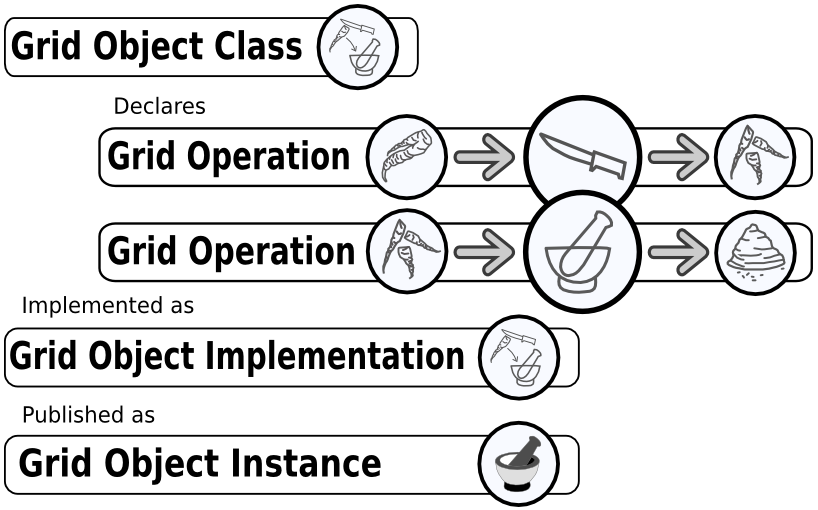
INNOWACYJNA GOSPODARKA
NARODOWA STRATEGIA SPÓJNOŚCI
CYFRONET
AGH
UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ
ROZWOJU REGIONALNEGO

Grid Resource Registry - functionality

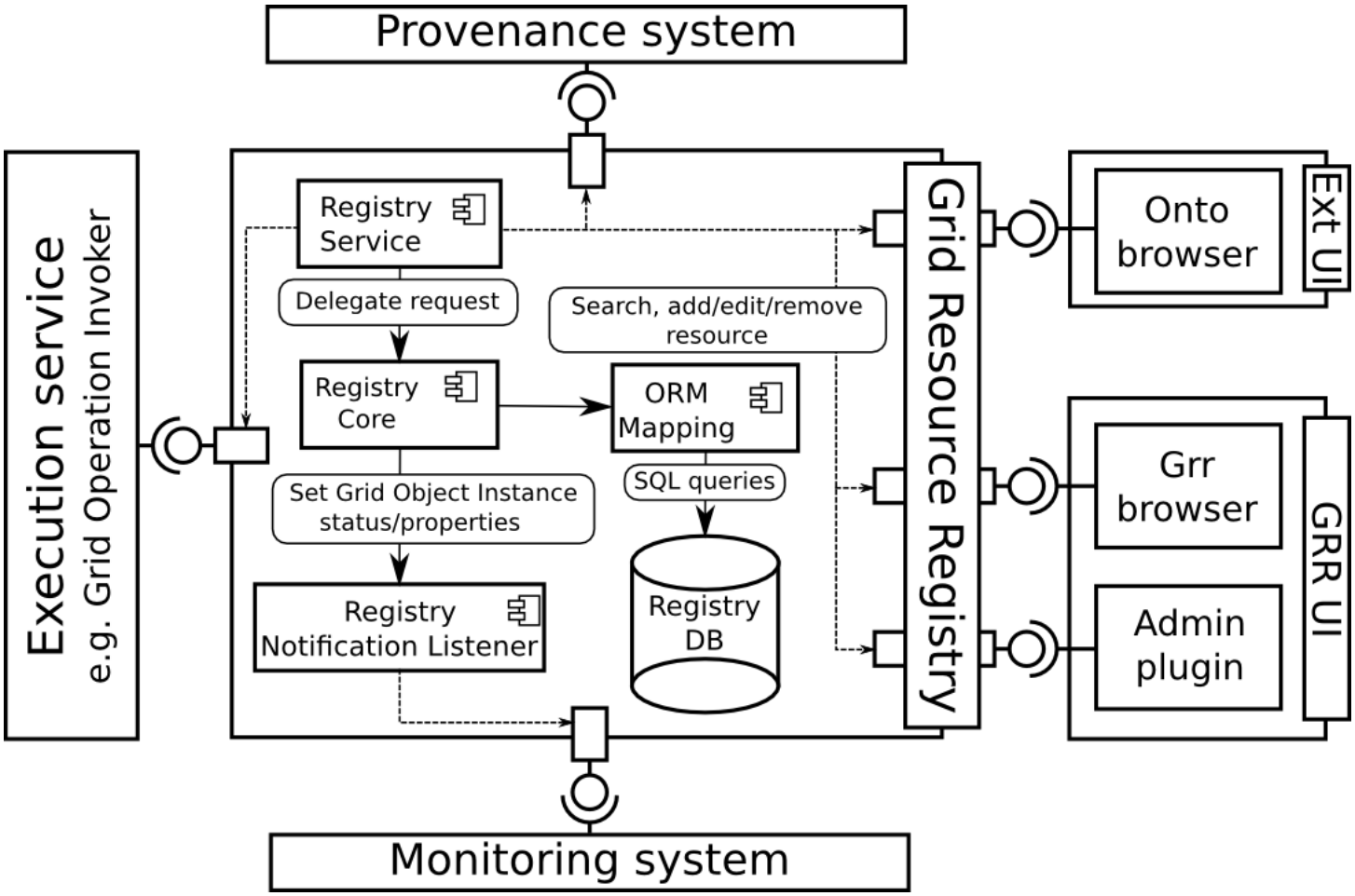
- GRR is a central place, where information about resources are located
- Clear separation between service behavior, service implementation and deployment details
- Dedicated interface for every group of e-science development team:
  - Domain expert: GRR is integrated with ontology browser
  - Application developer: Integration with Eclipse development environment
  - Resource provider: tools for resource registration, monitoring
- Integration with monitoring and provenance systems

Grid Object abstraction

- Resource description is composed of three levels:
  - Grid Object specification of the resource behavior
  - Grid Object Implementation stores information specific for given service implementation technology
  - Grid Object Instance describes deployment-specific properties



Grid Resource Registry architecture



External components, user interfaces

- Execution service – library which chooses the best instance of the GridObject
- Provenance system – stores information about used resources during e-science application run
- Monitoring – monitor registered resources, thus during e-science application execution only working services are taken into account

- Grid Resource Registry browser – integrated with Eclipse RPC and Ruby editor
- Grid Resource Registry administration tools – add/edit/remove GRR elements
- Ontology Browser – delivers domain knowledge about GRR content

Simple data mining application (1)

- Goal: create data mining application which loads training data from the database, train classifier and classify defined set of data.
- Used tools:
  - Grid Resource Registry
  - Grid Operation Invoker – ruby library for invoking different technologies in uniform way
- Steps:
  - Search GRR for required Grid Objects
  - Create e-science application (glue code)

Simple data mining application (2)

- 2 Grid Objects used:
  - *WekaGem*
  - *OneRuleClassifier*

```
retriever = GObj.create('WekaGem')
A = retriever.loadDataFromDatabase(
    DATABASE, QUERY, USER, PASSWORD)

B = retriever.splitData(A, 50)
trainA = B.trainingData
testA = B.testingData

classifier = GObj.create('OneRuleClassifier')
classifier.train(trainA, attributeName)
prediction = classifier.classify(testA)
classificationPercentage = retriever.compare(
    testA, prediction, attributeName)

puts 'Predicted data: ' + prediction
puts 'Prediction quality: ' + classificationPercentage
```

Web Service

CCA component

Conclusions

- E-science development team requires different tools for different group of users:
  - Domain focused tools for Domain Expert
  - High level programming language for glue code development and a set of libraries for uniform access to computational resources
- Grid Resource Registry delivers such tools and introduce resource abstraction description
- Developer can focus on required functionality instead of technology complexity

http://dice.cyfronet.pl

http://virolab.cyfronet.pl