



ASKALON

A Workflow-Application Development and Execution Environment for the Grid



Kassian Plankensteiner
Simon Ostermann
University of Innsbruck, Austria

Outline

- ✦ Overview
- ✦ Architecture & Components of ASKALON
 - ✦ AGWL & Frontend
 - ✦ GridARM & Glare
 - ✦ Execution Engine
 - ✦ Monitoring
- ✦ Example Workflows
 - ✦ Povray and Wien2k

Distributed and Parallel Systems Group



- Mission:

- Research that simplifies the effective use of distributed and parallel systems to support scientific application development

- Target Systems:

- Parallel systems

- Cluster architectures
- Massively parallel systems
- Multi-core architectures

- Distributed systems

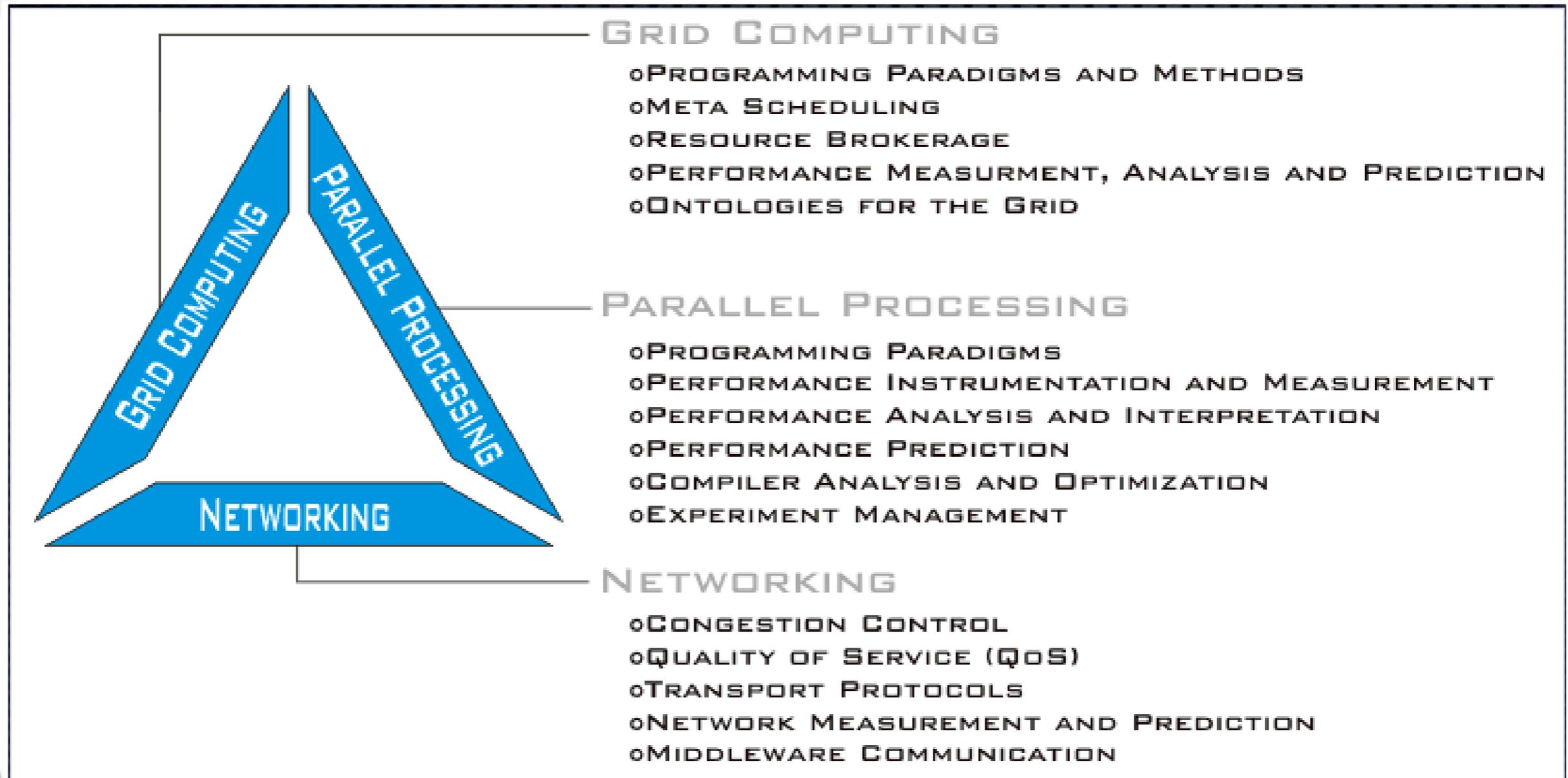
- Highly distributed, heterogeneous, dynamic, non-dedicated systems, failure-prone services and resources

- Target Users:

- Scientists, SMEs, industry, public institutions, and society



DPS: Research Directions



Projects:



Austrian Grid



Parallel computer	#	CPU	Clock	Architecture	Location
altix1.jku	64	ITA2	1,6	ccNUMA	Linz
hydra.gup	16	Athlon	1,6	COW	Linz
lilli.jku	256	ITA2	1,6	ccNUMA	Linz
schafberg.sbg	16	ITA2	1,6	ccNUMA	Salzburg
grid.fhv.at	21	Xeon	3	COW	Vorarlberg
gescher.vcpc	32	Xeon	3	COW	Vienna
karwendel.dps	72	Opteron	2,2	COW	Innsbruck
altix1.uibk	16	ITA2	1,6	ccNUMA	Innsbruck
hc-ma.uibk	16	Opteron	2,2	COW	Innsbruck
zid-grid	272	P4	1,8	NOW	Innsbruck

Components of ASKALON

Abstract Grid Workflow Language (AGWL)

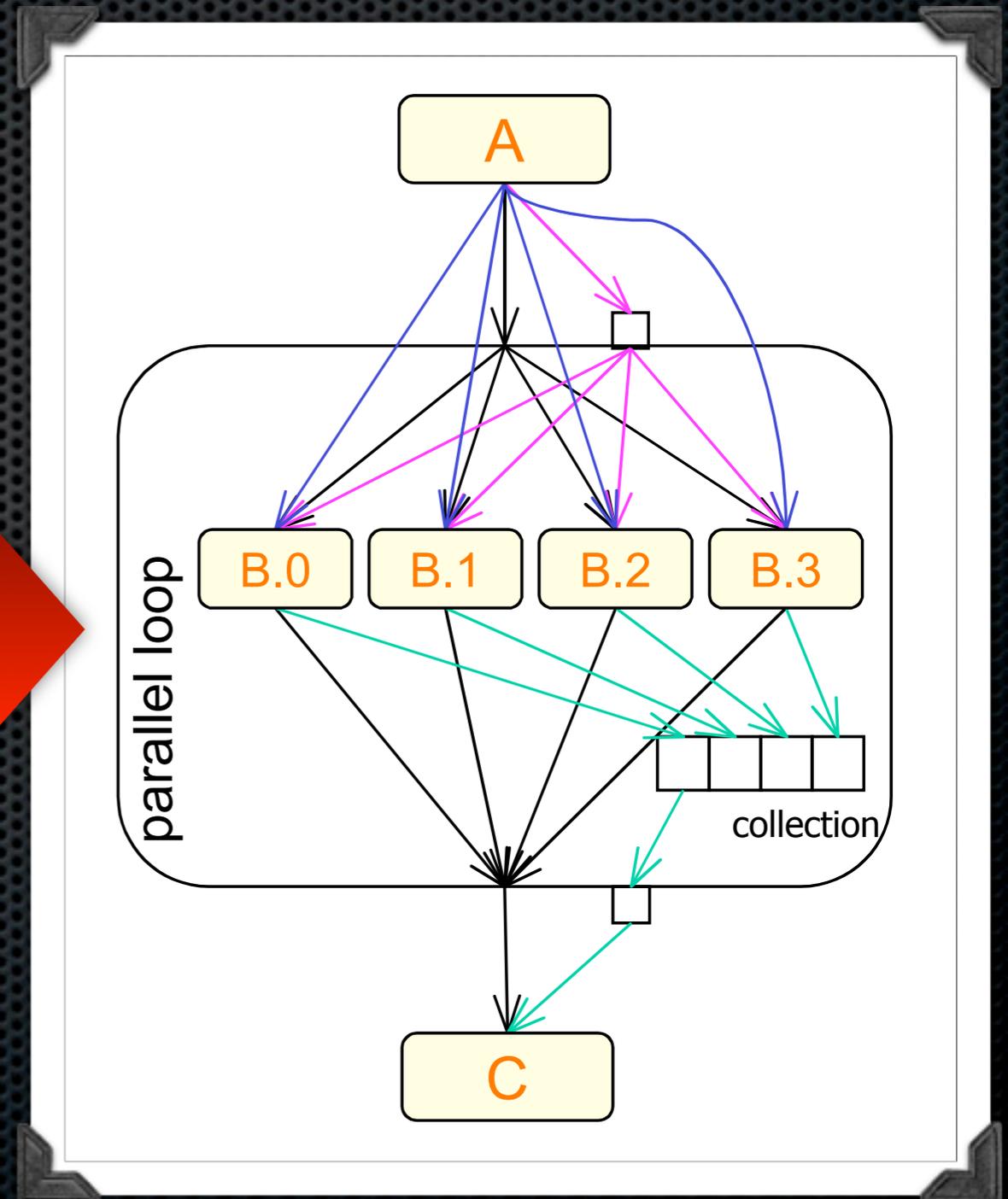
- ✦ Atomic activities
 - ✦ abstract from the real implementation, e.g. Web services, legacy applications
- ✦ Basic compound activities
 - ✦ Sequential constructs: <sequence>
 - ✦ Conditional constructs: <if>, <switch>
 - ✦ Loop constructs: <while>, <dowhile>, <for>, <forEach>
 - ✦ Directed Acyclic Graph constructs: <dag>

Abstract Grid Workflow Language (AGWL)

- ✦ Advanced compound activities
 - ✦ Parallel section constructs: `<parallel>`
 - ✦ Parallel loop constructs: `<parallelFor>`, `<parallelForEach>`
- ✦ Data flow constructs
 - ✦ `dataIn/dataOut` ports, collections, data repositories, data set distributions, etc.
- ✦ Properties
 - ✦ provide hints about the behavior of activities
- ✦ Constraints
 - ✦ Optimization metrics, Scheduling constraints

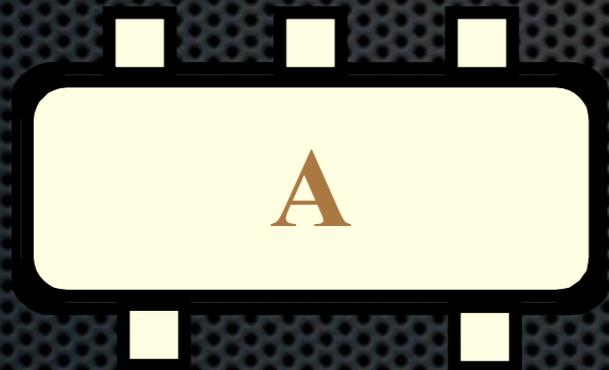
<parallelFor> Comp. Activity

```
<activity name="A" ... />  
<parallelFor name="pfor">  
  <dataIns>  
    <dataIn name="in"  
      source="A/out" />  
  </dataIns>  
  <loopCounter from="0" to="3"  
    step="1" />  
  <loopBody>  
    <activity name="B">  
      <dataIns ... />  
      <dataOuts ... />  
    </activity>  
  </loopBody>  
  <dataOuts ... />  
</parallelFor>  
<activity name="C" ... />
```



Activity Types

- ✦ Logical representation of a group of activity deployments (deployed in the Grid), which
 - ✦ Realize the same functionality
 - ✦ Have the same input/output data structure
 - ✦ implementation-independent



UML representation

```
<activity name="A" type="type Of A">  
  <dataIns>  
    <dataIn name="a" .../>  
    <dataIn name="b" .../>  
    <dataIn name="c" .../>  
  </dataIns>  
  <dataOuts>  
    <dataOut name="d" .../>  
    <dataOut name="e" .../>  
  </dataOuts>  
</activity>
```

AGWL representation

Registration of Activity Types

GLARE: activity type composition

Name: POVrayRenderer

Base Type: Povray

Domain: Imaging

Container: GT2

Build File URL: http://askalon.org/glare/povray.build

Total Deployments: 0

Auto:

Input:

GLARE: Type argument dialog

Edit the name/type fields and press Ok to update, Cancel otherwise.

Name: TotalFrames

Type: Integer

Ok Cancel

Output Arguments

Image.PNG File

Add Remove Update Link

Click to save type entry in GLARE Cancel to undo

Save Cancel

Describe an Activity

- Name
- Base Type
- Hosting Environment

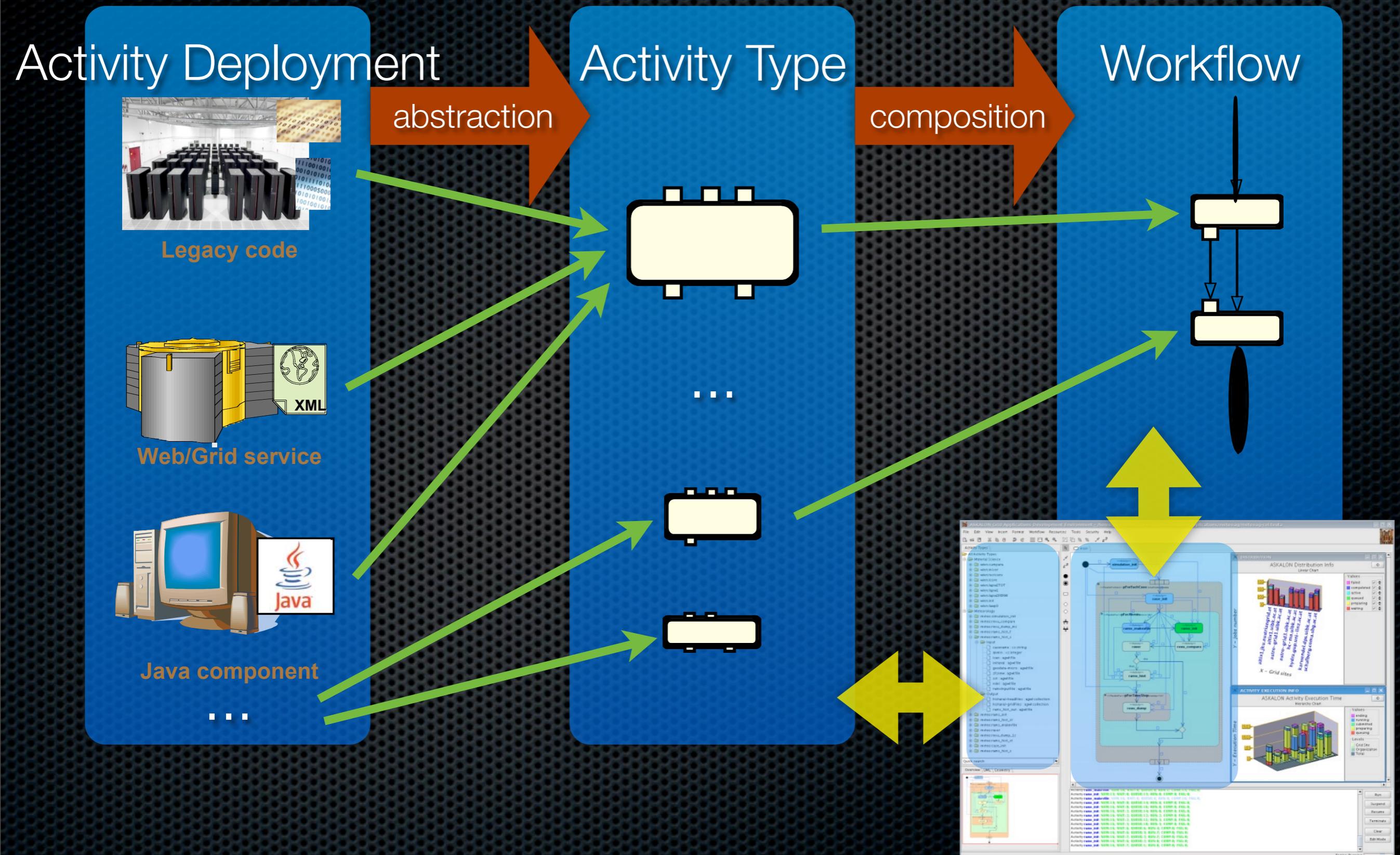
Define Installation steps in a Build File

Define & add arguments

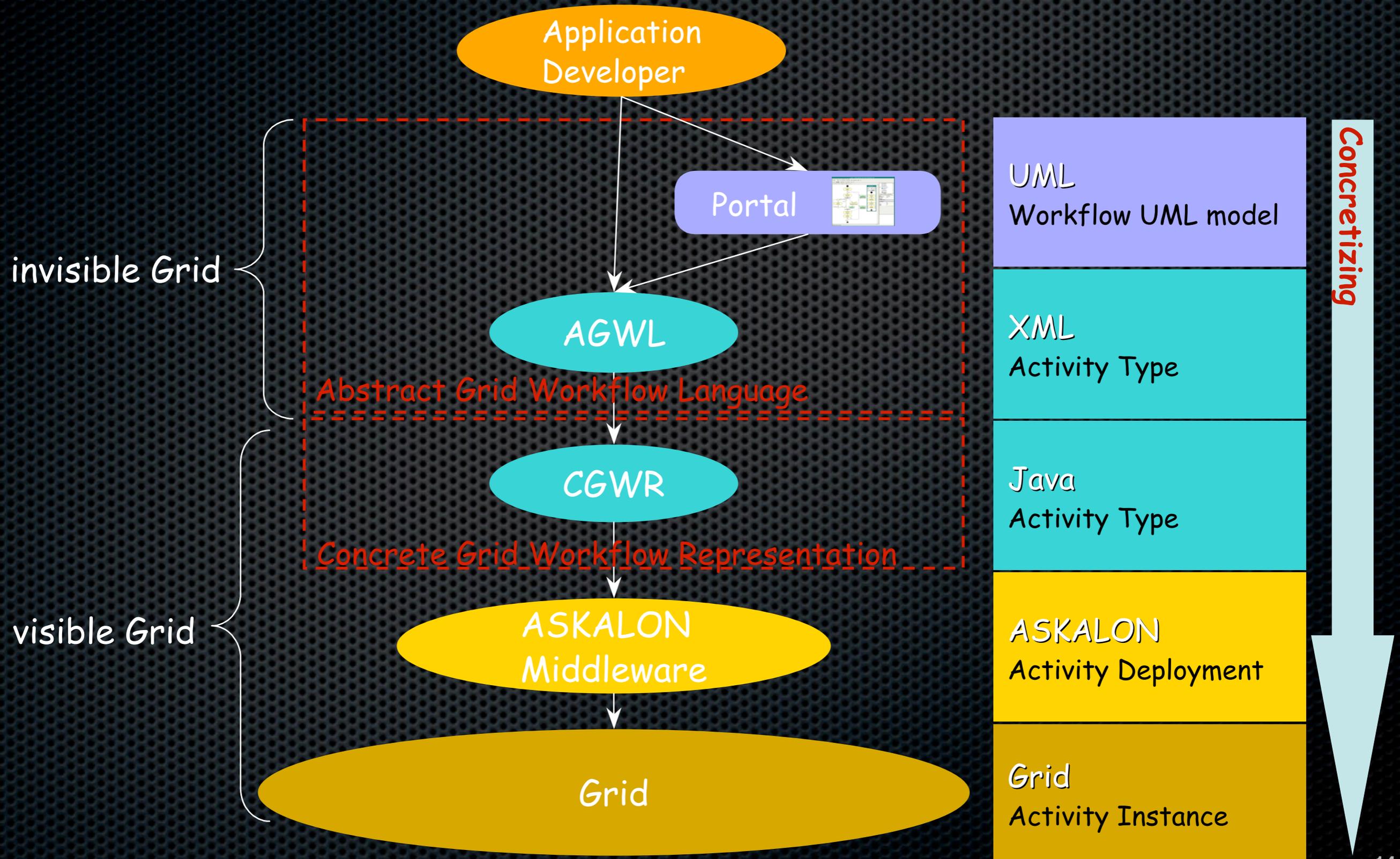
Activity Deployment

- ✦ Binds an activity type to a concrete installed implementation
- ✦ Description how to instantiate the activity
- ✦ Registered by the application provider in a special registry of the Resource Management service

ASKALON Workflow Composition

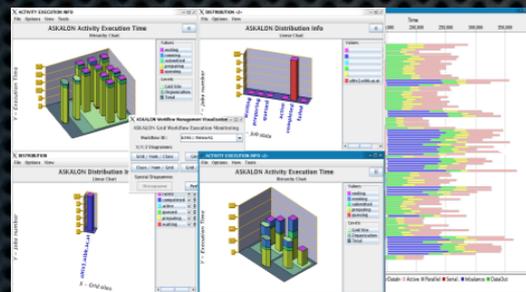
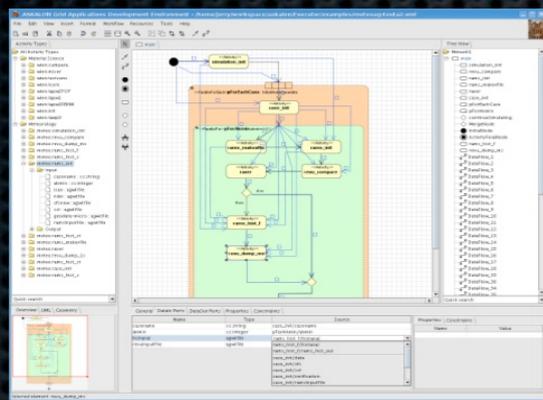


The Development Stack



Workflow Composition and Runtime Environment

UML-based Workflow Composition

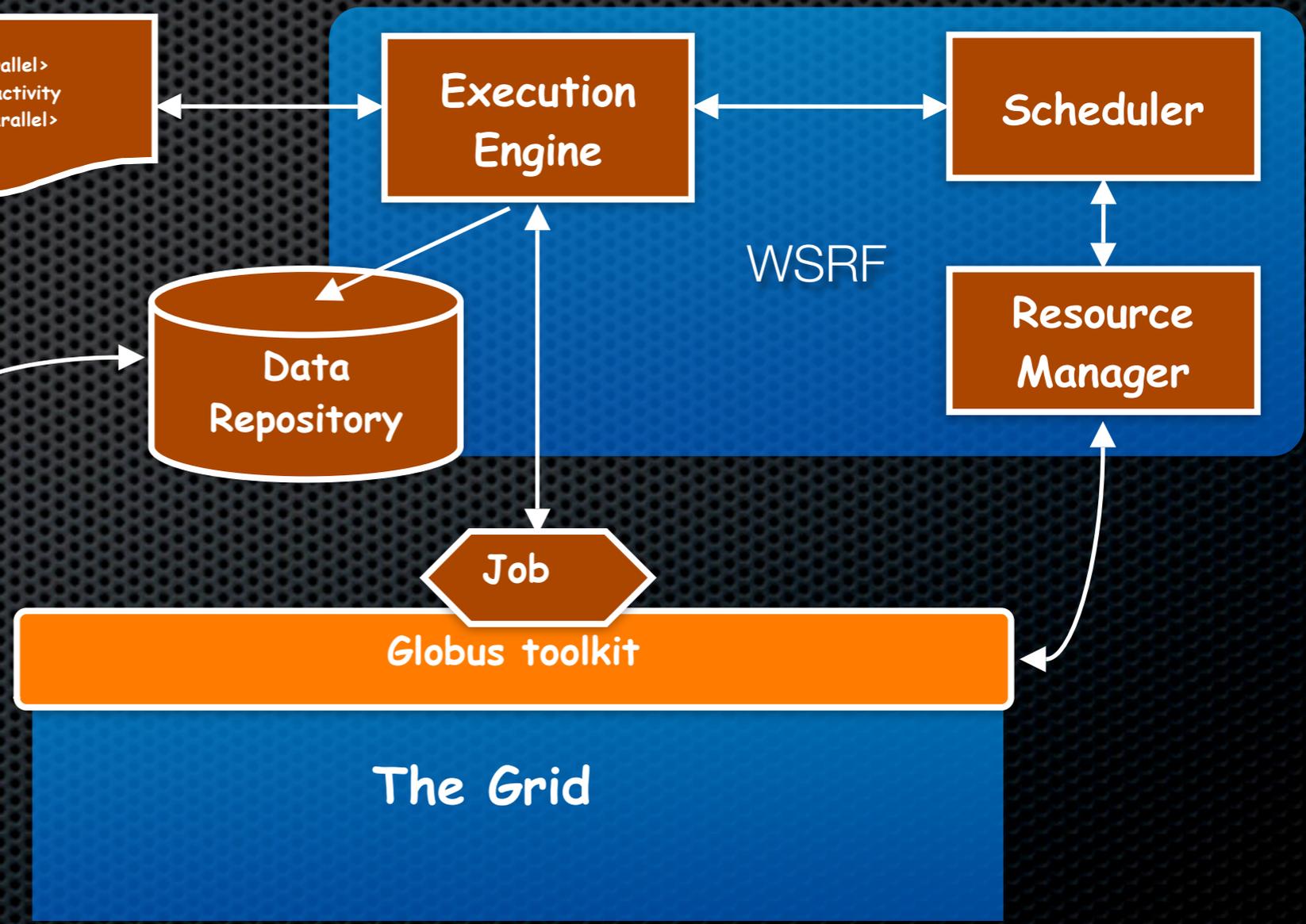


Performance Analysis

AGWL

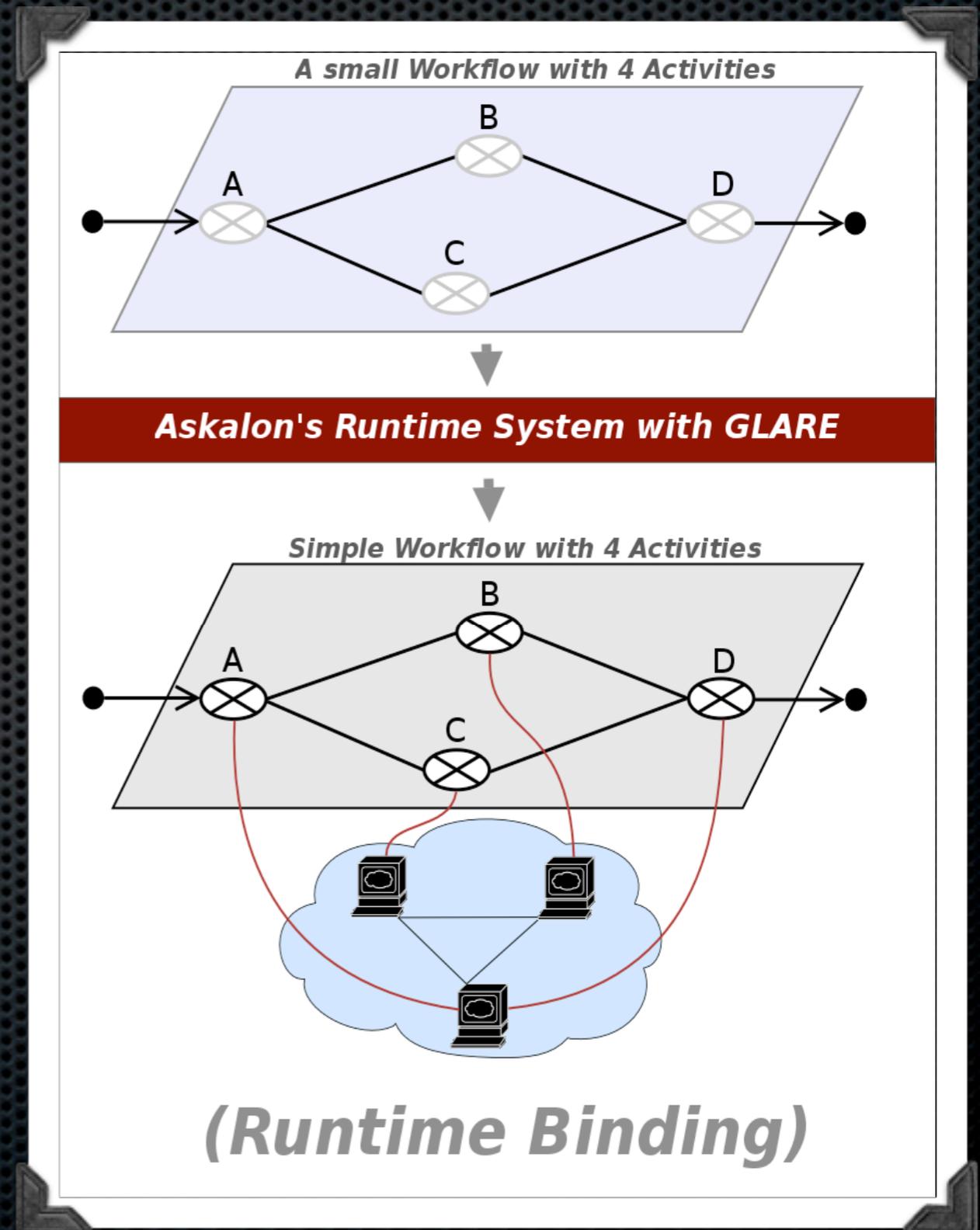


Runtime Middleware Services



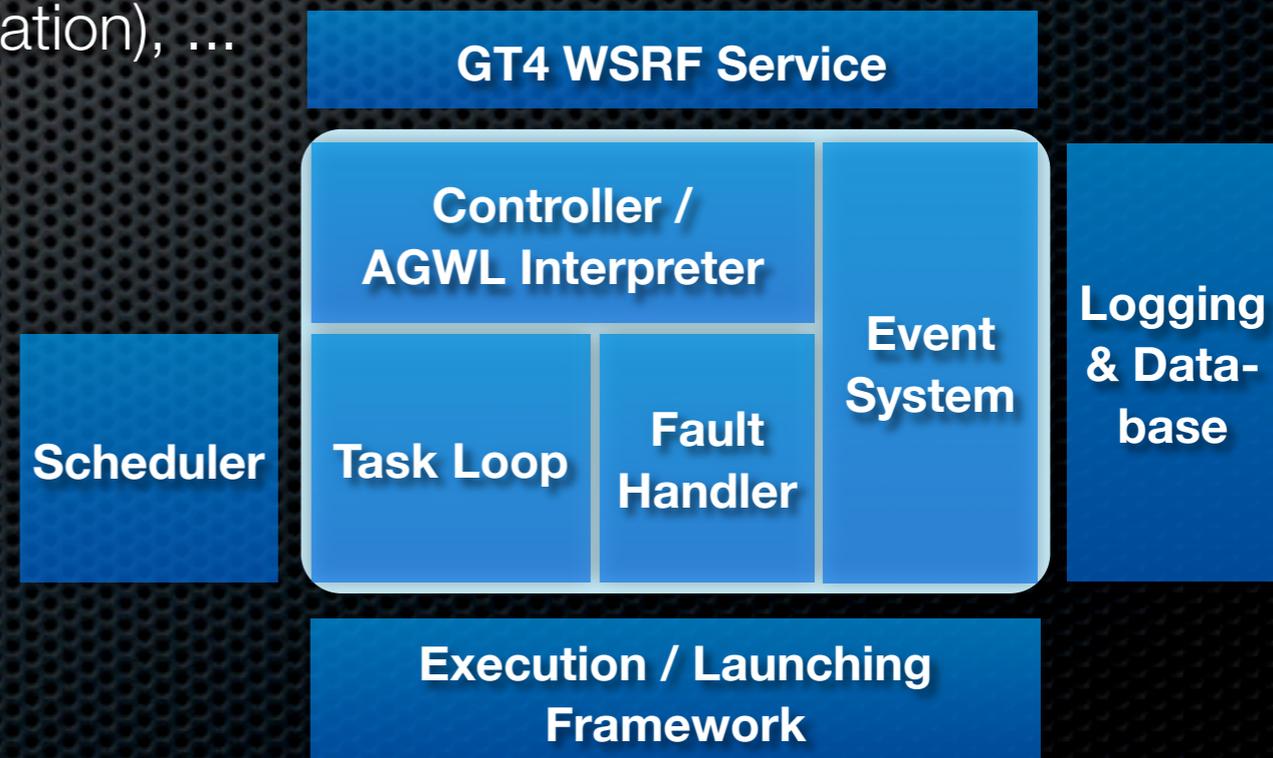
Concretizing Activities at Runtime

- Look up Activity Deployments for given Activity Type
- Choose suitable resources
- Concretize activities by mapping them to concrete deployments



Execution Engine

- ✦ Workflow controller
 - ✦ Converts XML-based specification (AGWL) to internal representation
 - ✦ Executes the workflow according to control and data flow dependencies
 - ✦ One separate Controller for every workflow instance
- ✦ Event system
 - ✦ Other components can subscribe to the internal events
 - ✦ e.g. logging, controller, tool (WS-Notification), ...
- ✦ Logging and database
 - ✦ For post-mortem performance analysis
- ✦ GT4 WSRF wrapper
 - ✦ Send WS-Notifications to the frontend for real-time monitoring



Resource-management

The screenshot shows a web-based interface for managing grid resources. On the left is a tree view of resources, divided into 'Sites' and 'Activities'. The 'Sites' section lists various locations like 'altix1.jku.austriangrid.at' and 'astro-grid1.uibk.ac.at'. The 'Activities' section lists various services like 'middleware', 'blender1', and 'Povray'. On the right is a control panel with tabs for 'Description', 'Properties', and 'Configure'. The 'Configure' tab is active, showing options to edit or remove activity types, update, delete, or add deployments, and registers/unregisters deployments for specific sites. Hand-drawn annotations include a black cloud pointing to the 'Sites' list labeled 'Browse Grid nodes', another black cloud pointing to the 'Activities' list labeled 'Browse Grid activities', and a red cloud pointing to the 'Deploy*' and 'Undeploy*' buttons labeled 'Deploy/undeploy'.

Grid Resources

- Sites
 - altix1.jku.austriangrid.at
 - invm:FindBest_altix1.jku.austriangrid.at_4
 - invm:WasimA_altix1.jku.austriangrid.at_40
 - invm:WasimB1_altix1.jku.austriangrid.at_4
 - invm:WasimB2C_altix1.jku.austriangrid.at
 - invm:WasimD_altix1.jku.austriangrid.at_40
 - tutorial:helloWorld_altix1.jku.austriangrid.
 - altix1.uibk.ac.at
 - astro-grid1.uibk.ac.at
 - grid.uibk.ac.at
 - hc-ma.uibk.ac.at
 - hephygr.oeaw.ac.at
 - hydra.gup.uni-linz.ac.at
 - karwendel.dps.uibk.ac.at
 - schafberg.coma.sbg.ac.at
- Activities
 - middleware
 - askalon.service
 - blender1
 - echoDate
 - grasil
 - helloWorld
 - invm:mod
 - lqcd
 - Povray
 - wien2k
 - wien:first
 - wien:first_karwendel.dps.uibk.ac.at
 - wien:fourth
 - wien:fourth_karwendel.dps.uibk.ac.at

Description Properties Configure

Edit/Remove activity type: Povray

Update Delete Add Deployment

Select site to register and unregister deployments

altix1.jku.austriangrid.at Register Unregister

Select site to register and unregister deployments

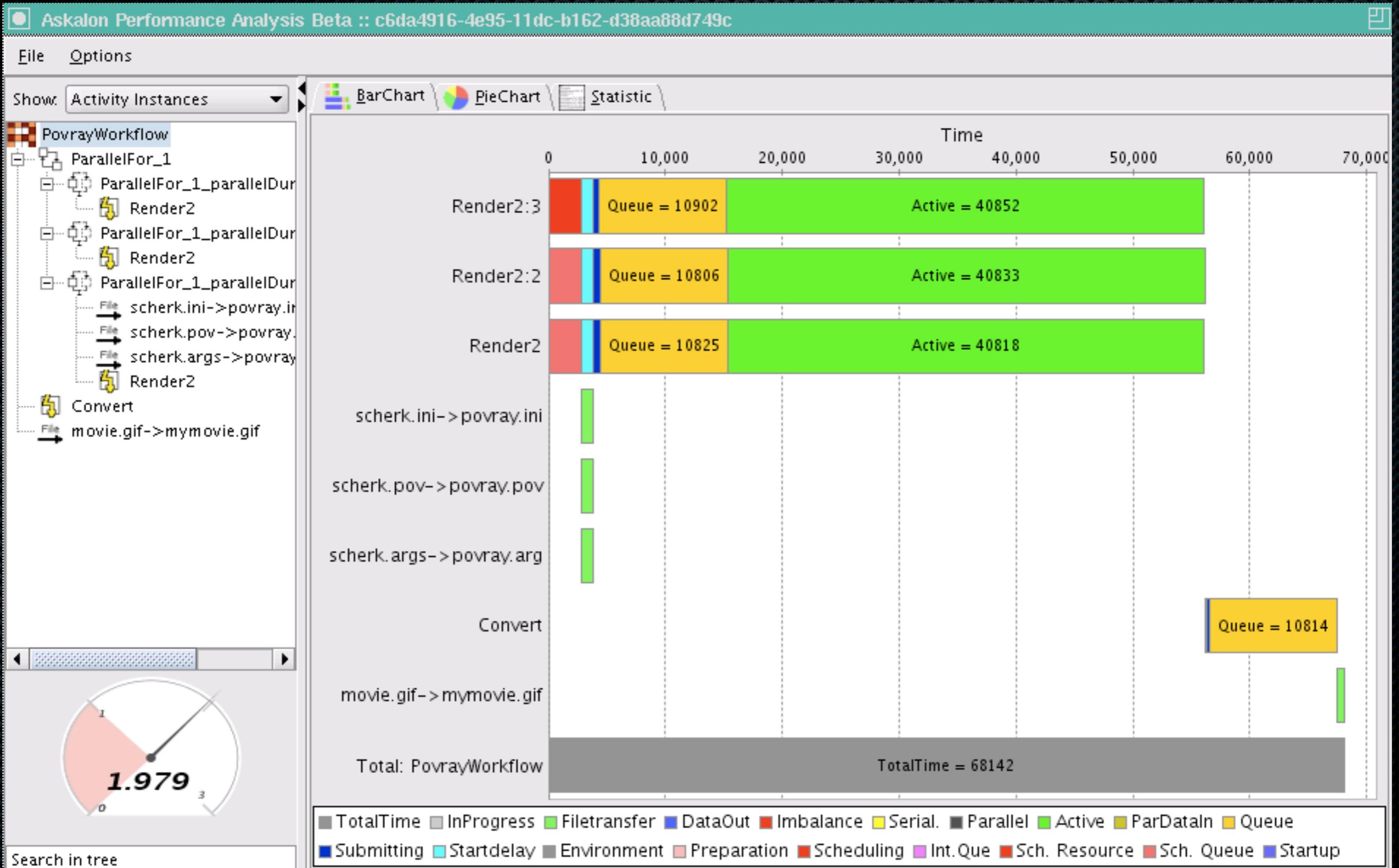
karwendel.dps.uibk.ac.at Deploy* Undeploy*

Browse Grid nodes

Browse Grid activities

Deploy/undeploy

Monitoring tool



Overhead Analysis

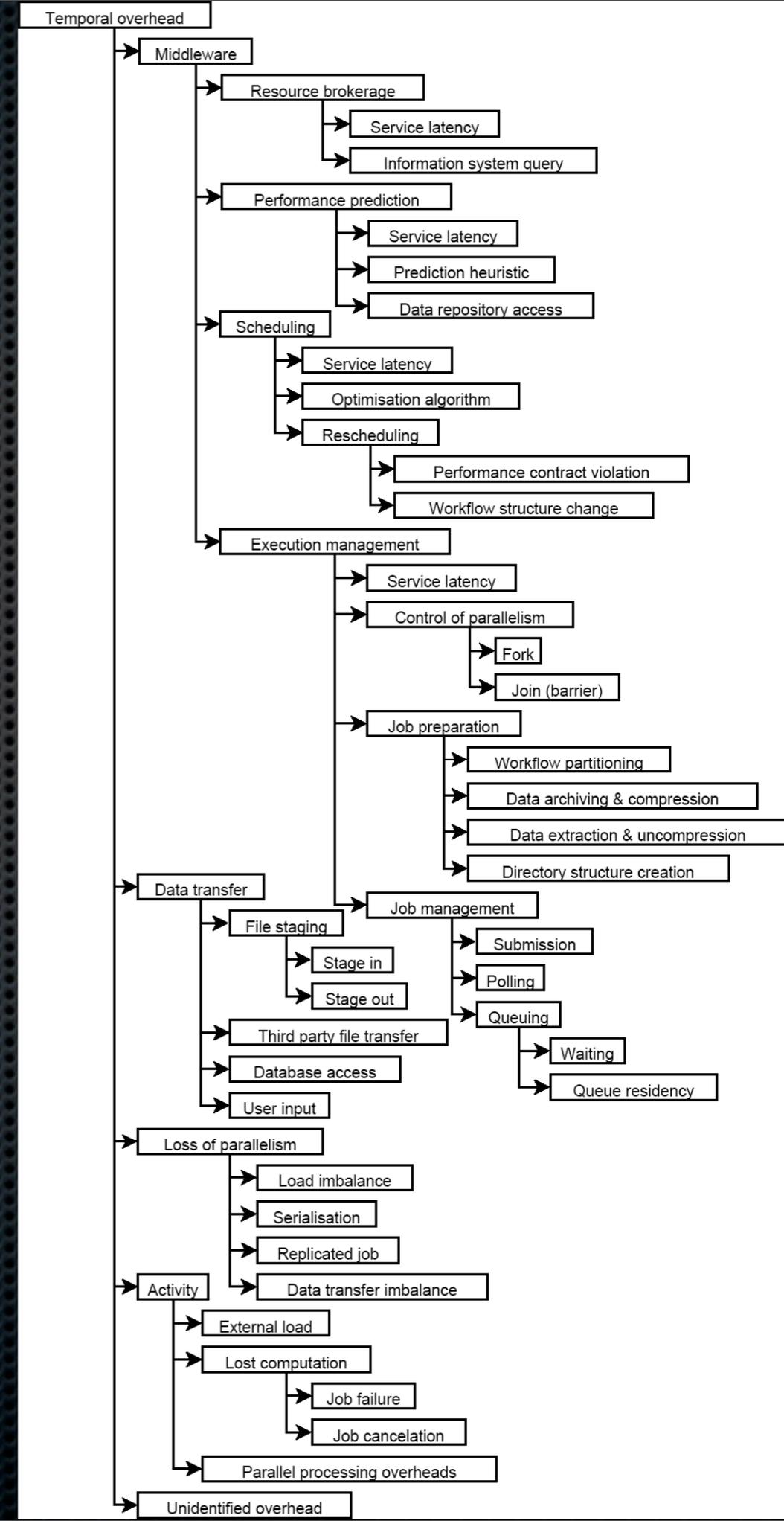
✦ Ideal time:

$$T_{ideal} = \frac{1}{\sum_{\forall Site \in Grid} (T_{sequential}^{Site})^{-1}}$$

✦ Realistic theoretical bound of what the users can achieve for an application

✦ Wallclock time = Ideal time + overheads (non-overlapping)

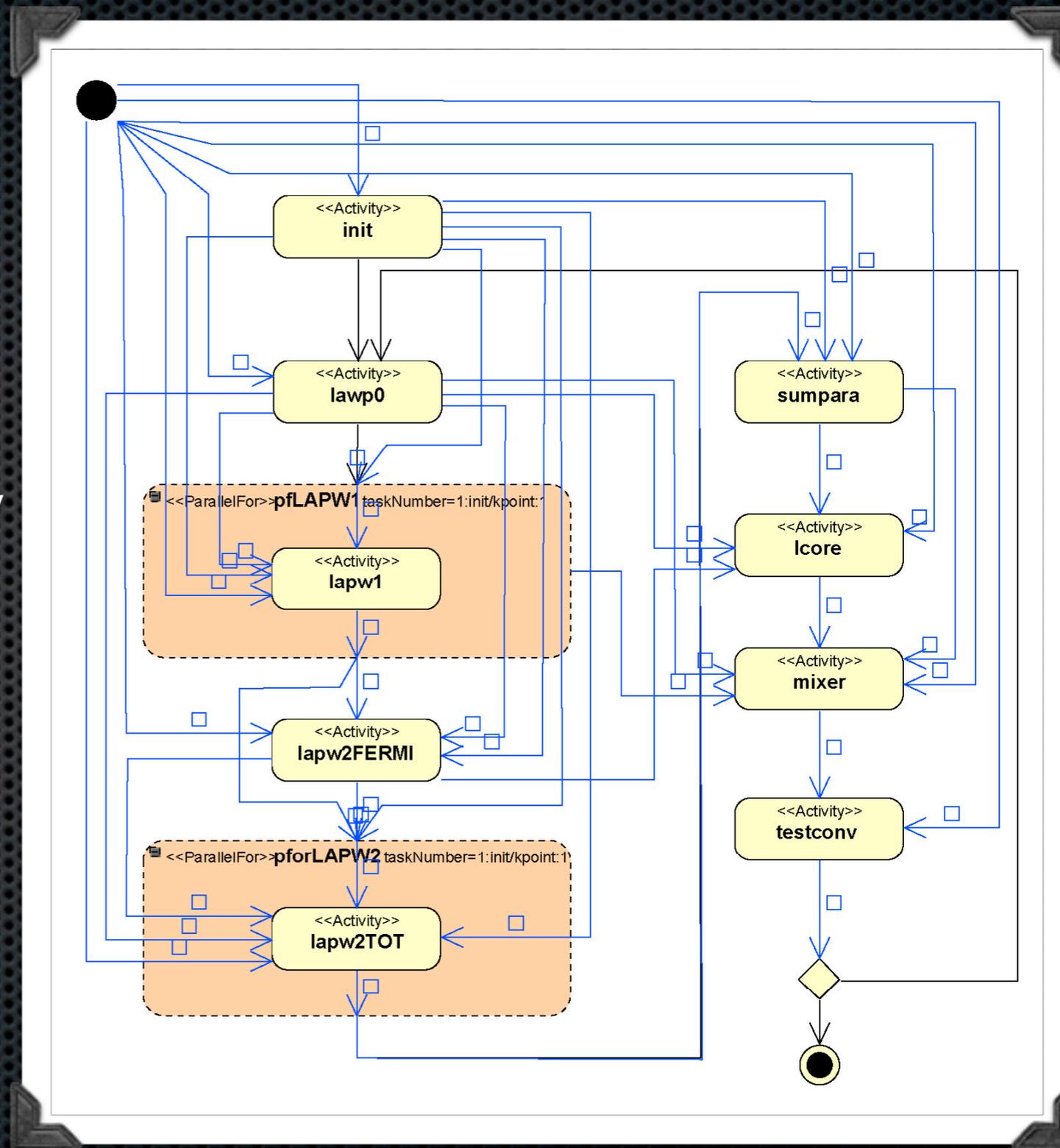
✦ Overheads = communication + synchronisation + waiting time + latencies + ...



Examples

Wien2K Material Science with ASKALON

- Material science application
- electronic structure calculations of solids using density functional theory
- Institute of Theoretical Chemistry, Vienna University of Technology
- 1000+ user groups world-wide
- We have ported the application as a Grid workflow
- Seven activity types
- 2 activity types have 200-300 parallel instances



Persistence of Vision Raytracer (POVray)

- ✦ Free portable tool for creating high-quality 3-dimensional graphics

- ✦ <http://www.povray.org>

- ✦ Input

- ✦ pov files are descriptions of scenes and objects
 - ✦ ini files are rendering configuration files

- ✦ Output

- ✦ png image for each frame
 - ✦ merge frames into a final movie

