

Cactus Metadata Management

Thomas Radke (AEI)

5th AstroGrid Meeting at MPE Garching
14./15. November 2006

- Cactus users run many jobs on many different machines
 - a Cactus metadata management system helps scientists to keep a better overview on all their simulations
 - Where are my output files and results stored ?
 - What simulations are other people running ?
 - Has someone else already done the same simulation ?
 - How can I reproduce a specific run ?
- Cactus code development is very complex
 - automated integration tests help Cactus programmers in maintaining, verifying, and developing the toolkit



The AstroGrid Approach



1. Definition and description of Cactus metadata
2. Metadata generation
 - announcement of simulation metadata
 - automated Cactus integration tests
3. Storing metadata in a persistent datastore
4. Querying and presenting metadata to the end user



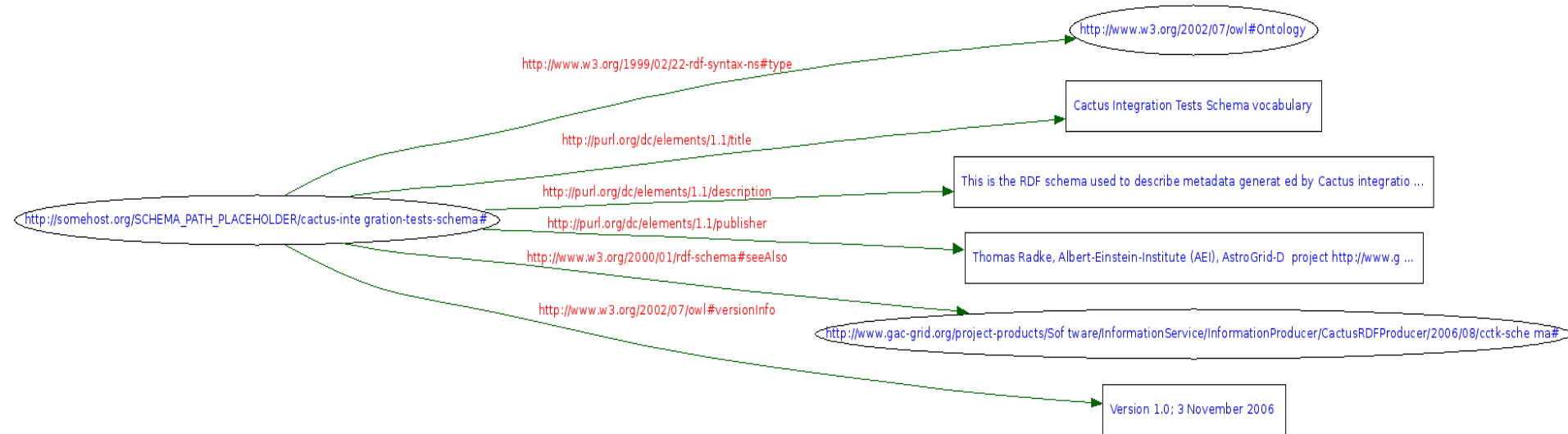
- Resource Description Framework (RDF) markup language is used to generically describe metadata
 - ◆ W3C standard technology for building Semantic Webs
 - ◆ extensive set of powerful tools and open source development libraries available
- RDF was chosen by WP-II to serve technical base for our AstroGrid information service
- *Icecore* prototype implementation of an RDF datastore is available already for development and testing



Cactus Metadata Description



- RDF tools provide a set of vocabularies to describe certain standard types of metadata
- application-specific metadata doesn't really fit into any of these, therefore a *Cactus metadata schema* was designed





Announcing Simulation Metadata



- Thorn `Formaline` implements a Cactus RDF data producer
 - ◆ generates RDF/XML with static simulation metadata and posts it to the AstroGrid information service
 - 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
- Thorn `Publish` provides a general Cactus API to announce user-defined dynamic metadata
 - ◆ periodic updates of important runtime information (eg. current physical time and iteration, termination cond.)





Automated Integration Tests (I)



combination of interdependent unit tests for a given Cactus configuration (thornlist, config options, #procs)

1. freshly checkout development version of Cactus and thorns
2. create a configuration with given machine-specific options: compiler settings, optimisation, MPI, external packages
3. build the configuration (run `make` to produce an executable)
4. build all utility programs
5. run all available testsuites on the given number of processors and evaluate their status





Automated Integration Tests (II)



- implemented as standalone perl script which runs as a nightly cron job on AEI's main production clusters
- integration test results are sent to an RDF server
`mintaka.aip.de:24006/context/CactusIntegrationTests/`
 - ◆ unique integration test description: configuration name, thornlist, options, user, host, #procs, datetime stamp
 - ◆ status of each unit test (passed / failed / skipped), `stderr` log for failed tests
 - ◆ status of all Cactus testsuite runs, in case of failure either with `difflog` or `stdout/stderr` log





Storing Cactus Metadata



- AstroGrid information service is used to store Cactus RDF metadata
- instances are running on AstroGrid testbed installations (`mintaka.aip.de`, `astrogrid.aei.mpg.de`)
- all Cactus metadata are stored under a unique context
- everything is publicly available for demo purposes
 - ◆ see description on our public AstroGrid webpages
<http://www.gac-grid.org/project-products/Software/InformationService/InformationProducer/CactusRDFProducer.html>
- IS instances with production datastore and restricted access for Cactus users are planned at AEI and CCT





The Cactus User Interface (I)



- users can easily access/query simulation metadata and integration test results through the *Cactus portal*
- a set of Cactus RDF/SPARQL portlets was developed and integrated into the *GridSphere* portal framework
- via persistent user preferences, an RDF storage server can be selected and metadata queries be refined
 - ◆ search only for specific configurations of individual users on given machines via `regex` pattern matching
 - ◆ limit query results to the N most recent entries in a list





The Cactus User Interface (II)



- the portal presents metadata in a clear structure
- the same metadata can be viewed in different ways
 - sorted table of most recent integration test results
 - test result details for a specific testsuite
 - history of this testsuite across all integration tests
- add-on functionality
 - ◆ compare Cactus parameter files and thornlists
 - ◆ users can edit existing and add new SPARQL queries
 - ◆ in the future: send user notification on selected events



- after 15 months into the project we can offer tools for our scientists to organise their scientific work more efficiently
- standard Grid and web technologies were taken and adopted for the specific requirements of Cactus
- the *Grid as such* is mostly hidden to the end user
- successful integrated effort by many people
 - ◆ WP-II: Metadata Management
 - ◆ WP-VI: Application Monitoring and Steering
 - ◆ WP-VII / DGI: User Interfaces
 - ◆ Cactus collaborators at CCT, Cardiff Univ. and AEI



Open Issues



- improve packaging and robustness for production use
- increase performance and scalability for large sets of metadata
- knowledge dissemination
 - ◆ AstroGrid Metadata Management workshop for other D-Grid partners
- secure access to metadata: vertical integration of metadata and VO management in the portal
 - ◆ will be one of the goals in the D-Grid2 project *D-Mon*

