Scheduling of Grid Workflows

Seminar "Grid Computing 2"
Institute of Computer Science
University of Innsbruck

by Sarah Löw and Markus Brejla
Structure Of Today's Presentation

- Grid Workflow – Design
- Grid Workflow – Scheduling
- Examples for recent scheduling projects
- GWFE – Grid Workflow Engine
Grid Workflow – Design

- How can we specify and combine grid services?
Grid Workflow – Design

- Workflow Structure
  - DAG
    - Sequence
    - Parallelism
    - Choice
  - Non-DAG
    - DAG-entities
      - Iteration
  - Abstract
    - Concrete
    - Time
    - Cost
    - Reliability
    - Security

- Workflow Design
  - Workflow Model/Specification
  - QoS Constraints
Grid Workflow – Design (Structure)

- **Workflow**: Concatenation and combination of multiple tasks, according to their dependencies

- **Workflow structure**: Indicates temporal relationship of all tasks in the workflow.

**Types of Structure**
- Direct Acyclic Graph (DAG)
  - Sequence, Parallelism, Choice
- non-DAG
  - All structures of DAG + Iteration
Grid Workflow – Design (Model)

- Defines anything containing tasks and structures in the workflow.

- Types of model definition
  - Abstract Model
    (no low-level implementation details)
  - Concrete Model
Grid Workflow – Design (QoS Constraints)

- Users expect realisation of QoS constraints
- Workflow management systems have to consider QoS constraints
- *Exemplary constraints:*  
  time, cost, reliability, security
- *Two possibilities of assigning QoS Constraints:*  
  - Task-level  
  - Workflow-level
Grid Workflow – Design (QoS Constraints)

Example for Task-level Assignment:

```xml
<Workflow>
  <tasks>
    <task name="A">
      <qos-constraints>
        <qos-constraint name="time" value=" "/>
        <qos-constraint name="cost" value=" "/>
      </qos-constraints>
      ...
    </task>
    ...
  </tasks>
  ...
</Workflow>
```
Example for Workflow-level Assignment:

```xml
<Workflow>
  <qos-constraints>
    <qos-constraint name="time" value="2006-29-11"/>
    <qos-constraint name="cost" optimal="on"/>
  </qos-constraints>
  <tasks>
    ...
  </tasks>
  ...
</Workflow>
```
Grid Workflow – Scheduling

- "Workflow scheduling focuses on mapping and managing the execution workflow tasks on shared resources that are not directly under the control of workflow systems."
  (Jia Yu, R.B.: A taxonomy of scientific workflow systems for grid computing. Technical report, GRIDS Laboratory, The University Of Melbourne, Australia (2005))

- **Goal of scheduler**: find optimal mapping of tasks on machines (NP-complete problem)
  - Scheduler approximates solution
Grid Workflow – Scheduling

- Architecture
  - Centralised
  - Hierarchical
  - Decentralised

- Decision Making
  - Local
  - Global
  - Static
  - Dynamic
  - User-directed
  - Simulation-based
  - Prediction-Based
  - Just in-time

- Planning Scheme

- Workflow Scheduling

- Scheduling Strategies
  - Performance-Driven
    - Market-Driven
    - Trust-Driven

- Performance Estimation
  - Simulation
  - Analytical
  - Modeling
  - Historical Data
  - On-line Learning
  - Hybrid
Grid Workflow – Scheduling (Architecture)

- Centralised
  - Only one scheduler
  - Needs information of entire workflow

- Hierarchical
  - One central manager and multiple sub-workflow schedulers
  - Schedulers can work with different scheduling policies
  - Hazard: central manager

- Decentralised
  - Multiple flat schedulers
  - Same rights
  - Can communicate
Grid Workflow – Scheduling
(Decision Making)

- Difficult to find best solution for mapping

- Two possibilities:
  - Local
    - Decision based on current task
  - Global
    - Decision based on entire workflow

- Keep balance between imperfect quick scheduling (local) and very good time-exhaustive scheduling (global)
Grid Workflow – Scheduling (Planning Scheme)

- How to transform abstract into concrete workflow model
- Two methods for planning scheme:
  - Static (full-ahead planning)
    - Generate concrete model before starting workflow execution with static information of execution environment
    - User-directed Planning
    - Simulation-based Planning
  - Dynamic
    - Uses static information and information about changing states of resources
    - Prediction-based Planning
      - Based on dynamic information and prediction
    - Just in-time Planning
      - Make decision only at run-time
      - Stable in respect to changing working environment
Grid Workflow – Scheduling (Scheduling Strategies)

- **Scheduling** = finding approximation of NP-complete problem that satisfies user interest best

- **Different user interests:**
  - Performance-driven
    - Minimise overall execution time
  - Market-Driven
    - Minimise cost
  - Trust-Driven
    - Trust level of resources is known
    - Only map to machine if trust level is higher than user’s trust level
    - Trust level contain a.o. security policy and attack history
    - This scheduling can help increasing reliability of workflow execution
Grid Workflow – Scheduling (Performance Estimation)

- Estimation on how long it will take to execute a task of given type

- Schedulers have to predict the behavior of tasks or sub-workflows

- *Four prediction types:*
  - Simulation
  - Historical Data
  - Analytical Modeling
  - On-line Learning
  - Hybrid
Examples For Recent Scheduling Projects

- **GridFlow**
  - *developed by* University of Warwick, Coventry, UK, (Junwei Cao)

- **GrADS**
  - *developed by* collaboration between several American Universities (Rice University, University of Tennessee, University of Chicago, . . . )

- **The Gridbus Project**
  - *developed by* University of Melbourne (Rajkumar Buyya)

- **Askalon**
### Examples For Recent Scheduling Projects

<table>
<thead>
<tr>
<th>Workflow Design</th>
<th>GridFlow</th>
<th>GrADS</th>
<th>Gridbus</th>
<th>Askalon</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Structure</strong></td>
<td>DAG</td>
<td>DAG</td>
<td>DAG</td>
<td>Non-DAG</td>
</tr>
<tr>
<td><strong>Model</strong></td>
<td>Abstract</td>
<td>Abstract</td>
<td>Abstract/Concrete</td>
<td>Abstract</td>
</tr>
</tbody>
</table>
Examples For Recent Scheduling Projects

<table>
<thead>
<tr>
<th>Workflow Scheduling</th>
<th>GridFlow</th>
<th>GrADS</th>
<th>Gridbus</th>
<th>Askalon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture</td>
<td>Hierarchical</td>
<td>Centralised</td>
<td>Hierarchical</td>
<td>De-centralised</td>
</tr>
<tr>
<td>Decision Making</td>
<td>Local</td>
<td>Local/Global</td>
<td>Local</td>
<td>Global</td>
</tr>
<tr>
<td>Planning Scheme</td>
<td>Simulation-based</td>
<td>Pred.-based</td>
<td>User-directed, just in-time</td>
<td>Just in-time/Pred.-based</td>
</tr>
<tr>
<td>Strategies</td>
<td>Performance-driven</td>
<td>Performance-driven</td>
<td>Market-driven</td>
<td>Performance and Market driven</td>
</tr>
<tr>
<td>Performance Estimation</td>
<td>Analytical modeling</td>
<td>Historical Data, Analytical modeling</td>
<td>Historical Data</td>
<td>Analytical modeling, Historical Data</td>
</tr>
</tbody>
</table>
Gridbus Workflow Engine

- Introduction
- Architecture
- Workflow Language (xWFL)
- Scheduling System
- Summary
Gridbus Workflow Engine

Introduction

- developed by the Gridbus Project
- University of Melbourne, Australia
- Mission:
  Taking Grid computing to the Nth Level: Define, Differentiate, and Disrupt!
The Gridbus Middleware
Gridbus Workflow Engine
Features

- Simple linking of stand-alone applications
- XML-based workflow language
- Scheduling system:
  - decentralized architecture
  - just-in-time scheduling
  - event-driven communication
- GPL license
Gridbus Workflow Engine Overview

Workflow Enactment Engine

Globus Toolkit (Middleware Support)

- Security
- Data Management
- Execution Management
- Information Services
- Common Runtime
- IBM TSpaces
- MySQL
- Gridbus Market Directory
- Gridbus Broker
Gridbus Workflow Engine Architecture

Workflow Description & QoS
- Workflow Planner
- Application Composition
- Scientific Portal

Workflow Enactment Engine
- Workflow Submission Handler
- Workflow Language Parser
  - Tasks
  - Parameters
  - Dependencies
    - Workflow Scheduler
      - Dispatcher
      - Resource Discovery
      - Data Movement
        - Gridbus Broker
        - Globus
        - Web Services
        - HTTP
        - GridFTP

Info Services
- GMD
- Replica Catalog
- MDS
Gridbus Workflow Engine Architecture

- **Workflow Submission Handler**: Accepts workflow enactment requests from the planner level applications.
- **Workflow Language Parser**: Converts XML workflow description into Java objects (Task, Parameter, DataConstraint).
- **Resource Discovery**: Queries Grid information services such as Globus MDS, GMD and replica catalogs, to locate suitable resources for the tasks.
- **Dispatcher**: A dispatcher for every different middleware allows interaction with several middlewares.
- **Data Movement**: Enables data transfer between Grid nodes using HTTP or GridFTP protocols.
- **Workflow Scheduler**: Central component interacting with resource discovery to find suitable Grid resources at run-time.
Workflow Language Overview

- xWFL2.0
- XML-based modeling language
- describe tasks and their dependencies
- supports parameterization
Workflow Language Overview

- **Parameters**
  - single, range, select, random, file, multi-files

- **Tasks**
  - set of tasks in the workflow
  - inputs/outputs are specified (file, parameter value or data stream)
  - concrete vs. abstract workflow

- **Data Links**
  - specifies data flow between tasks
Workflow Language Schema
Workflow Language Parameter Example

<parameters>
  <para type="single">
    <name>X</name>
    <value type=integer>10</value>
  </para>
  <para type="range">
    <name>Y</name>
    <min>1</min>
    <max>20</max>
    <step>2</step>
  </para>
</parameters>
<tasks>
  <task name="A">
    <executable>
      <name>Preprocessing</name>
      <host>altix1.uibk.ac.at</host>
      <accesspoint>/home/cb56/cb561081/preprocessing.sh</accesspoint>
      <input>
        <port number="0" type="file" url="http://homepage.uibk.ac.at/~csad3277/files/input.rar">input.rar</port>
        <port number="1" type="msg">1</port>
      </input>
      <output>
        <port number="2" type="file">data.out</port>
      </output>
    </executable>
  </task>
  ...
</tasks>
<workflow>
  ...
  
  <links>
    <link>
      <from task="A" port="2" />
      <to task="B" port="0" />
    </link>
    <link>
      <from task="A" port="2" />
      <to task="C" port="0" />
    </link>
    <link>
      <from task="B" port="1" />
      <to task="D" port="0" />
    </link>
    <link>
      <from task="C" port="2" />
      <to task="D" port="1" />
    </link>
  </links>
</workflow>
Workflow Language
Fazit

- Supported workflow patterns:
  - sequence
  - parallelism
- Not supported:
  - iteration
  - choice
Gridbus Workflow Engine Scheduling System

- **decentralized** architecture
  - every task has its own scheduler

- **just-in-time** scheduling
  - resource allocation at run-time

- **event-driven** communication
  - subscription-notification model
Scheduling System Architecture

- Workflow Coordinator (WCO)
  - creates TM for each task
  - control lifetimes

- Task Manager (TM)
  - maintain set of resources
  - monitor remote jobs

- Event Service Server (ESS)
  - platform for events
  - notifies TMs, WCO when event occurs
Scheduling System
Communication Scheme

- IBM TSpaces implementation
  - middleware to link computing devices using tuple spaces

- ESS event tuples
  - task status event
  - output event
  - job status event
Scheduling System Communication Scheme
Scheduling System
TM state machine
Scheduling System
Sequence diagram
Scheduling System
Sequence diagram

activate the children task managers

notify TMI Status

send status done

send status executing

submit job on a remote resource

notify

send status executing

notify

submit job on a remote resource

send status done

send status done
Scheduling System

Fazit

- good design and ideas
  - flexible scheduling system
  - use of a lot of existing components
- strongly under development
- list of improvements
  - support economy-based workflow scheduling
  - support advanced control flow (cycles, cond.)
  - support parameterization
Summary

- Grid Workflow Scheduling is complex
  - mapping is NP-hard
  - shared resources
  - no central ownership

- Many different approaches
  - centralized vs. decentralized
  - performance-, market-, trust-driven

- Further research needs to be done!
Scheduling of Grid Workflows

Seminar "Grid Computing 2"
Institue of Computer Science
University of Innsbruck

by Sarah Löw and Markus Brejla