

# Status Report WP-VI

## *Grid Job Monitoring and Steering*

Thomas Radke (AEI)

4<sup>th</sup> AstroGrid Meeting at ZAH  
25 July 2006

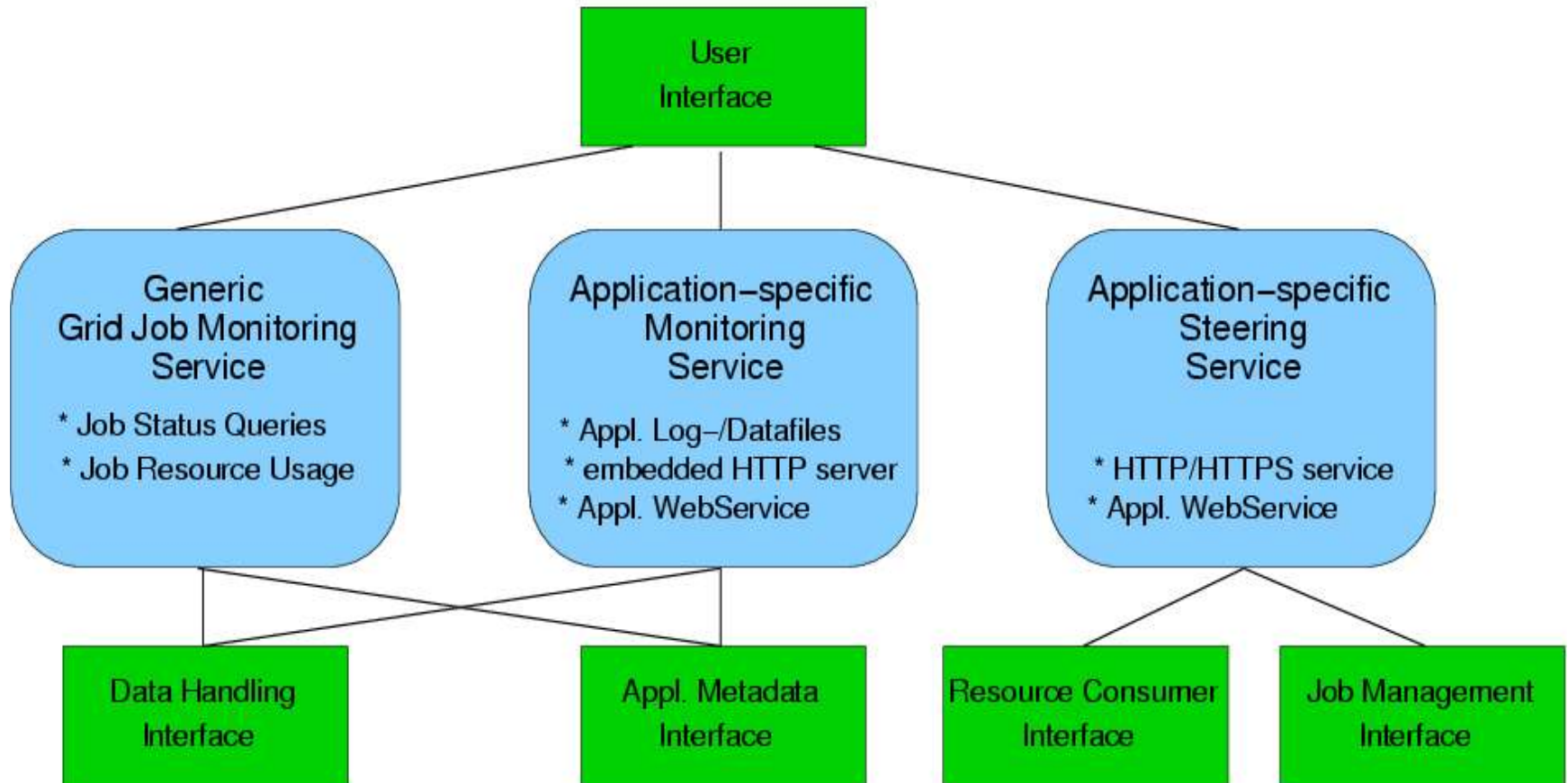
Quite some progress since the last meeting

- deliverable D6.1 is out, D6.2 is in the works
- Cactus application monitoring using WP-II's Information Service prototype

But also delays for various reasons:

- still incomplete Grid infrastructure (WP-I)
- dependencies on development of other services which are currently being designed/implemented by WP-II, WP-III, WP-V

## Application Monitoring and Steering Services



- Requirement: interactive access to a Grid job's ASCII logfile(s)
- Proposed solution:
  - ◆ equivalent to interactive logfile access  
-> same access methods could/should be used here
  - ◆ job registers itself and the list of its output files with WP-II's Information Service (IS)
  - ◆ portal then queries job metadata and presents it to the user
- Delayed until WP-II's first IS prototype is ready.



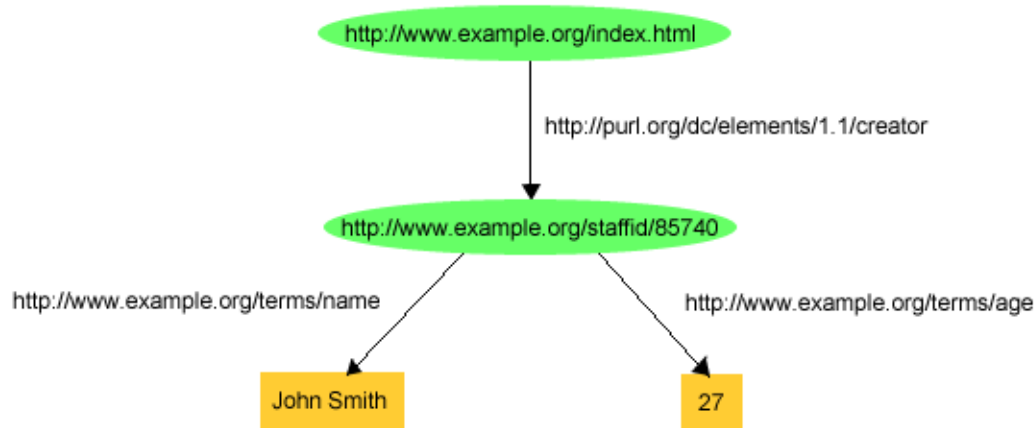
# “Simple” Application Monitoring and Steering



- Requirement: Controlled shut-down of AMIGA and NBODY6++ jobs
- Proposed solution:
  - ◆ job registers itself with WP-II's Information Service (IS)
  - ◆ portal then queries list of grid jobs and their location
  - ◆ on termination request by the user, issue command `globus-job-run <hostname> touch terminateAMIGA`
- Delayed until WP-II's first IS prototype is ready.



- RDF allows to make statements about resources, expressed as triplets: (subject, predicate, object); SPO
- O elements can take a constant value (literal), SPO elements can reference other nodes via URIs
- literals can have a predefined datatype





# RDF Schema



- RDF defines the syntax for how to make statements but not a vocabulary
- RDF Schema provides facilities to build a vocabulary by defining classes and named properties of classes
- subjects and objects are instances of classes, predicates are instances of class properties
- properties have a domain (the URIref to all classes with that property) and a range (the datatype that an instance of this property can take)



- There exist RDF vocabularies for various applications (eg. Dublin Core, FOAF, DOAP) but none to sufficiently describe metadata of AstroGrid-specific applications
- GLUE Schema (Grid Laboratory Uniform Env.) to describe metadata about Grid resources and services (but not jobs)  
-> interesting for WP-V: GLUE-to-RDF translator
- everything else will probably be application-specific, eg. the proposal for a Cactus RDF vocabulary:

<http://www.cct.lsu.edu/~dstark/cctk/0.1/>



# Cactus RDF Vocabulary Example



## RDFS Class *Parameter* with properties *name* and *value*

```
<!-- a generic parameter class -->
<rdfs:Class rdf:ID="Parameter"/>

<rdf:Property rdf:ID="name">
  <rdfs:domain      rdf:resource="#Parameter"/>
  <rdfs:range       rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</rdf:Property>

<rdf:Property rdf:ID="value">
  <rdfs:domain      rdf:resource="#Parameter"/>
</rdf:Property>

<!-- an integer parameter subclass -->
<rdfs:Class rdf:ID="IntegerParameter">
  <rdfs:subClassOf  rdf:resource="#Parameter"/>
</rdfs:Class>

<rdf:Property rdf:ID="integerValue">
  <rdfs:subPropertyOf rdf:resource="#value"/>
  <rdfs:domain        rdf:resource="#IntegerParameter"/>
  <rdfs:range         rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/>
</rdf:Property>
```





# Cactus RDF Data Producer



- Thorn AEIThorns/Formaline implements a prototype of a **Cactus RDF data producer**
- Generates RDF/XML documents and posts them to IS
  - ◆ At startup it announces static simulation metadata
    - executable name + version, parameter filename
    - job owner, run date, run host, nprocs, cwd
    - contents of the parameter file
    - all thorns with all parameters and their values
  - ◆ update information: current physical time and iteration number, termination flag



- Each simulation stores all its metadata in the global RDF space under a unique ID:

<http://astrogrid.aei.mpg.de:24002/context/CactusSimulations/<jobID>/>

```
<cctk:Simulation rdf:about="<jobID>">
  <cctk:host>nidud.aei.mpg.de</cctk:host>
  <cctk:hasThornList rdf:resource="#ThornList"/>
```

- Graphs are mapped to subpaths

```
<cctk:ThornList rdf:about="#ThornList">
  <cctk:containsThorn rdf:resource="#Thorns/Boundary">

<cctk:Thorn rdf:about="#Thorn/Boundary">
  <cctk:hasParameter rdf:resource="#Parameters/Boundary/radpower">
```

- Validating RDF data
  - ◆ [W3C's RDF Validation Service](#)
    - output both as RDF/XML and graph
    - sometimes buggy (caching of old data)
  - ◆ [Redland Raptor RDF Parser](#)
    - only RDF/XML output but reliable
- Querying RDF data
  - ◆ [SPARQLer](#) - web-based general purpose processor
  - ◆ IS-internal query engine
  - ◆ [Twinkle](#) - standalone Java-based query tool



# Some Example Queries



A demo IS instance is running on <http://astrogrid.aei.mpg.de:24002> which provides some example Cactus RDF metadata.

These can be queried with the example SPARQL queries provided on our [Cactus RDF provider](#) WP-II intranet page.

1. all simulations, listed by parfile, owner, date
2. all parameters (by name and value) of a thorn
3. all WaveToy simulations (FILTER)
4. dynamic update/termination information (OPTIONAL)

*Feel free to test and modify these example queries !*



- IS must become scalable and more robust:
  - ◆ context sensitivity: how to search subgraphs only
  - ◆ in-memory storage -> database backend with backup
- How to deal with security
  - ◆ GSI is difficult to add to the existing Cactus infrastructure
  - ◆ possible solution: an IS server instance just for Cactus
    - simulations can post metadata anonymously from a restricted set of production machines, without GSI
    - accessing metadata (through a Cactus portal) requires a user/group certificate

- Application-specific user interfaces for queries
  - ◆ GridSphere portlets which interface to IS (via JSON?)
- Get the Cactus community actively involved
  - ◆ get some users' metadata from real-physics simulations
  - ◆ adapt user interfaces to users' needs  
(eg. predefined queries, fixed query forms)
  - ◆ provide a generic Announce API for Cactus  
to register event-based and other dynamic metadata

*This is an iterative process which requires close interaction between developers and users.*



# Collaborations



- Erik Schnetter, staff member at CCT / LSU Physics Dpt.
  - ◆ author of Thorn AEIThorns/Formaline
  - ◆ alpha tester of our services
- Dylan Stark, CCT student
  - ◆ designed RDF vocabulary to describe Cactus metadata  
<http://www.cct.lsu.edu/~dstark/cctk/0.1/>
- Ed Seidel, Director of CCT
  - ◆ will meet him tomorrow
- Tom Goodale, Cardiff University
  - ◆ chief Cactus developer, will also meet this week

