- DRAFT – Version 14.2.2014 -

Master Course in Distributed Computing Systems Engineering

Brunel University – West London
&
Technische Akademie Esslingen

Course 2013/2014

Workshop WS5

Grid Programming with GridGain
(formerly Globus Toolkit 4 - GTK4)

Friday 14/2/2014 – Saturday 15/2/2014

Lecturer M. Pyschny, MBA

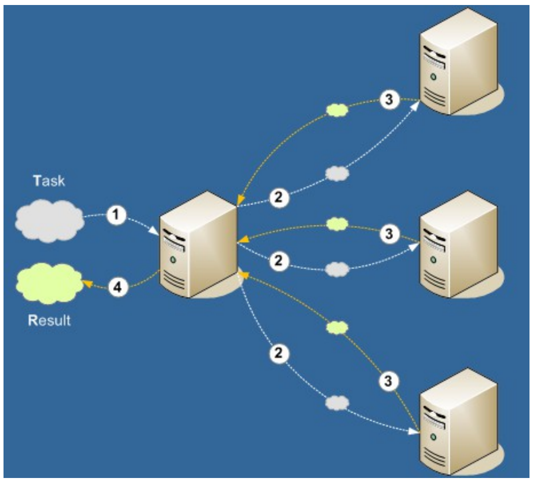
Brief Summary about Grid Computing

Basically there are two types of grids:

1. Computing Grids
2. Data Grids

Computing Grids – Provision of computing power

Computing Grids provide computing power. In the grid context the power is not provided by a central processing unit (like a single mainframe computer) but more by a computer network (grid) that can parallelize and consolidate the computing tasks and results.
Computing grids are often used to solve tasks with an extensive demand of calculation like weather forecasts, crash simulations, DNS calculation etc.



Computing grid requirements:

- The requests must be „tokenizable“ - split into pieces (jobs), that can be distributed in a network and computed on itself.
- The results must be „distributable“ and consolidated on other nodes in the network

Data Grids – Provision of Storage

Data grids provide storage and the access methods to handle storage requests. In the grid context the power is not provided by a central database or database cluster but more by a computer network (grid) that can parallelize and consolidate the access requests to data resources in different nodes on the network.
Samples for data grids: satellite pictures, email-sniffers, tracking and provisioning of telecommunication connections, etc.

Data grid requirements:

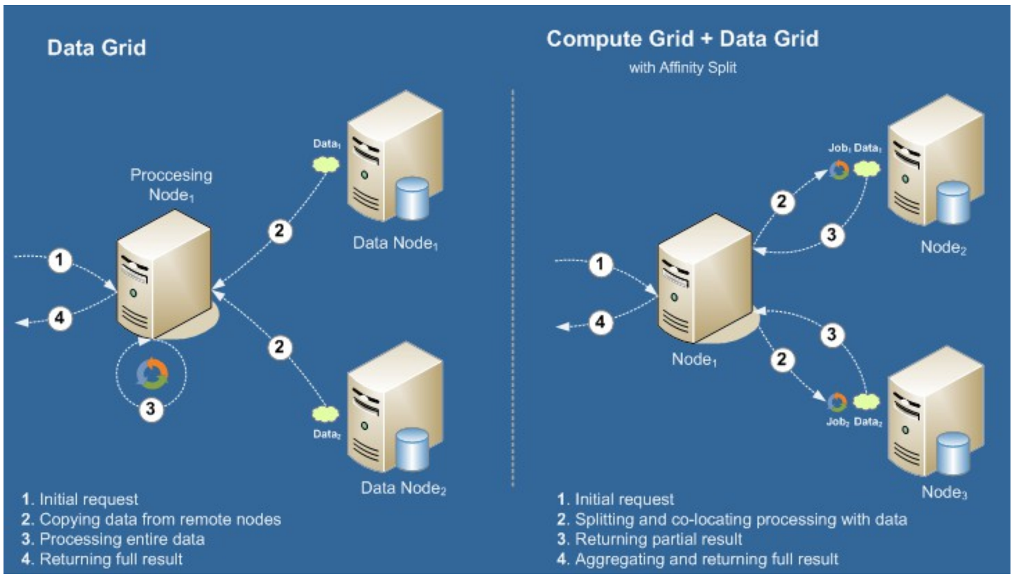
- The requests must be „tokenizable“ - split into pieces, that can be distributed in a network and must be self-contained.
- The results or data retrieval requests must be „distributable“ and consolidated on other nodes in the network. Therefore data grids are mostly not used in complex database environments where transaction safeness is a requirement (e.g. Banking).

Grid computing (Grid Services)

Operation of either computing grids, data grids or a combination of both.

Requirements:

- Networks with a high bandwidth



Grid computing – next step: Cloud Computing

Commercial Grid computing with flexible and automated handling of computing power and storage demands.

Requirements:

- Networks with a high bandwidth and
- a solid infrastructure support.

Business needs for Cloud Computing:

- Web Portals and Internet sites with billions of hits, seasonal peaks, rush hours etc. like Google and Amazon.
- Amazon, like ebay and yahoo, built their own Computing Clouds called EC2 (Elastic Cloud) and nowadays sell „Grid“ services to other companies.

*) Pictures are taken from GridGain Presentation

Grid Workshop Objectives

The purpose of this workshop is to familiarize you with the opportunities of Grid Computing concepts and technologies. The workshop brings you hands on to technologies evolving in the area of grid computing.

The workshop is based on the knowledge you achieved through the lectures on topics of Middleware Grid Computing.

While grid computing still seems to belong to scientific areas like research on complex calculations and mass data manipulations the acceptance of the grid technology is heavily increasing in parallel with its improvement of ease of use and the encapsulation of its complex architectures and infrastructure requirements.

Shifting the Paradigm

For this workshop we decided not to use GTK4 anymore (which was one of the original grid tool kits) and to overcome its shortage in adaptiveness for state of the art web application development projects.

Especially for Java Developers who are used to stable frameworks and many subsystems to make their job more focussing on the business logic more than on the framework itself there are well designed Grid Middleware frameworks on the market now leaving all the hassles of resource and infrastructure management behind the scene.

One of the most famous frameworks is **GridGain**.

GridGain since 2005 has become a well designed, stable and enterprise ready grid framework with a huge community and nowadays with commercial customers for their newly released Gridgain Enterprise Edition.

With Gridgain 3.0.x GridGain introduced state of the art functional APIs and also started to support Scala APIs and Groovy.

For this workshop we use newest version of GridGain to implement grid technologies and services.

GridGain - The Cloud Application Platform

Excerpt from GridGain Home Page (old version):

„GridGain is a computational grid product. In a nutshell, it allows you to parallelize the execution of the piece of code onto a set of computing resources. Computing resource can be a laptop, desktop computer, workstation, rack-server, mainframe or any other computing device with Java 5 or higher compatible JVM available.

Set of computing resources can be homogeneous (same kind) or heterogeneous (different kinds), can be generally located locally, within enterprise, globally or any combination of the above.

Most grid computing products fall into two categories: data grid and computational grids and usually are very different in how they are designed. Data grids essentially solve distributed data caching problem (storing large sets of data in distributed manner for high availability), while computational grids solve distributed computation problem (executing computation in parallel fashion).

GridGain is a state of the art computational grid product."

Real Time Big Data Processing

Excerpt from GridGain White Paper (new version):

GridGain is a JVM-based application middleware that enables companies to easily build highly scalable ***real-time compute and data intensive distributed applications*** *that work on any managed infrastructure - from a small local cluster, to private grid, to large private, public and hybrid clouds.*

To achieve this capability GridGain developed the only middleware in the world that integrates two fundamental technologies into one cohesive product:

- *Computational Grid*
- *In-Memory Data Grid*

These two technologies are axiomatic for any real-time distributed application as they provides the means for co-located parallelization of processing and data access - the cornerstone capability for enabling flat linear scalability under extreme high loads.

Goal of the Workshop

The intention of this workshop is to get practical experiences in solving real world problems with a grid toolkit – here GridGain -, to achieve understanding about how to “gridify” a solution.

The students will gain insights about the impact of grid architectures on program flow and the according infrastructures.

The students also learn about how to conduct the project management, to document the approach and the solution, to setup test cases and measurements. They have to answer questions about scalability, grid behaviors when scaled out, impact on infrastructure layouts.

At least the students will be able to decide whether and where grid technologies will be adoptable to the daily work of their own IT projects.

Workshop Assignment

Build a Java (web) application using the concepts and APIs of the GridGain framework.

The purpose of the application is to answer the following questions:

1. Can the adaption of computational grid concepts like GridGain improve performance of computational task in web applications
2. At which cost, program design, development, deployment, roll-out, skill requirements, infrastructure requirements
3. How scalable is the GridGain gridified application, on the same CPU, on more than one machine (real hardware or virtualized Hardware)
4. What is the saturation point of gridifying an application, can a thumb rule be constructed for general purposes?

Workshop Preparation

Because of the duration of 2 days for the Grid workshop it is essential for all participating students that they learn the fundamentals about the tools used in this workshop. To increase effectiveness for the workshop students should install all software packages required to start programming the grid solution on their own computers if available.

1. To find required software packages and dependencies students should read the GridGain book which is provided for online access
   http://www.gridgainsystems.com/wiki/display/GG15UG/Table+Of+Contents
2. Provide a functional Java development framework of your choice (e.g. eclipse); see GridGain documentation for preferred Java environments
   http://www.eclipse.org/downloads/
   http://eclipse.org/downloads/moreinfo/jre.php
3. Download the latest GridGain Software Version
   http://www.gridgain.com/download/
4. Download GridGain user guide and tutorials
   http://www.gridgain.com/book/book.html#_taste_of_gridgain
   http://www.gridgain.com/media/gridgain_java_refcard.pdf
   http://www.gridgain.com/media/imcg_explained.pdf
   http://forum.gridgain.com/
5. Try to build some of the tutorial applications to get the idea behind GridGain; if you are not a Java programmer yet ask someone from your course/team to support you
6. Learn about the GridGain APIs; learn how GridGain splits tasks, distributes and controls them and steels them back when profitable
7. If you chose the PDF proposal for your team:
   Familiarize with the PDF document structure to be able to build and analyse searchable PDF documents; there are some open source packages available to convert documents to PDF (e.g. OpenOffice.org, LibreOffice, LateX) and many more less known algorithms and software snippets
   Learn a little about bitmaps or .tif files to understand why these formats are used to compare the results

Team Roles

Project Leader

Coordinates the project activities,

develops with the team the breakdown structure for the solution,

defines the documentation structure

is responsible for the delivery of the assigment before closing date

Development Team

Setup the development environment, design the solution,

define the architecture, develop und test application

Test Manager

Defines and tests use cases, monitores performance,

develops reusable test cases; provides test data

Documentation Writer

Responsible for the project documentation, including documentation of the

different project phases: kickoff, planning, development, testing.

Team Structures

Team A

Acquadro, Davide (Testmanager)

Frey, Stephan (Developer)

Lehmann, Kristian (Developer)

Oßwald, Marco (Doc.-Writer)

Rauscher, Jennifer (Testmanager)

Van der Schoot, Roland (Projectleader)

Team B

Eisele, Raimond (Projectleader)

Fichtner, Viktor (Developer)

Keller, Daniel Philipp (Developer)

Merk, Sergej (Testmanager)

Sayler, Matthias (Doc.-Writer)

Zacharias, Philipp (Doc.-Writer)

Team C

Fernandez, Ana (Projectleader)

Gato Barrios, Maria (Doc.-Writer)

Hahn, Immanuel (Testmanager)

Riedel, Matthias (Developer)

Trumpp, Philipp (Developer)

Weiß, Samir (Testmanager)

Team D

Baur, Benjamin Felix (Doc.-Writer)

Ehlert, Jens (Testmanager)

Köse, Muhammed (Testmanager)

Maier, Frank (Doc.-Writer)

Rösner, Benjamin (Developer)

Wirsum, Fabian (Projectleader, Developer)

Workshop Exercise

The teams should chose from the proposals made in the GridGain InfoLetters or are free to define a problem on their own.

While the problem must not be too sophisticated it should leave room for implementation of many GridGain APIs as applicable – which is the focus of this workshop.

Tasks distributed to the grid will run and finish within different time frames, therefore jobs sent to grid machines have to be monitored, stolen back and redistributed. Discover which distribution strategy does the best for performance optimization; find out which data caching is feasible.

Results should be visualized in a simple web application or using GridGains dashboard.

Proposals given in InfoLetter #6

1. PDF Inspector: Read a bunch of text documents, convert them to PDF Files, count characters, words, chapters and pages; convert the PDF Files into images

2. Naked Skin Detektor: Read a bunch of images to detect if there are naked skin images or significant part of the images covered by naked skin; the images can therefore be segmented to produce more and smaller workload for the underlying Grid Nodes. At least decide whether a web blocker should prevent display of the images

3. Read text files to detect the language the text is written; use algorithmns freely available to find out which characters are specific to which language; provide 100 and more HTML documents to analyse

4. Brute force attack; try to decrypt passwords encrypted by keyword programs.

Workshop Timetable ***– Day 1***

1.	Introduction to Grid Programming *- Lecturer*	9.00 – 09:30
2.	Team Building Build teams with a maximum of up to 6 team members Assignment of team roles *- All Students*	9.30 – 10:00
3.	Problem & Architecture Presentation *- Project Leaders, Team Members -*	11.00 – 11.30
4.	Project work *- Each Team*	11.30 – 13.30
5.	PL-Meeting *- Project Leaders*	13.30 – 13.45
6.	Dev-Meeting *- Developer*	14.30 – 14.45
7.	Doc-Writer-Meeting *- Documentation Writer*	15.30 – 15.45
8.	Test-Meeting *- Testmanager*	16.00 – 16.15

Workshop Timetable – Day 2

1.	Project work *- Each Team*	8.00 – 10.00
2.	Project Presentations *- All Students*	10.00 – 11.00
3.	Project work *- Each Team*	11.00 – 14.00
4.	PL-Meeting *- Project Leaders -*	14.00 – 14.30
5.	Project work *- Each Team*	14.30 – 16.00
6.	Final Status *- All Students*	16.00 – 16.15
7.	*Project work* *- Each Team*	*16.15 – 18.00*

Project Results & Deliverables

- Solid description of the solution
- Architecture overview and detail views
- Code samples (and instructions how to make the code run)
- References to web sites and documents used for citation and solution building
- Outlook of the solution; what is missing to make it a commercial/usefull product? Focus on the requirements in a production environment
- Discussion of the obstacles, possible enhancements, alternative approaches, differences to non-grid solutions

Project Results and Packaging

The project results have to be bound to 1 major zip file:

Naming convention: MScGrid_assignment_WS5_team**[x]**.zip

The zip contains 2 more files as follows:

1. The assigment documentation as PDF document
   - Project Report (including code snippets and instructions usefull for describing the work, screen shots when applicable)
   - 1[2] Pages written by each team member describing his role, valuation of the grid computing and valuation of the project itself
   - SED02 – submission form and signature under the terms of originality: each team member has to fill out a form, all forms can be assembled to one page, the page then can be scanned and included in the documentAll scripts and writings
   - Use either MS-Word or OpenOffice.org (>3.x) and produce an Adobe PDF file as final output
   - Naming convention: MScGrid_assignment_WS5_team**[x]**.pdf
2. The code of the runnable application as .zip file
   - how_to_run.txt
   - [application subdirectories as described in how_to_run.txt
   - [application ressources as applicable]
   - Naming convention: MScGrid_assignment_WS5_team**[x]_app**.zip

Project Results & Deadline

- Final deadline for the deliverable**s**

  March 14nd, 2014 - 23:59 pm

- Send deliverables to michael@pyschny.de
- Subject: „MScGrid WS 2014 Spring Team x“
  (where **x** is your team number)
- cc: your team members
  cc: nina.goetz@tae.de

- Deliverables sent in beyond the deadline will be rejected
- Do not forget to attach the packages!

  Note: If the deliverables of the code package are too large to sent please provide a download link from where the package can be download. Leave at least 2 days for the download before the final deadline

Project Results & Valuation:

- The valuation will be per team, i.e. **all team members achieve the same evaluation**
- The deliverables should demonstrate understanding of concepts of grid computing - evaluates to 60%
- Code should be executable and will underline the basic understanding of grid computing - evaluates to 40%

Useful hints for your assignment (by former students)

- Integration of Tomcat and GridGain can be time consuming; if Tomcat and GridGain will not work together use simple Swing to visualize the results
- Learn how to use GridGain under eclipse in front of the workshop; it may take some time to get the samples run
- Try to make some GridGain samples run before the workshop to make sure that everything is in place for the development during the workshop
- Focus of the assignment is the finding about GridGain and its rich APIs and NOT the beautifying of the solution in form of a nice GUI or well designed WebPages
- In case that the solution lacks a functional GUI to initialize the scenarios to achieve the prospected results the how_to_run.txt must include sufficient description to easily run the code and produce the results in a form which makes them reproducible for the reader
- The assignment should have at least 20 pages+ and the additional personal conclusions + appendices (optional) + submission form for each team member (optional)
- The problem solution is meant as a red line for gaining insights about Grid Computing and GridGain adoption – there is no expectation to achieve a full functional solution
- Do not use path names and file names with blank or german special characters
- Use the same Java Environment for all nodes (SUN, Oracle, OpenJDK), do not mangle the JDKs

Some GridGain Implementation How-Tos

How to restrict work to your team

Use setMulticastGroup in GridMulticastDiscoverySpi:

```
GridMulticastDiscoverySpi spi = new GridMulticastDiscoverySpi();

// Put another multicast group.
spi.setMulticastGroup("228.10.10.157");
GridConfigurationAdapter cfg = new GridConfigurationAdapter();
// Override default discovery SPI.
cfg.setDiscoverySpi(spi);
// Start grid.
GridFactory.start(cfg);
```

How to balance work

GridJobStealingCollisionSpi supports job stealing from over-utilized nodes to under-utilized nodes. This SPI is especially useful if you have some jobs within task complete fast, and others sitting in the waiting queue on slower nodes. In such case, the waiting jobs will be stolen from slower nodes and moved to the fast under-utilized nodes

How to handle data in the computing grid

If you have to deal with large portions of data you will recognize that shifting data to nodes can be very time consuming – unless you use optimized transport protocols etc.

Therefore you should install a share if possible and make it available to all your team members and then send links to the data instead of sending the data itself.

Assignment Template

Page 1 content:

Brunel University West London
School of Engineering & Design Electronic & Computer Engineering
M.Sc. in Distributed Computing Systems Engineering
Workshop 5
Grid Programming with GridGain
Team x
Team Members 1., 2., ….

Tutor: M. Pyschny
Date

Contents

1. Introduction
   1. Workshop aims & objectives
2. Background
   1. Overview of Grid Computing
   2. GridGain framework
3. Projekt Description
   1. Problem Definition
   2. Requirements
4. Project Management
   1. Project Team
   2. Project Roles
   3. Project Plan
   4. Project Activities
   5. Project Milestones
5. Design & Implementation
   1. System requirements
   2. System architecture
      1. Client package
      2. Server package
      3. GUI implementation
      4. Applied algorithms
   3. Measurements
6. Test Management
   1. Test Environments
   2. Test Cases
   3. Test Scenarios
      1. Setting up the test environments
      2. Using grid nodes
      3. Load balancing
   4. Test Conduction
7. Analysis
   1. Assessing the results
   2. Project final report
8. Conclusion
   1. Obstacles
   2. Outlook
9. Appendix
   1. Personal statements of the team members
   2. Assigment Submission Forms

Table of References
Table of Figures