
Towards Collaborative Workbench for Science 2.0 Applications

Marian Bubak

AGH University of Science and Technology, Krakow PL; University of Amsterdam, NL

and

Bartosz Balis, Tomasz Bartynski, Eryk Ciepiela, Wlodek Funika,
Tomasz Gubala, Daniel Harezlak, Marek Kasztelnik, Joanna Kocot,
Maciej Malawski, Jan Meizner, Piotr Nowakowski, Katarzyna Rycerz,
Adam Belloum

dice.cyfronet.pl

Outline

- New applications and computing infrastructure
- Workflow systems and virtual laboratories
- High-level scripting approach, provenance
- Implementation for multiscale-multiphysics and warning systems
- Environment for open science applications
- Challenges

Background

- **Scientific applications**
 - Compute- and data-intensive
 - Used in dynamic scenarios – experiments
 - Multiscale, multiphysics
 - Various levels of coupling and composition types
 - Collaborative
 - Legacy codes in many programming languages
 - Linking to publications
- **Computing environments**
 - distributed and heterogeneous
 - resources are shared, possibly between different organizations
 - resources may dynamically change and may be not reliable
 - there is no single middleware
 - Collaborations in virtual organizations can be highly dynamic
 - Clouds – new opportunity

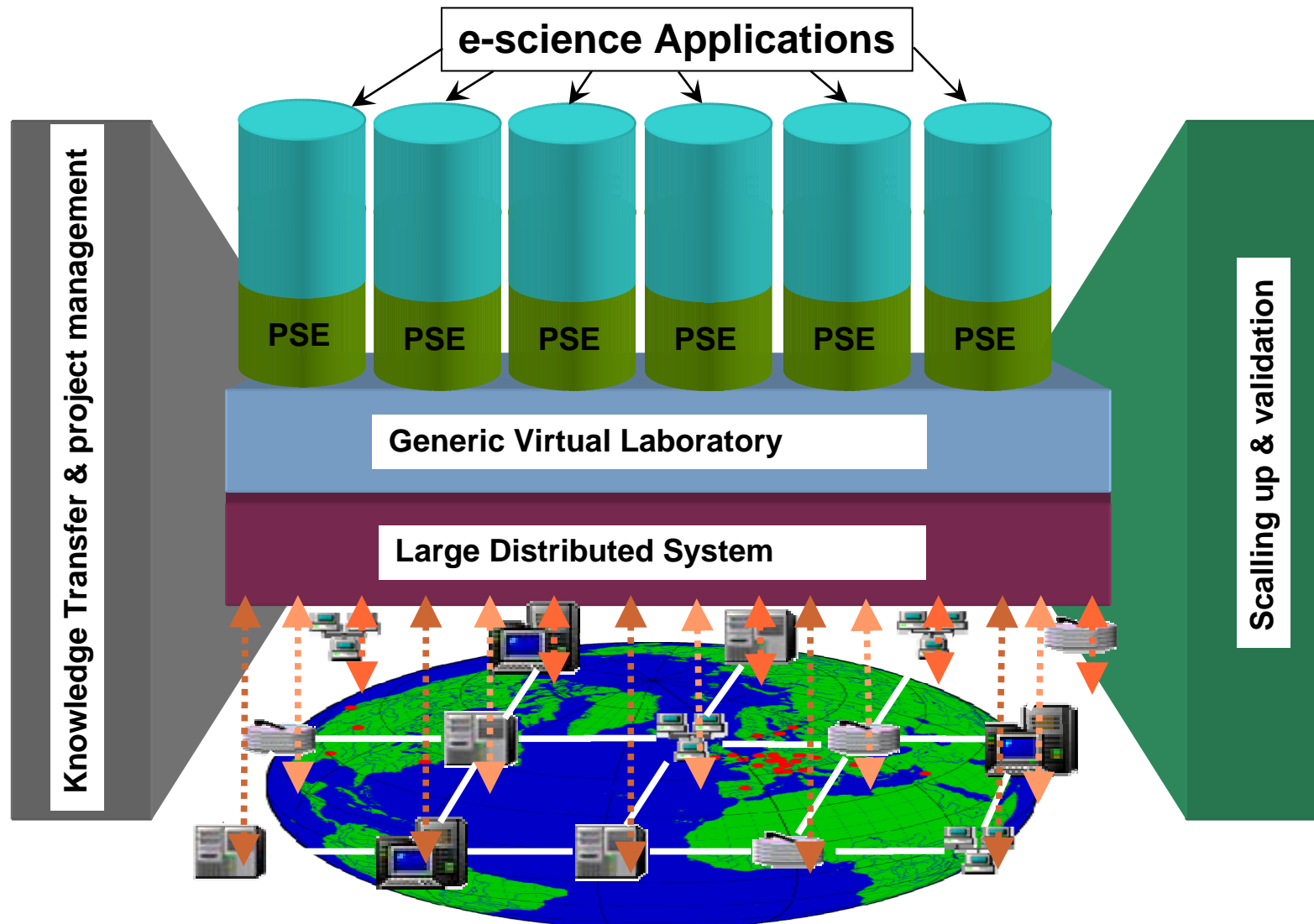
Required Functionality (1/2)

- **Development:**
 - focus on computational functionality, rapid application development
 - easy access to remote data
- **Experiment sharing:**
 - creation of applications by teams of developers
- **Browsing experiments:**
 - web application for browsing and executing experiments
- **Running experiments:**
 - single-click application execution
 - interactive communication between experiment and user
- **Gathering results:**
 - transparent
 - provenance storage and searching
- **Feedback:**
 - easy communication between end users and developers

Required Functionality (2/2)

- **Facilitating high-level programming**
 - Composition: temporal, direct links, hide middleware details
- **Facilitating deployment on shared resources**
 - virtualization layer, dynamic un-/re-deployment
- **Scalable to diverse environments**
 - PCs, clusters, grids, clouds; programming model independent from the infrastructure
- **Communication adjusted to various levels of coupling**
 - from P2P, WAN, LAN to local links, parallel communication
- **Supporting multiple programming languages**
 - Fortran, C, C++, Java
- **Adapted to (unreliable) computing environments**
 - monitoring, adaptive features, reconfiguration
- **Interoperable**
 - with standards, other models and implementations

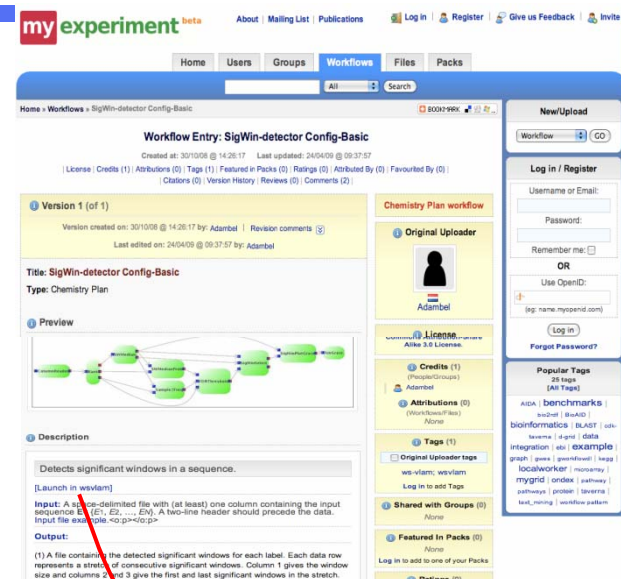
VL-e virtual laboratory



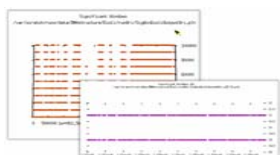
SigWin – detector Application

SigWin-detector: is a grid-enabled *workflow application* that takes a sequence of numbers and a series of window sizes as input and detects all significant windows for each window size using a moving median false discovery rate (mmFDR) procedure.

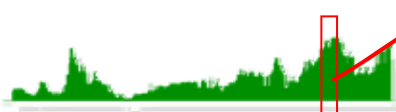
The results of a SigWin-detector analysis are summarized in a graph called SigWin-map. In the special case that the input sequence is a transcriptome map, the significant windows are called RIDGES and the output graph is called a RIDGEOGRAM.



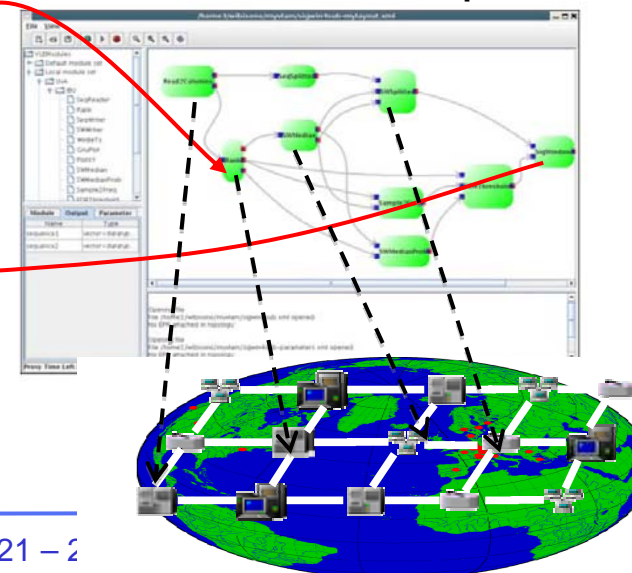
discovered RIDGE



Human transcriptome map



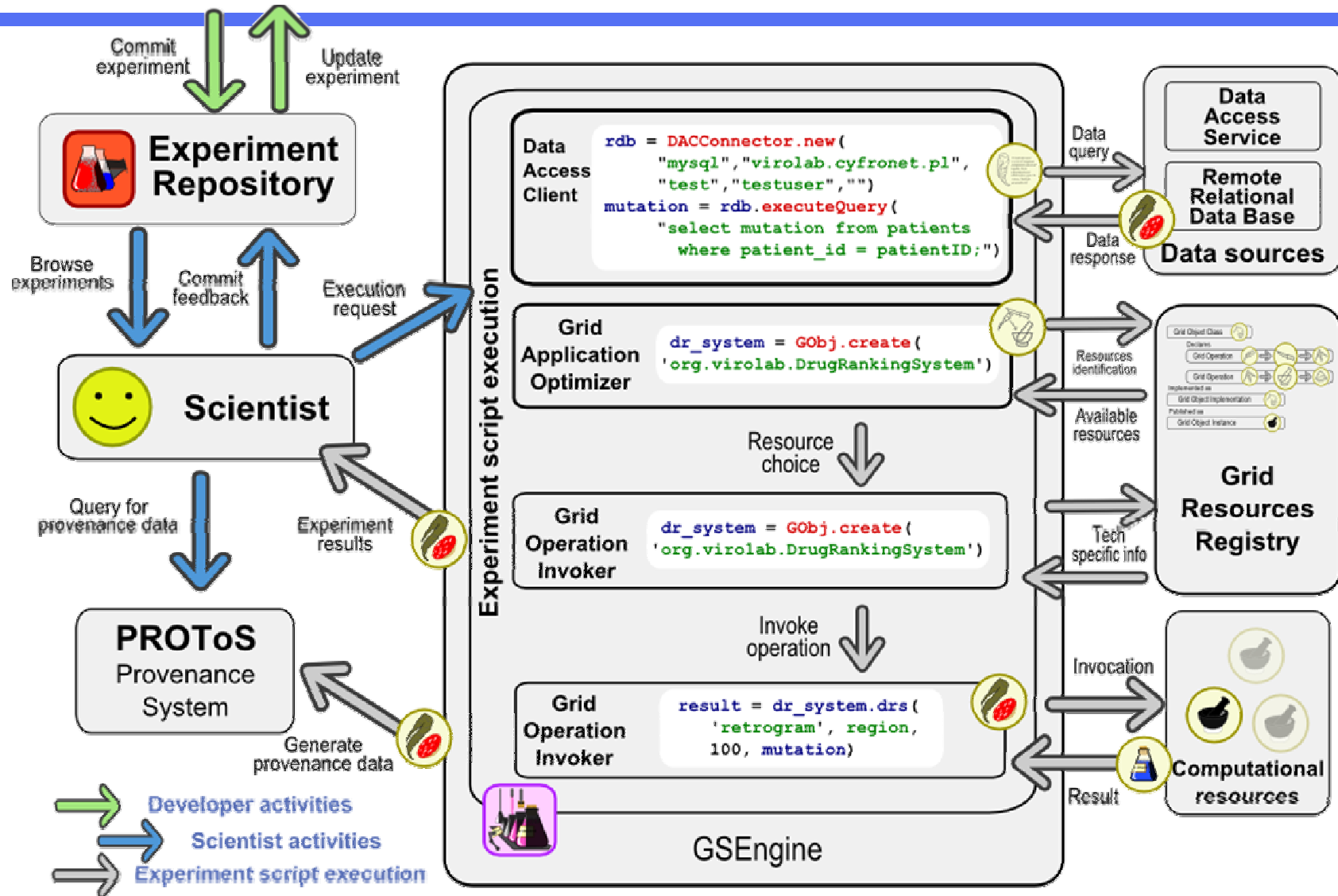
WS-VLAM composer



High-level Script Approach

- Application composition based on dynamic scripting language
- Rapid application development, prototyping, experiments (scientific applications)
- High-level API
 - Rich functionality (standard library), few lines of code needed
 - Easy to learn, clear and readable code
 - Supports full set of control structures (high expressiveness)
- Candidate languages
 - Python, Ruby, Lua, Perl ...
- Solution: JRuby
 - Object oriented, clear syntax, dynamic
 - Integration with Java (important for use of existing technologies)
- Uniform interface to computational resources provided the Grid Operation Invoker library – **high level of abstraction**

Collaborative Applications on VLvl



Scripts vs. workflows (1/2)

Features	Scripts	Workflows
Hierarchical constructs	classes, methods, blocks	subworkflows
Dependency detection	requires static analysis or runtime instrumentation	directly supported
Control structures	rich set of control structures	Multiple workflow patterns
Evaluation of expressions	supported by a standard library	external operations or specific modules
Data conversions	supported by a standard library	require external operations

Scripts vs wfs (2/2)

Features	Scripts	Workflows
Interactive execution	interactive interpreter	dynamic execution and refinement
Scheduling	requires dependency detection	dependency-based
Formal model	depends on a model of a specific script language	various reasoning and refinement techniques
Checkpointing	interpreter state is difficult to save	can be saved persistently
Automatic composition	advanced code assist in development tools	possible to automatically construct abstract workflows
Execution engine	based on a standard interpreter	need a specialized execution engine

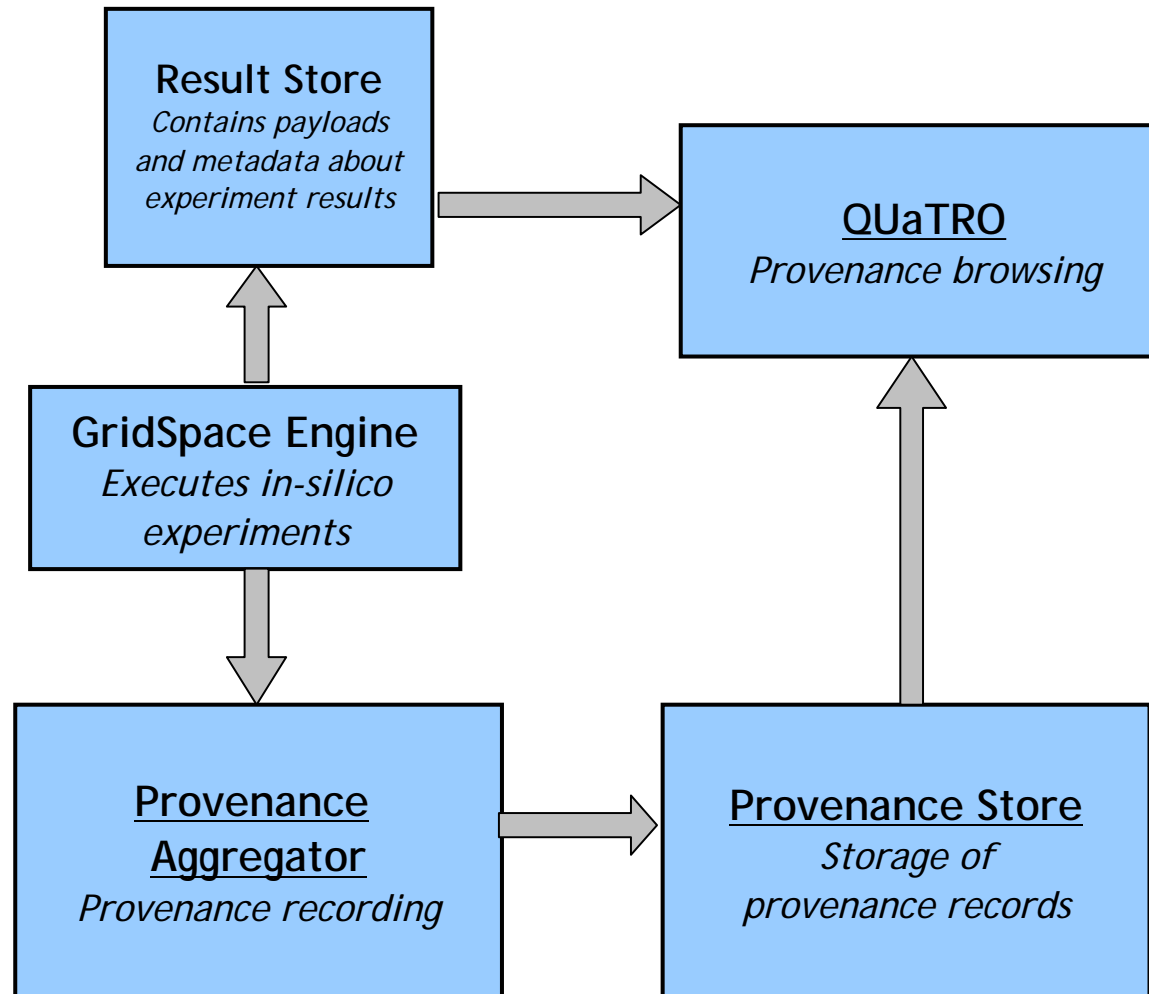
Provenance of virtual experiments

- **Provenance** = metadata describing records of past experiments
- What experimenters can learn from provenance:
 - Who else performed a similar experiment?
 - How was a particular result obtained?
 - What results were derived from a piece of data? Etc.
- Three aspects of a solution for provenance:
 - Provenance **model** (how records are represented)
 - Provenance **recording** (capturing essential events)
 - Provenance **browsing** and **querying** (enabling users to query provenance data and to view it, e.g., as dependency graphs)

Objectives and employed solutions

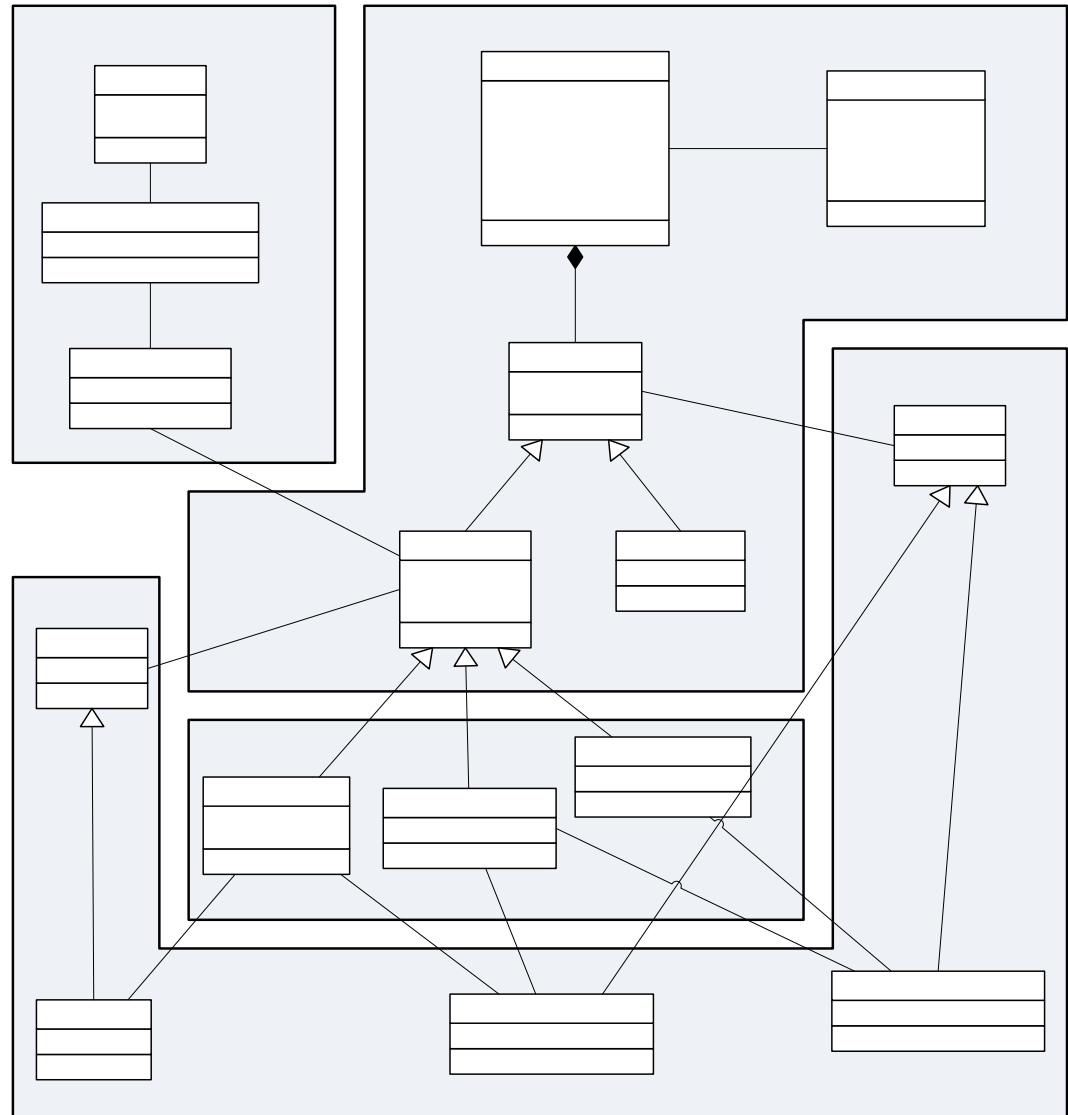
- Objectives:
 - To provide a provenance model enabling complex queries including querying by domain semantics
 - To enable end-users to construct complex queries with minimum knowledge of underlying data model or database technologies
- Solutions:
 - **Provenance model:** based on Semantic Web technologies (OWL ontologies, RDF representation)
 - **Provenance recording:** capturing events from GridSpace engine, translation to OWL/RDF
 - **Provenance querying:** visual web-based tools, ontologies as visual querying language

Provenance in GridSpace VL

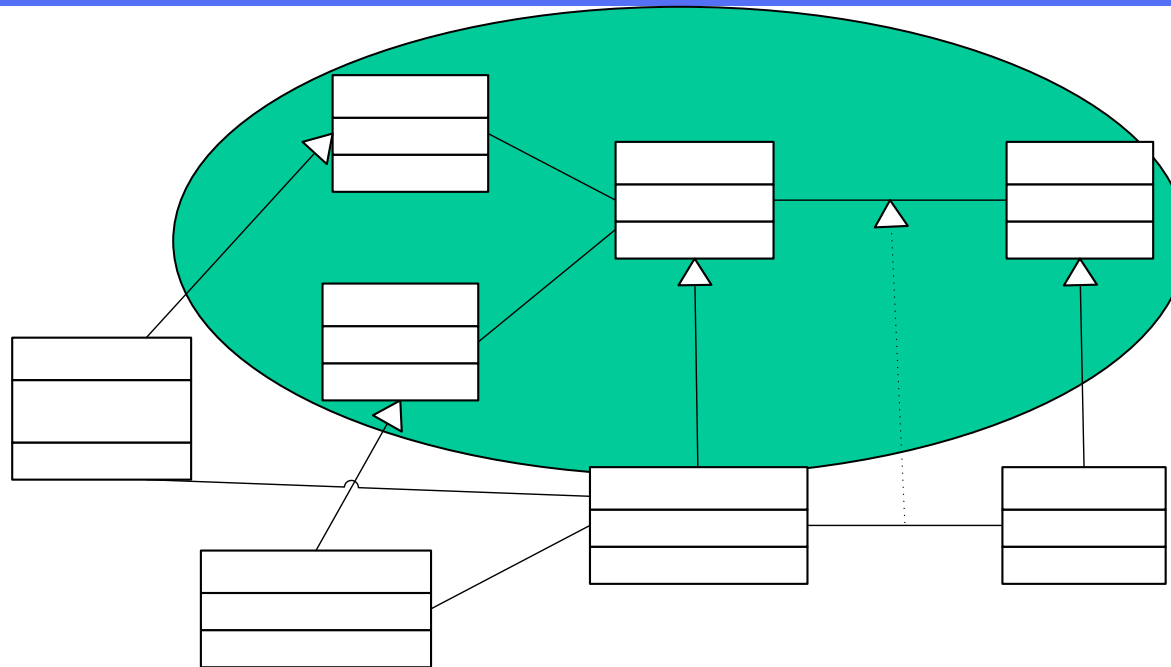


Provenance model

- Complex graph of ontologies
- Ontologies serve as:
 - Data model to structure provenance info and capture domain semantics
 - Querying language for end users
- Ontologies enable visual end-user oriented semantic querying



Example provenance record



- Representation of a **single computation of a new drug ranking**
 - **Two inputs, one output**
- Generic level: 'Computation', 'DataEntity'
- Domain semantics level: 'NewDrugRanking', 'RuleSet', 'DrugRanking', 'VirusNsSequence'
- Domain ontology concepts are connected to generic ontology concepts via **gen-spec relationship**

Provenance querying capabilities

- Obtaining provenance data dependency graph
 - Derivation path of how an experiment result was computed
- Semantics-level or generic level queries
 - Full semantics: *'NewDrugRankings whose usedRuleset was a Ruleset'*
 - Only data semantics: *'Computations whose inputData was a Ruleset'*
 - Only computation semantics: *'NewDrugRankings performed by...'*
 - No semantics: *'AnyDataEntity whose inputData ...'*
- Reasoning over subClass (gen-spec) relationships
 - Query for *'AnyDataEntity'* automatically extends to all subclasses
 - (domain ontology concepts retain all behavior of their super-class concepts)

QuATRO - query and results

Query Tree

Sort Ascending

ViroLabDataEntity inputData Experiment [select please...]





and and and

note = Mutations: I54M,V82A,I

and and

Query Results

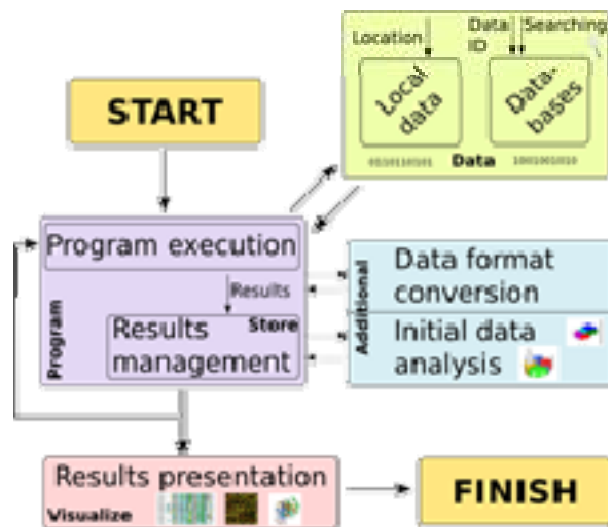
Recorded Provenance Information:

format	note	id	creator	timestamp	Provenance	Derivative
text	Mutations: I54M,V82A,I84V	https://virolab.cyfronet.pl/we	malawski@cyfronet.pl	2009-05-13T14:20:08.660		
text	Mutations: I54M,V82A,I84V	https://virolab.cyfronet.pl/we	balis@cyfronet.pl	2009-05-13T14:35:50.711		

Rows 1-2 of 2

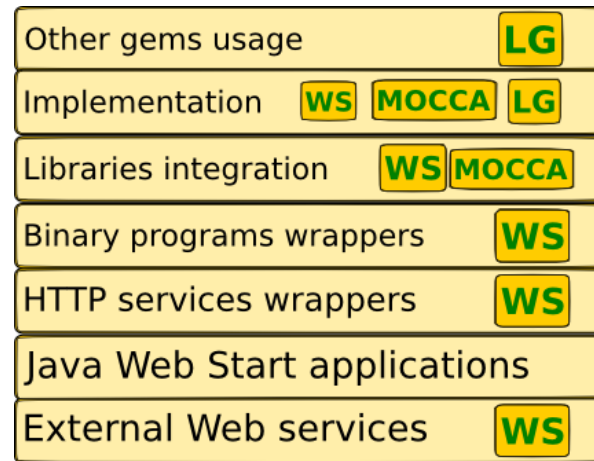
Bioinformatics Applications and Gems

- General model of bioinformatics experiment

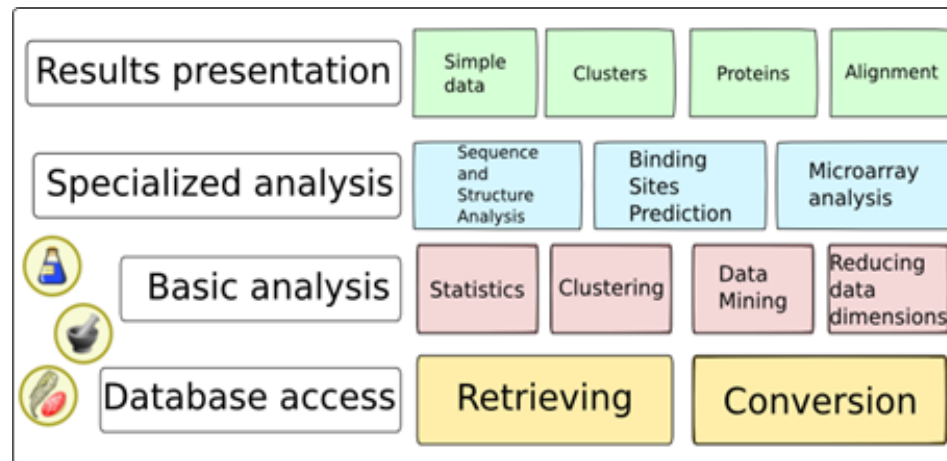


- Gem scope of usage
 - Database access
 - Basic analysis
 - Specialized analysis
 - Presentation

- Bioinformatics gem technologies



- Web service (WS)
- MOCCA component
- Local gem (LG)



Scientific Contribution

Development of collaborative applications

- A new method of collaborative application development
- Abstract layers to hide technological changes
- Semantic description of applications
- Integration of provenance recording and tracking

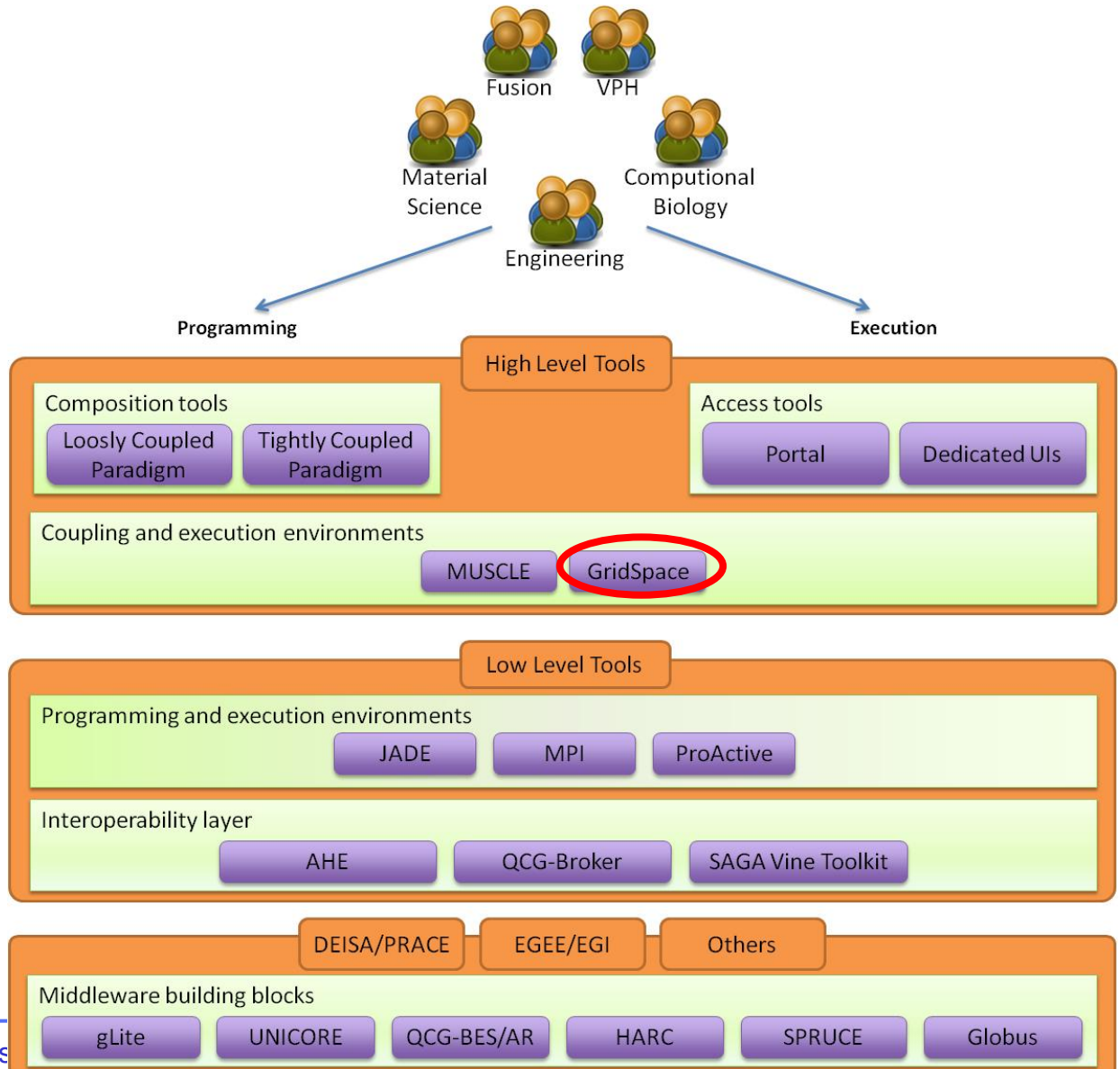
• Constructing the virtual laboratory

- Semantic description of resources
- Deployment on available Grid systems, clusters, and single CEs
- Integration of WS, WSRF, components, jobs
- Secure data access and integration
- Software engineering aspects

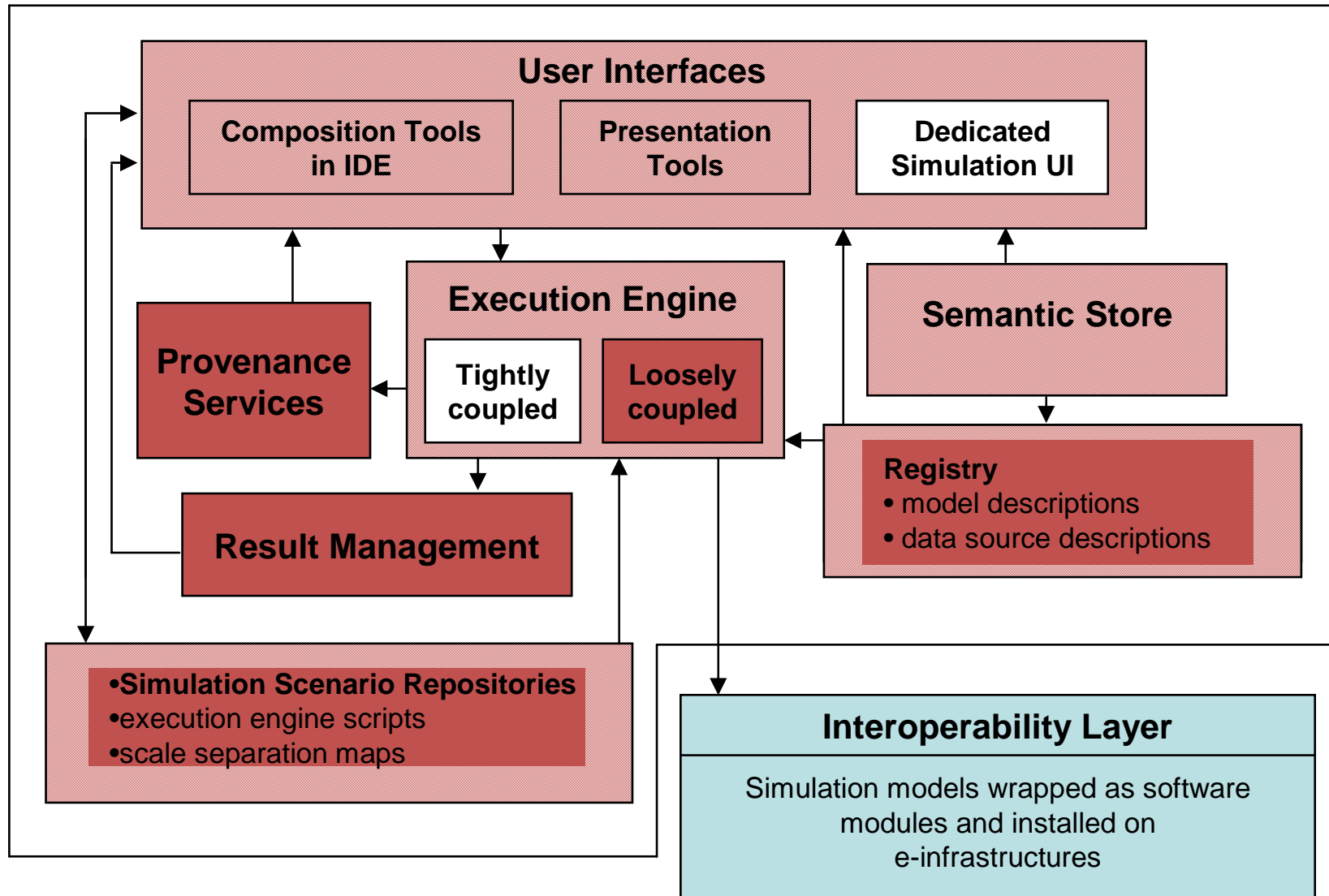
GridSpace for multiscale and multiphysics

For composing multiscale simulations from single scale models that:

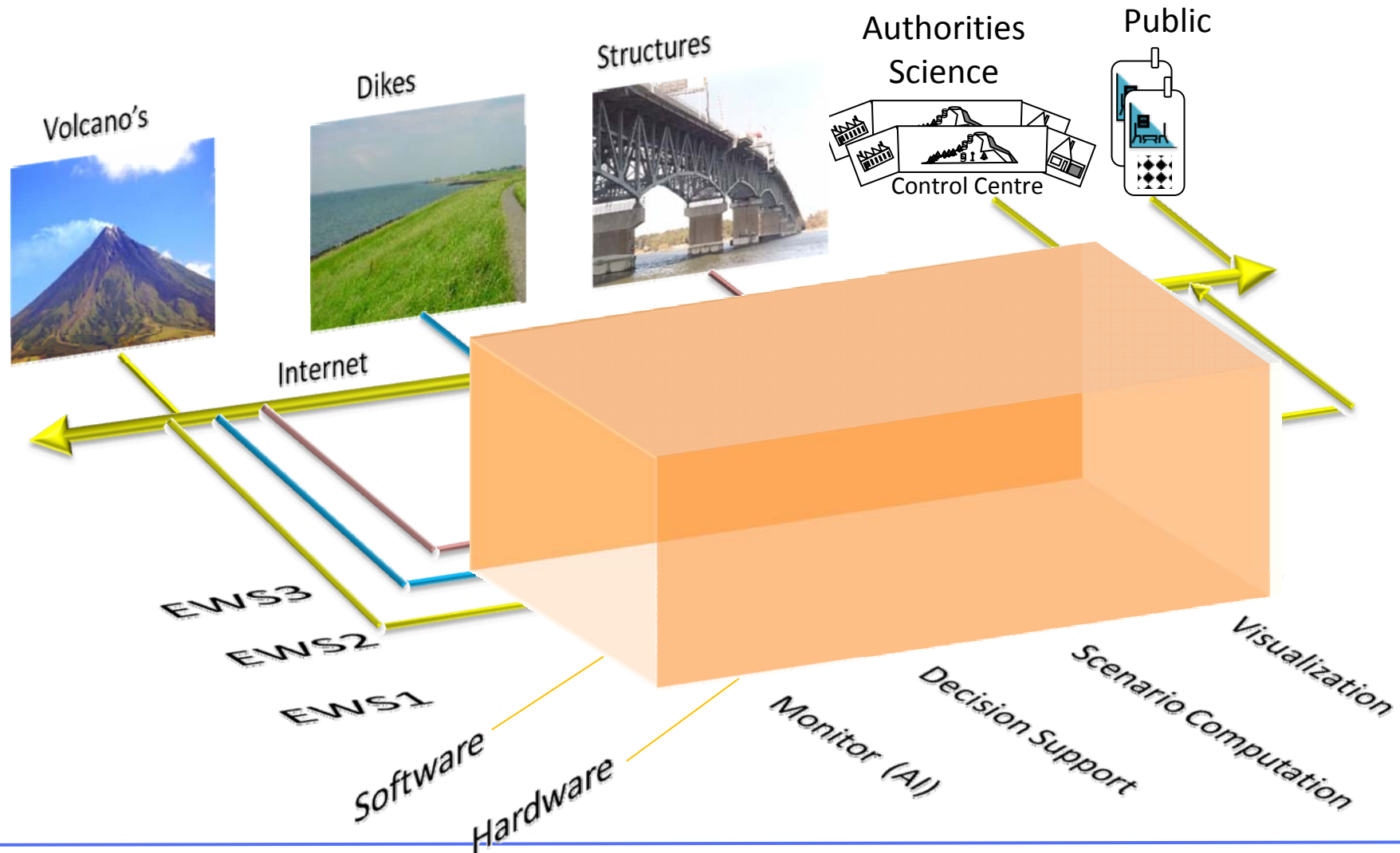
- are encapsulated as scientific software components
- distributed in various e-nfrastructures
- support the two main paradigms of multiscale computing: loosely coupled and tightly coupled.



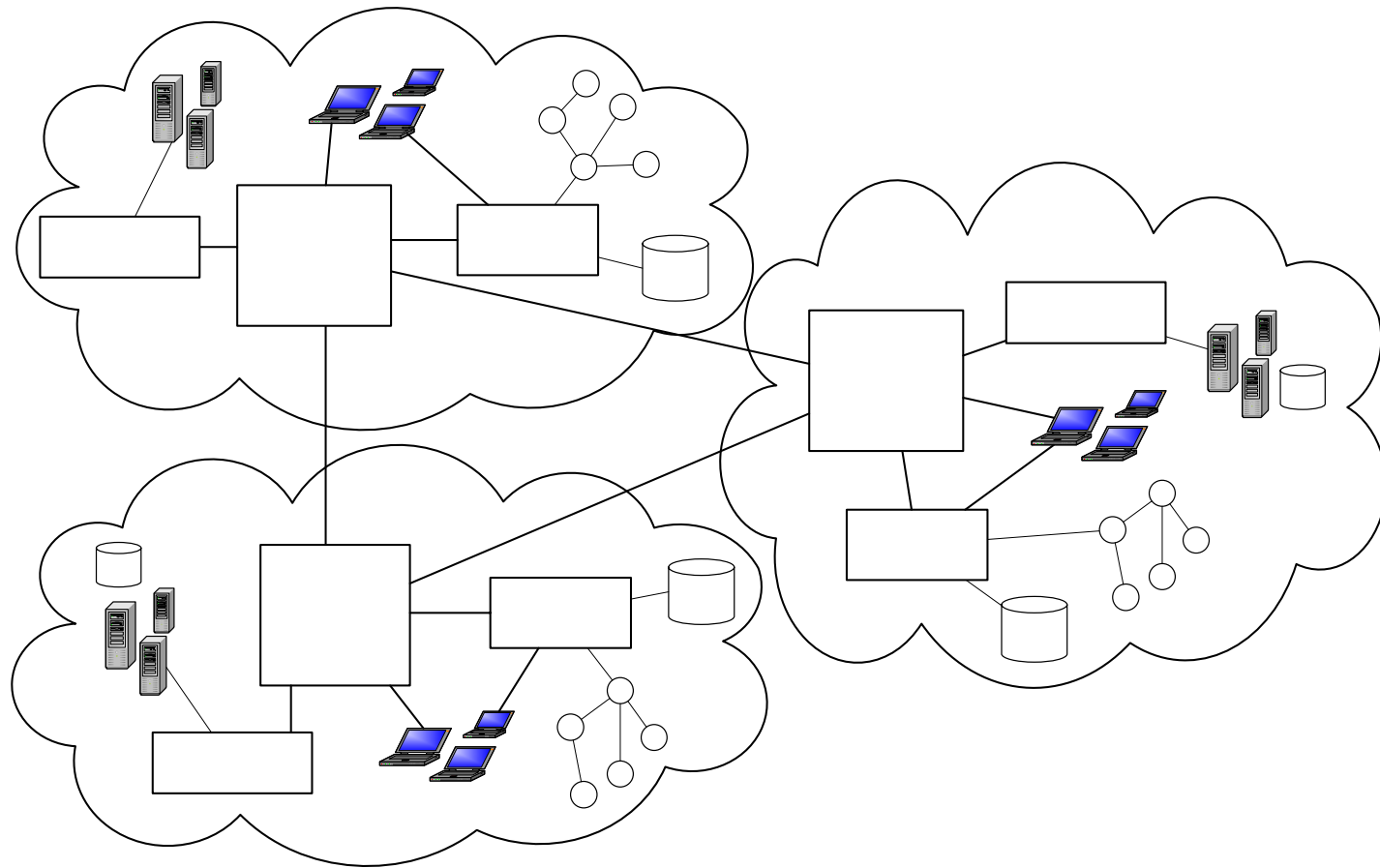
GridSpace modules for ms-mp



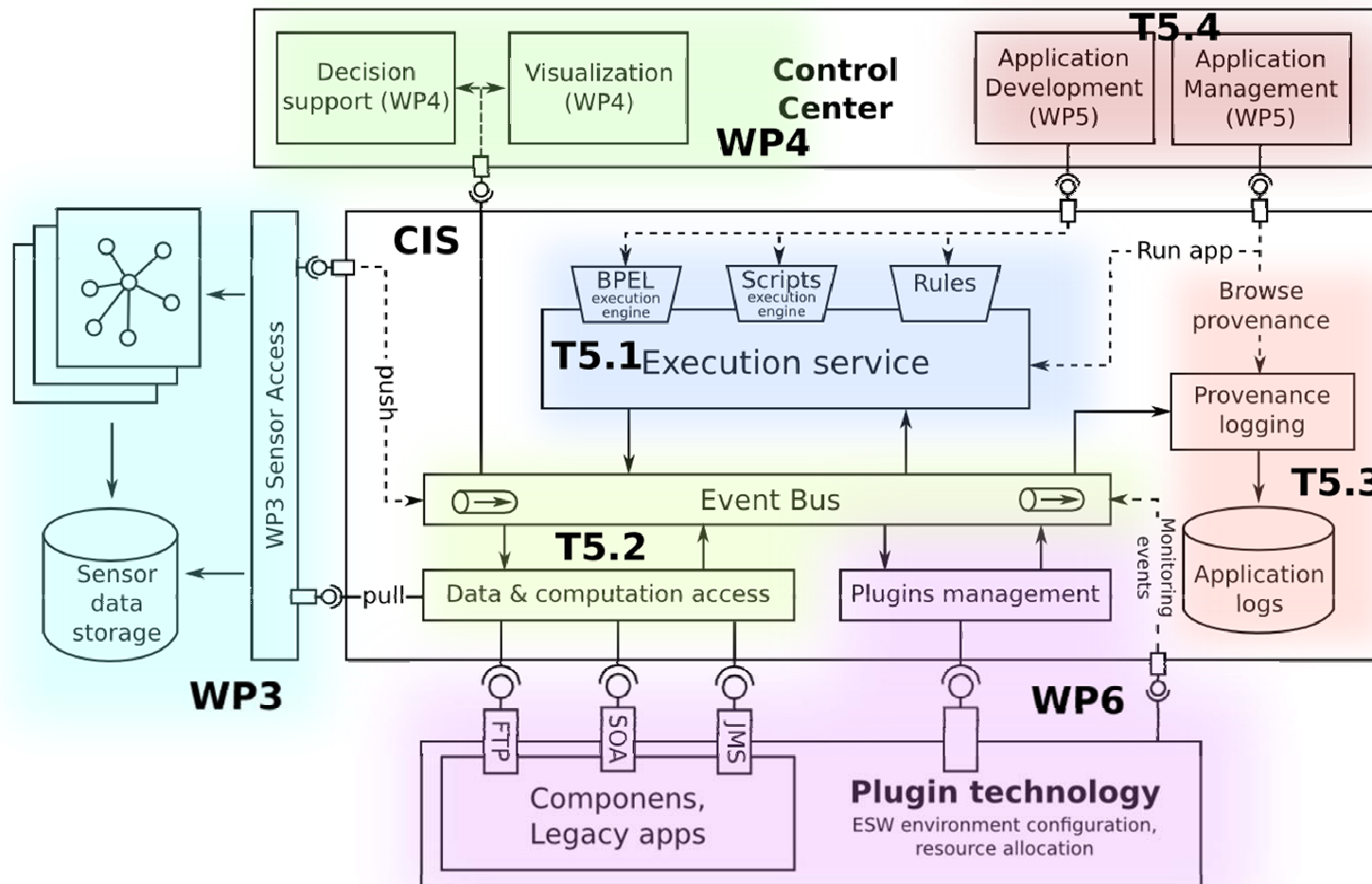
UrbanFlood System - Overview



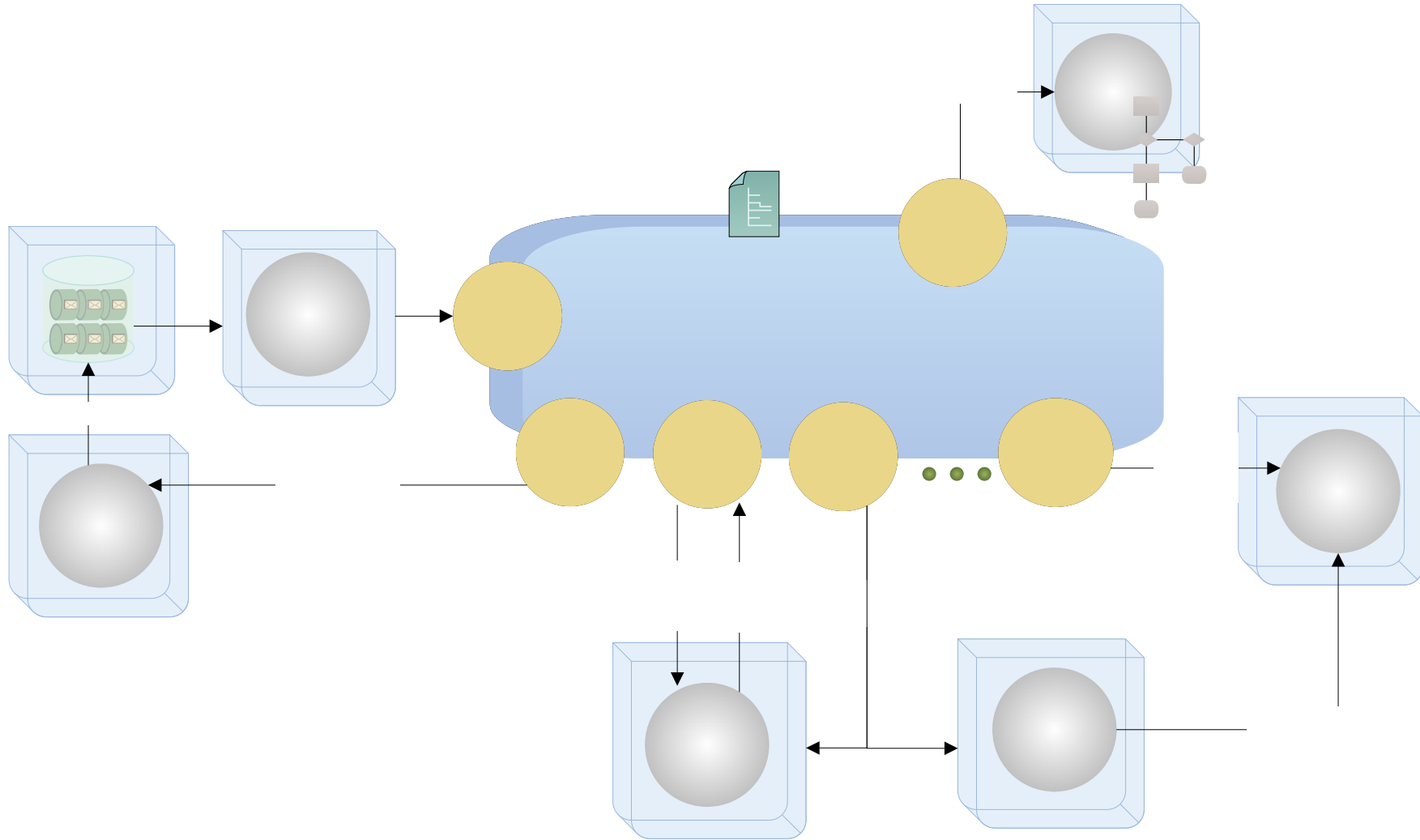
Common Information Space UF



CIS architecture



EWS components integration with CIS

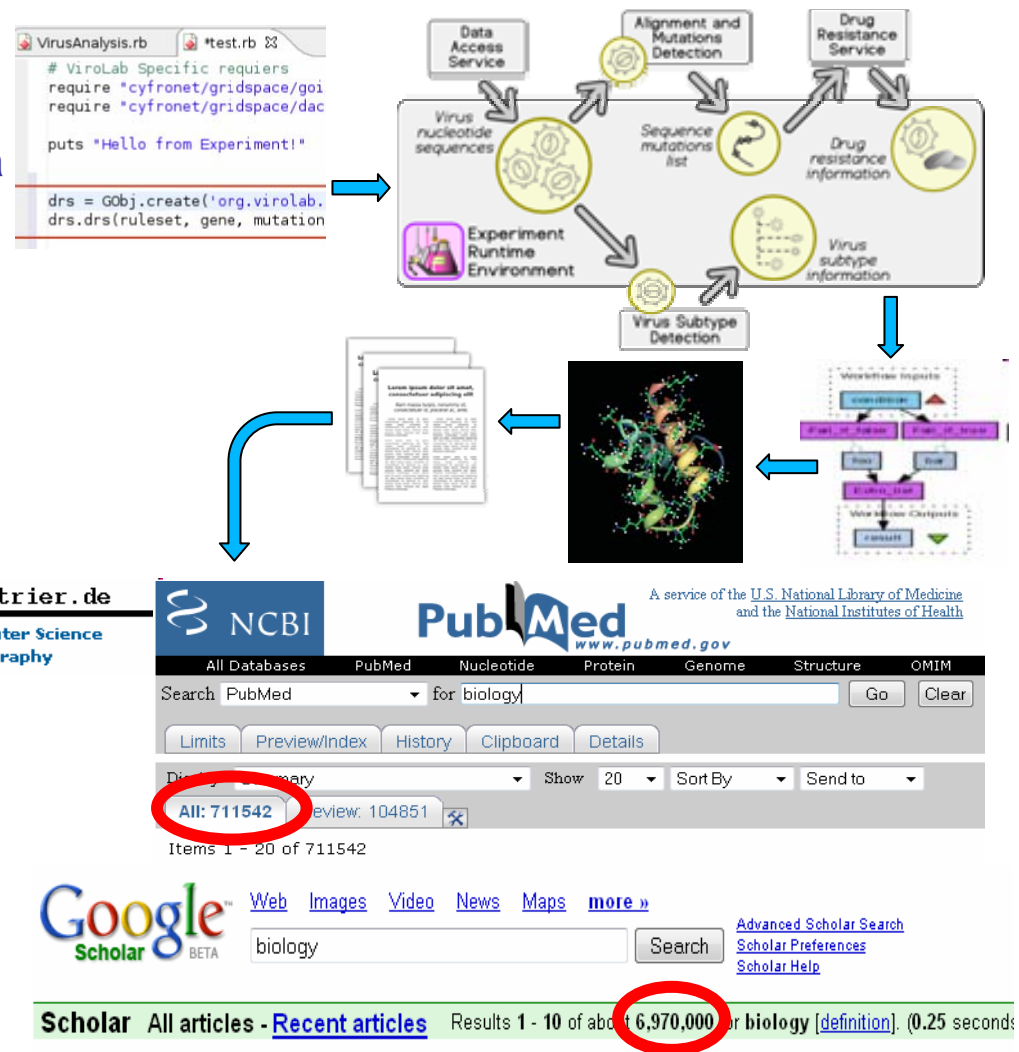


Open Science & Science 2.0

- ◆ New means of scientific communication:
 - ◆ Wikis, blogs
 - ◆ collaborative web 2.0 technologies
- ◆ New methods of conducting science:
 - ◆ e-science,
 - ◆ in-silico experiments,
 - ◆ exploratory applications
- ◆ Democratization of science
- ◆ Increasing role of openness

e-Science, Data and Publications

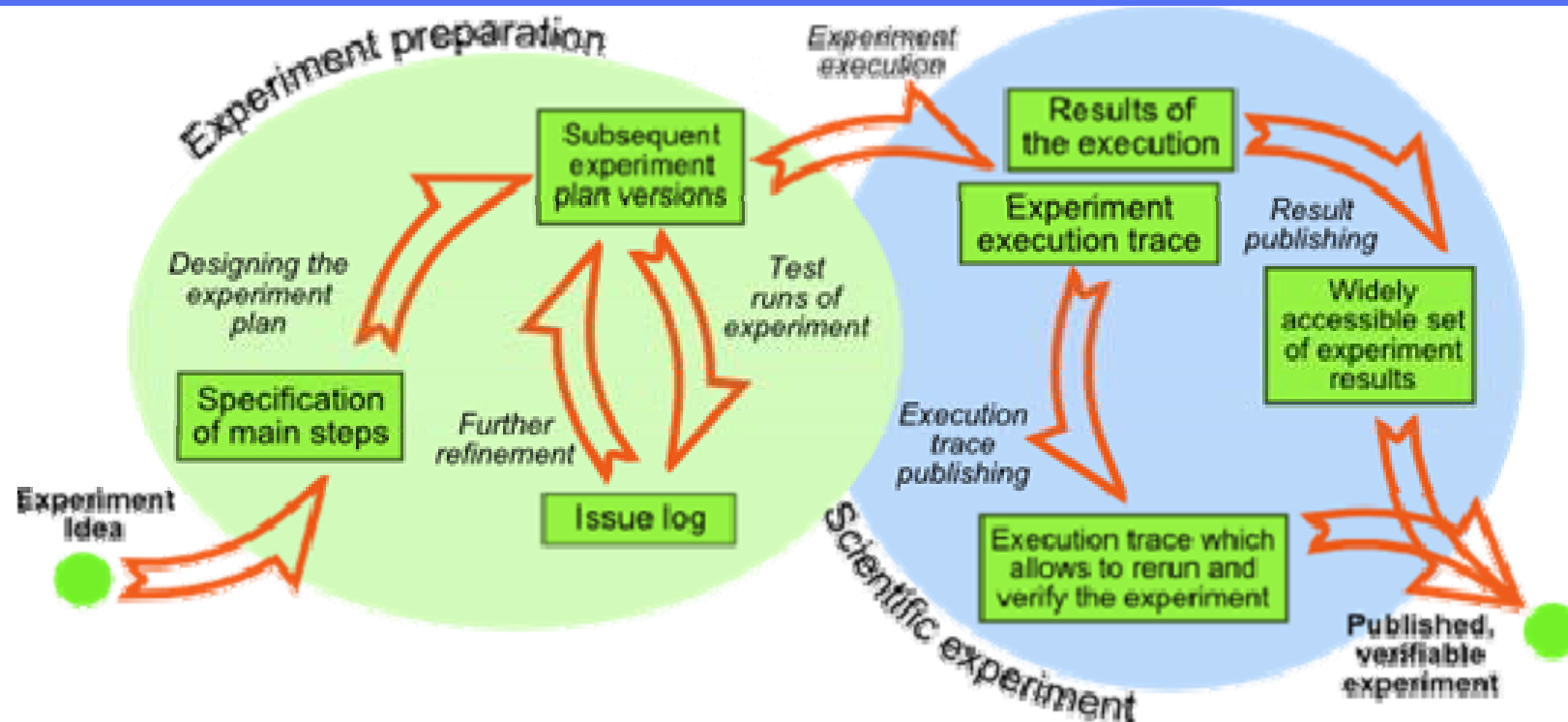
- Reproducible experiments, provenance in e-Science;
- Need to link publications with primary data (experimental data, algorithms, software, workflows, scripts);
- Plentitude of scientific software: jobs, workflows, services, components, scripts, experiment plans;
- Huge amount of scientific data consumed and produced by e-Science
 - Earth and life Sciences, HEP, etc.;
- Large number of publications makes research difficult:
 - Computer Science: DBLP contains more than $2^{20} = 1,048,576$ publications,
 - PubMed stores ~17 million articles to date,
 - CM digital library, ISI Web of Knowledge, Scopus, Citeseer, arXiv, Google Scholar;
- Emergence of the Web 2.0-based Scientific Social Community (SSC) model.



Challenges

- ◆ **Theoretical:** A model and methods for referencing primary data (experimental data, algorithms, software, workflows, scripts) as part of publications should be developed and integrated with modern e-Science infrastructures
- ◆ **Technological:** An integrated architecture for storing, annotating, publishing, referencing and reusing primary data sources. This architecture should span existing virtual laboratory and distributed computing systems

Experimentation Pipeline

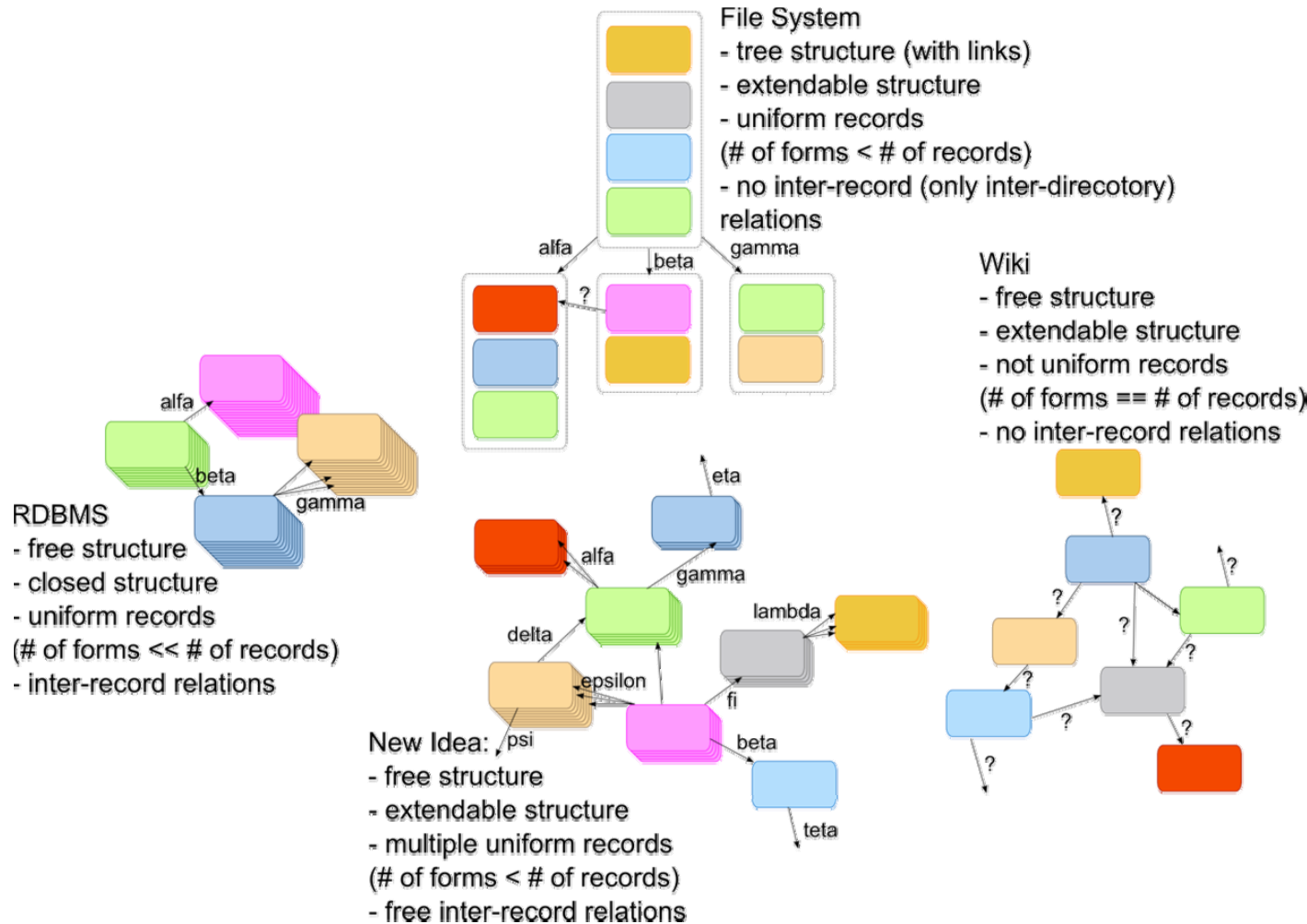


- ◆ **Phase 1:** Iterative experiment preparation
- ◆ **Phase 2:** Experiment execution involving semantic storage of results and ensuring repeatability

Sharing Primary Data: Data Nets

Data Net – unifying data storage mechanisms (relational databases, Grid-based file systems, Wiki pages etc.)

A **Data Net** is a group of data entities linked by named relationships. Such relationships impose a structure upon the dataset and facilitate querying for entities



Experiment Traces

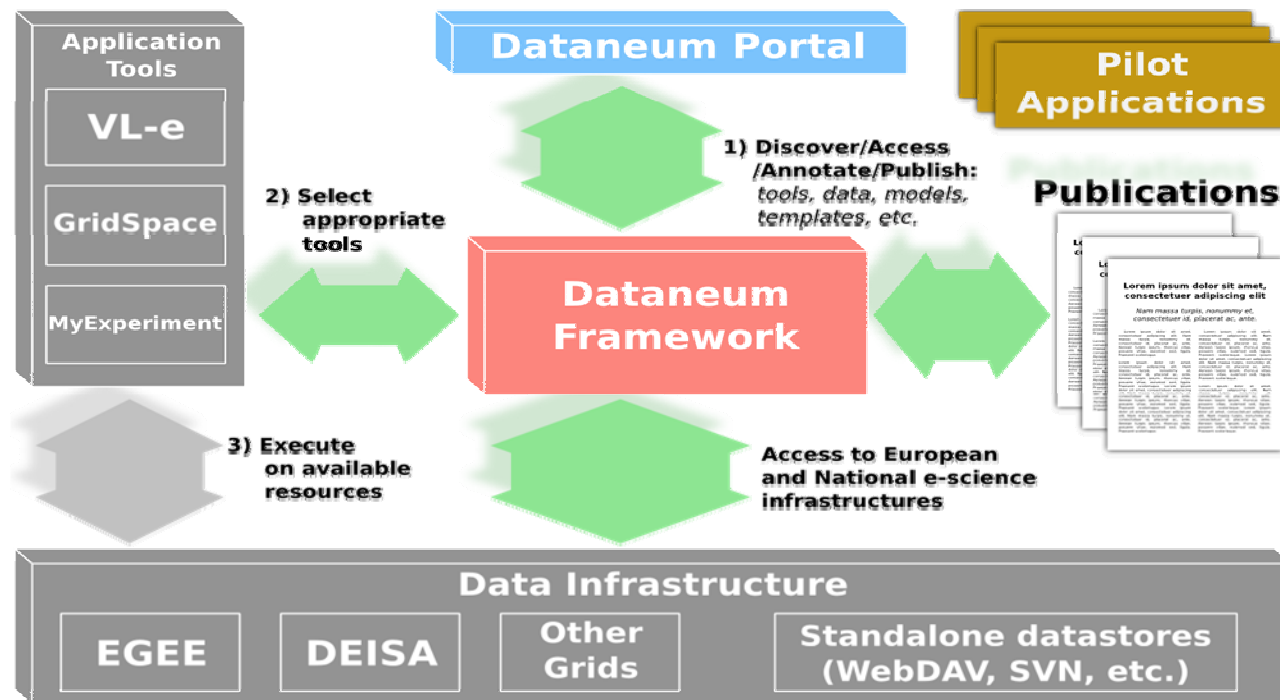
- ◆ An experiment trace consists of the following:
 - ◆ any input data provided by the experiment enactor;
 - ◆ all steps performed in order to transform this data into publishable scientific results (chronologically arranged);
 - ◆ the documentation of the experiment plan, prepared by a domain scientist (in the form of annotations and comments).
- ◆ The outcome of this process will be easily manageable and readable, similarly to weblog entries
- ◆ The system should enable enrichment of individual data elements with provenance information, linking them to appropriate stages of the experiment

Towards Integrated System

- **A model** enabling e-Scientists to expose the input data and results of their research to the wider scientific community without violating the security restrictions inherent in grid computing infra-structures or collaborative working environments

- **Integrated infrastructure and community web portal** for

- Authoring,
 - Publishing,
 - Managing,
 - Sharing,
 - Referencing,
 - Accessing,
 - Reusing,
 - Annotating,
- scientific data.



Challenges

- to go beyond today's social-oriented Web spaces by development of Web-based tools for managing laboratories, running experiments, gathering results, refining methods and achieving scientific goals within virtual groups,
- a model and formalism for specifying interaction schemes and a format for public description of component interaction schemes
- to support collaborative aspects of research through dedicated, specialized tools for different groups of users (administrators, hardware providers, experiment developers, scientists) to complete their tasks within the overall goal of their groups,
- a collaborative space for scientific collaborative application development, facilitating publishing and sharing of software content,
- a methodology of gathering and storing information about virtual laboratory users' experience by tracing their behavior

More

dice.cyfronet.pl