New resource allocation service in GridNNN

A.Kryukov*, A.Shamardin*, A.Demichev*, V.Ilyin*, N.Prihodko**

*SINP MSU, **NovSU
GRAM service in GT4/5

- Register task on the resource
- Stage-in/out files
- Transfer task to local resource manager
- Monitoring task
- Manage task (for example cancel the task)
- ... and more
GRAM GT4

- GT4
  - WSRF service
  - WS-MDS as a information system
- GT5
  - Really this is reincorporation of GRAM GT2
GridNNN

Project was started in 2008.

- Main target of the GridNNN is to provide access of scientists and engineers to the supercomputer resources of National Nanotechnology Network
- This is computational grid
- Joint supercomputers
- Application area is nanotechnology, aero- and hydrodynamics
The main object which GridNNN manipulate is job.

The job describes by direct acyclic graph (DAG)
- vertexes are a set of tasks
- edges are logical dependencies among them.
- Each task must be executed on some single resource (supercomputer). So it consider as an atomic object on the grid level.

This interpretation of task is very close to the interpretation of the similar object in other grid project, for example EGI/WLCG.
Job structure in GridNNN

```json
{
    "version": 2,
    "description": "task description",
    "tasks": [
        { "id": "a",
          "children": [ "b", "c" ]
        },
        { "id": "b",
          "children": ["d"],
        },
        { "id": "c",
          "children": ["d"],
        },
        { "id": "d",
        }]
}
```
New concept of task

- We consider the task as a composite object which is DAG itself.

- The vertexes of this graph may be two types.
  - First type vertex is a executable object which can be submitted to local resource manager and execute.
  - The second type is an auxiliary vertex which define some action on the grid-gate where service are working.
  - The set of such vertex is limited and should be predefine by system administration of the service.

- Second type of vertex is realized as a plug-ing component of the service with clear and simple API.
New task structure

- Two set of vertexes
  - Execution vertex (e1,...)
  - Service vertex (s1,...)
- Example: select execution module (e2, e3) depends on success e1.
  - [“RC > 0”, e2, e3]
Problem of compilation of parallel task

- It is necessary compile parallel program on the same resources where it should execute
- Special node for compilation (not in common pool working node)
- Current infrastructure can not do it because after completion task all information delete.
- In new approach just chain of subtask: compilation -> execution
Conclusions

- In the present time we have prototype of the allocation service which is under testing.
- It provide full set of services which has necessary functionality like authentication, authorization, information service, accounting, submission service and others.
- The service can also be used by user to remote access to supercomputer resources directly. So it can be adopted to non-grid distribution calculation.