UNICORE Desktop Grid

Jakub Jurkiewicz, Piotr Bała

Interdisciplinary Center for Mathematical and Computational Modelling,
University of Warsaw

Unicore Summit 2010
18-19 May 2010
Outline

1. Dekstop Grids, Community Grids

2. Architecture of UNICORE desktop grid
   - Communication UNICORE desktop grid
   - Security

3. Simple efficiency test

4. Conclusion and future work
Community Grids

- Example Seti@Home
- The most popular middleware BOINC
- Huge number of working nodes - Internet scale
- Disadvantage - hard job submission
Desktop Grids

- The most popular middlewares based on BOINC
- Not that many working nodes - institution scale
- Easy job submission
Why we need desktop grid based on Unicore?

- Connecting to normal Unicore Grid
- Unicore authentication and authorisation
Outline

1. Desktop Grids, Community Grids

2. Architecture of UNICORE desktop grid
   - Communication UNICORE desktop grid
   - Security

3. Simple efficiency test

4. Conclusion and future work
Not Unicore modules

- HTTPs distributed proxy
- Desktop grid manager
Https distributed proxy

- server is waiting from connections from clients and nodes
- node side connects to server and registers which address should be redirected to it
- allows running nodes behind NAT and firewall
Windows desktop grid node

- Userspace
  - Scripts
    - Activating
    - Deactivating
  - ScreenSaver

- TCP IP
  - Activate
  - Deactivate
  - Status

- Windows service
  - Start Process
  - Kill process

- UDG Computing Node software
  - UniocreX
  - Gateway
  - Distributed HTTPSPProxy
Connecting node to desktop grid

UDG manager
UCC client libraries
Unicore registry on manage
Storage On Manger
Worker node UnicoreX

Register worker node
Connecting node to desktop grid - httpsProxyView
Connecting node to desktop grid

UDG manager
UCC client libraries

Unicore registry on manage

Storage On Manger

Worker node UnicoreX

Register worker node
Discovering working nodes

UDG manager
UCC client libraries

Unicore registry on manage

Storage On Manger

Worker node UnicoreX

GetListofActiveNodes

Register worker node
Communication desktop grid manager node - httpsProxyView

HttpsDistributedProxy Manager part

HttpsDistributedProxy Worker node part
Communication desktop grid manager node - httpsProxyView
Checking node state - get node time

UDG manager
UCC client libraries

Unicore registry on manage

Storage On Manger

Worker node UnicoreX

GetListofActiveNodes

Register worker node

Check if node is active

Conclusion and future work
Running job

UDG manager
UCC client libraries

Unicore registry on manage

Storage On Manger

Worker node UnicoreX

- GetListofActiveNodes
- Register worker node
- Check if node is active
- Submit a job

Simple efficiency test

Conclusion and future work
Getting data for computations

UDG manager

UCC client libraries

Unicore registry on manage

Storage On Manger

Worker node UnicoreX

GetListOfActiveNodes

Register worker node

Check if node is active

Submit a job

GetData
Checking computations state

UDG manager

UCC client libraries

Unicore registry on manage

Storage On Manger

Worker node UnicoreX

GetListofActiveNodes

Register worker node

Check if node is active

Submit a job

Check job state

GetData
Storing results by the node
Complete communication diagram

UDG manager
UCC client libraries

Unicore registry
on manage

Storage On Manger

Worker node
UnicoreX

GetListOfActiveNodes

Register worker node

Check if node is active

Submit a job

GetData

Store results

Check job state

Check job state

Conclusion and future work
Grid manager uses his own certificate. In Future we could use explicit trust delegation for allowing running proper jobs on proper machines.
Nodes use one certificate (they are identical). We assume that in future nodes would be allowed to use more than one type of certificate. This would allow users to run jobs to only part of Desktop Grid - security of data.
Job while connecting to data storage uses any certificate provided by manager in job description. Now it’s one certificate for all, but it could be easy changed.
Node security

Actually all data stored now on disk are for:

- running job in UnicoreX: stdin, stdout, stderr – not job date
- work of Unicore - logging
Outline

1. Dekstop Grids, Community Grids
2. Architecture of UNICORE desktop grid
   - Communication UNICORE desktop grid
   - Security
3. Simple efficiency test
4. Conclusion and future work
Description

- tests before introducing https distributed proxy
- full availability of computers
- job - naive implementation of concurrent Mandelbrot set computing algorithm - pure balancing - may need more tasks than nodes
- jobs that took 236 seconds on a single computer
Time of computations for constant number of nodes

- **Total execution time (s)**
- **Number of sub tasks**: 2 nodes, 4 nodes, 7 nodes

The graph shows the total execution time in seconds for different numbers of sub tasks and node counts. Each line represents a different node count, with 2 nodes in red, 4 nodes in green, and 7 nodes in blue. As the number of sub tasks increases, the total execution time increases for all node counts, but the rate of increase varies. The 2 nodes line shows a sharp decrease initially before stabilizing, while the 7 nodes line shows a more gradual increase.
Time of computations for constant number of tasks

![Graph showing total execution time (s) vs. number of nodes for different numbers of subtasks.]

- 2 subtasks
- 6 subtasks
- 9 subtasks
- 16 subtasks

- The graph illustrates how total execution time decreases as the number of nodes increases for different numbers of subtasks.

Conclusion and future work
Scalability - speedup in test system

![Graph showing scalability with speedup as a function of the number of nodes. The speedup increases with the number of nodes, reaching a plateau after a certain point.](image-url)
Outline

1. Desktop Grids, Community Grids
2. Architecture of UNICORE desktop grid
   - Communication UNICORE desktop grid
   - Security
3. Simple efficiency test
4. Conclusion and future work
Conclusions

- It is possible to create Desktop grid using Unicore middleware.
Future work

- Efficiency tests using more computing nodes in real environment with test jobs
- Deployment of real application, without UI
- Creation UI for submitting jobs
- Creating multi manager desktop grid.
Acknowledgements

Work supported by:


- the project kardionet