Dynamic Workflows for Grid Applications

Fraunhofer Institut
Rechnerarchitektur
und Softwaretechnik

Dynamic Workflows for Grid Applications

Fraunhofer Resource Grid

Fraunhofer Institute for Computer Architecture and Software Technology – Berlin Germany

Andreas Hoheisel
(andreas.hoheisel@first.fraunhofer.de)

Uwe Der
(uwe.der@first.fraunhofer.de)

FhRG Fraunhofer Resource Grid

Fraunhofer FIRST Institut Rechnerarchitektur und Softwaretechnik

Outline

Fraunhofer Resource Grid

Describing Grid jobs with Petri Nets

Dynamic workflows

Fault management

Grid Job Handler

Conclusions and future work

Fraunhofer Resource Grid (FhRG)

Challenge

Development and implementation of a stable and robust grid infrastructure

Software layer on top of Globus to enable fast realizations of distributed applications

Integration of available resources

FIRST
Institut
Rechnerarchitektur
und Softwaretechnik

ITWM
Institut
Techno- und
Wirtschaftsmathematik

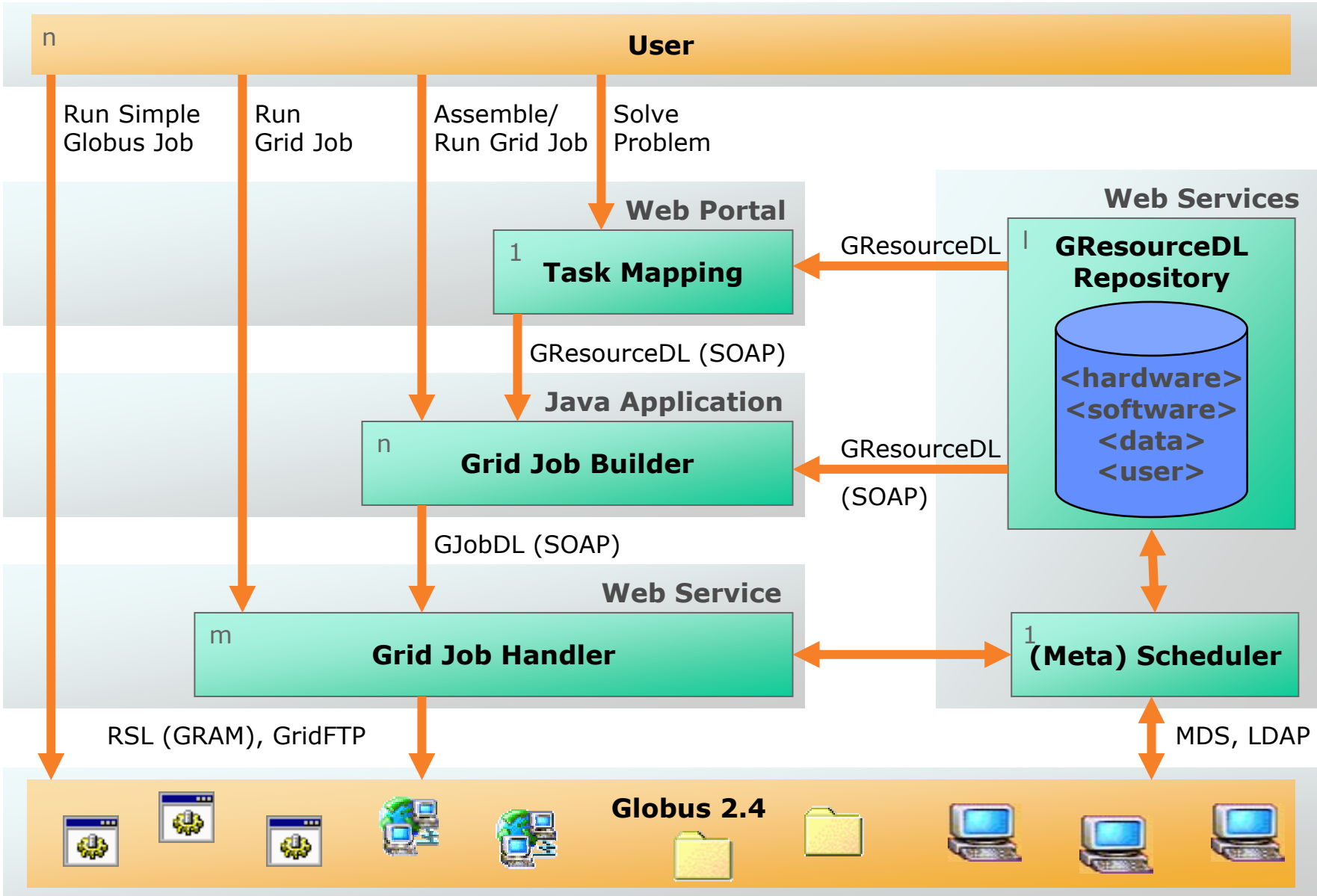
IAO
Institut
Arbeitswirtschaft und
Organisation

IGD
Institut
Graphische
Datenverarbeitung

SIT
Institut
Sichere Telekooperation

FhRG Fraunhofer Resource Grid

Fraunhofer FIRST Institut Rechnerarchitektur und Softwaretechnik


n
User
Run Simple Globus Job
Run Grid Job
Assemble/ Run Grid Job
Solve Problem
Web Portal
1
Task Mapping
GResourceDL
l
GResourceDL Repository
<hardware>
<software>
<data>
<user>
Web Services
GResourceDL (SOAP)
Java Application
n
Grid Job Builder
GResourceDL
(SOAP)
GJobDL (SOAP)
Web Service
m
Grid Job Handler
1
(Meta) Scheduler
RSL (GRAM), GridFTP
MDS, LDAP
Globus 2.4

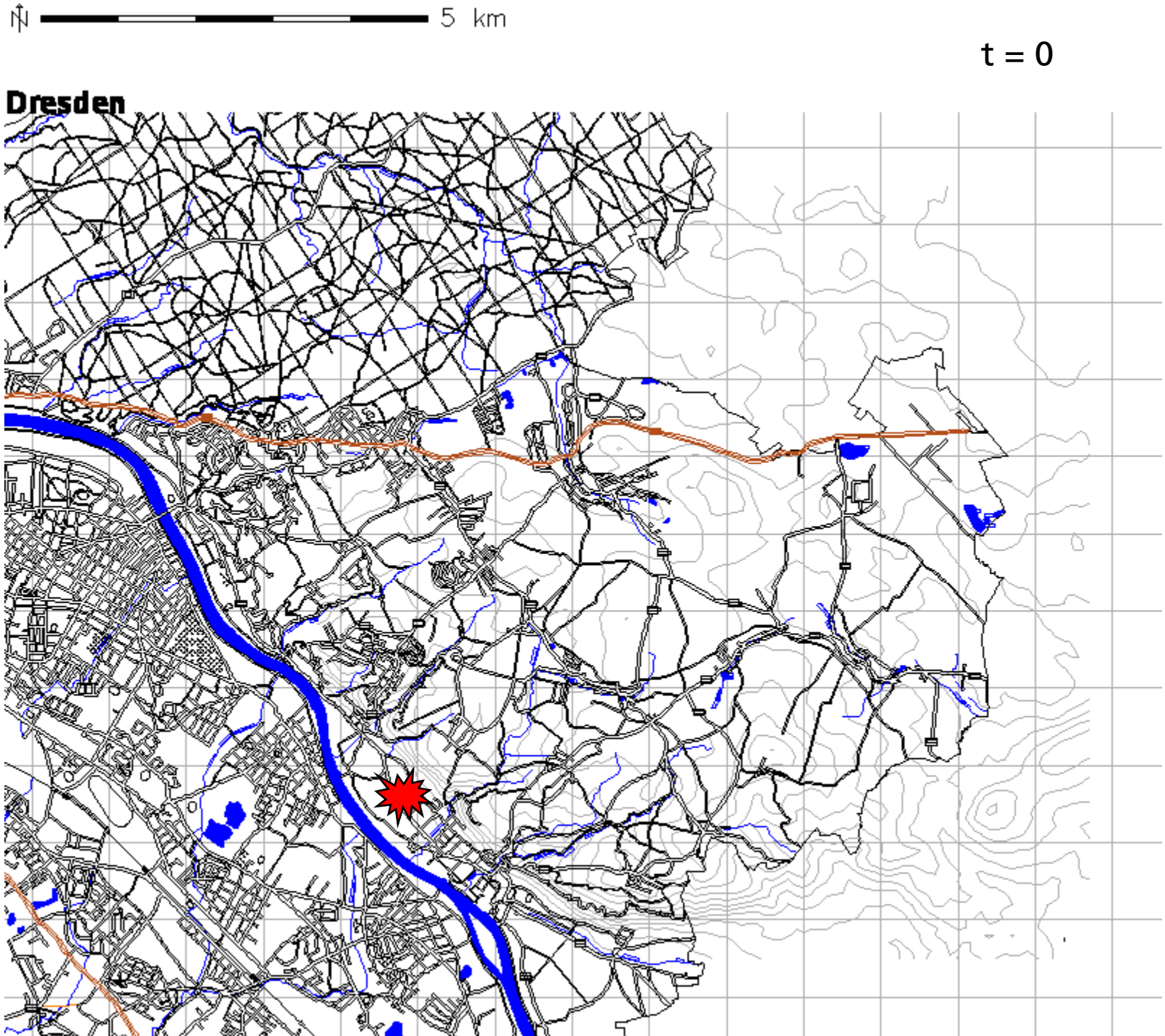
Describing Grid Jobs with Petri Nets

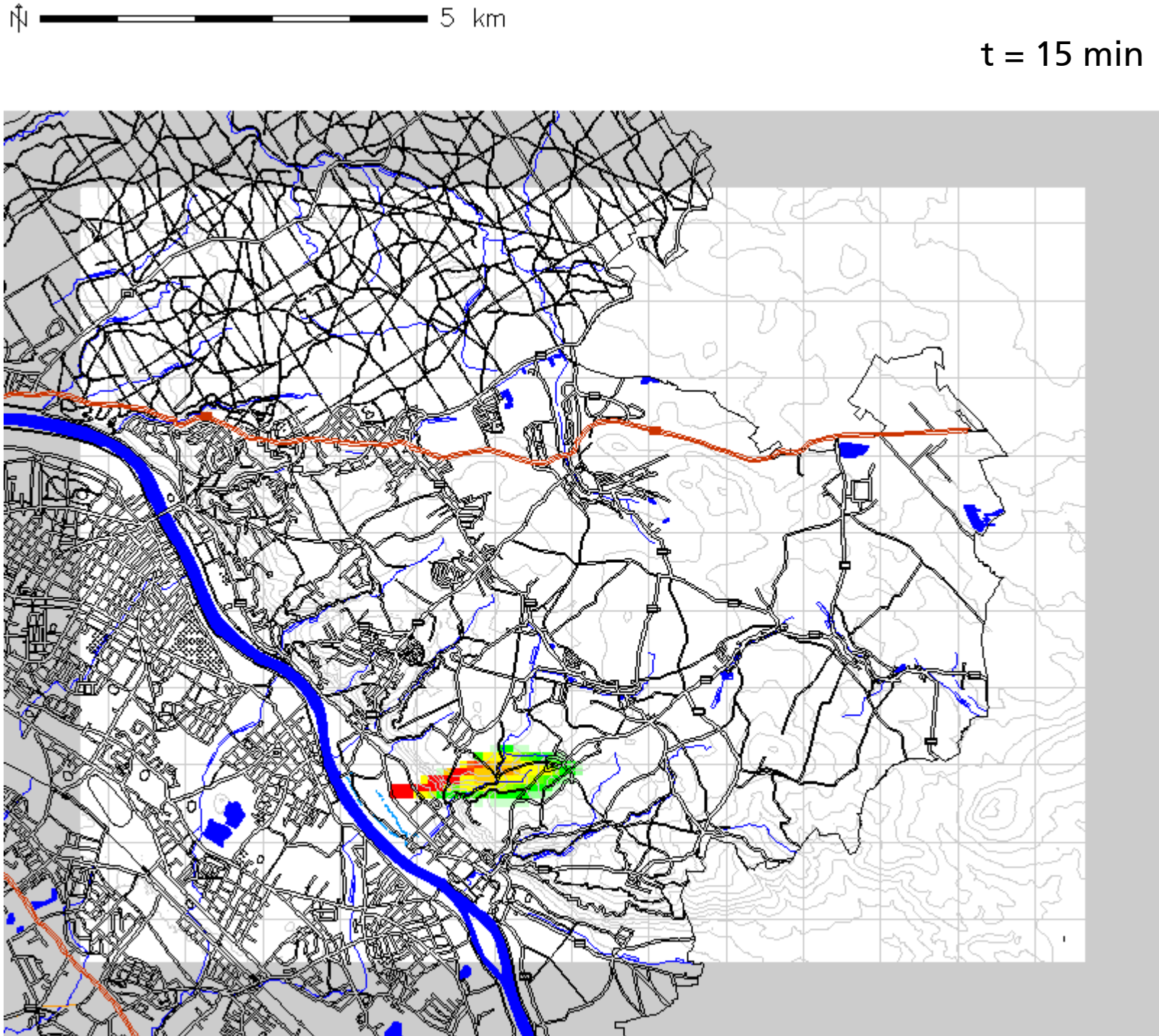
What is a Grid Job?

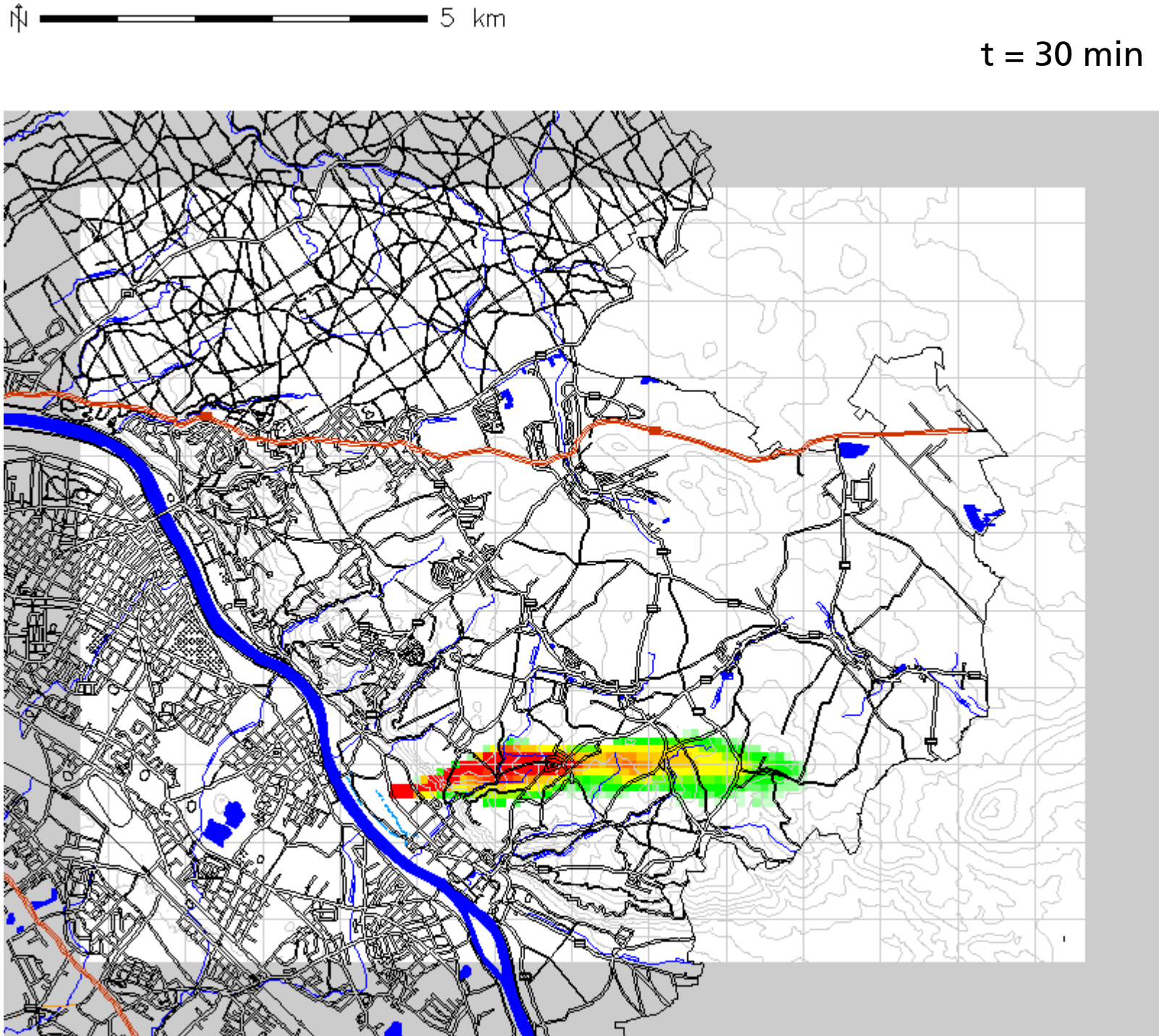
Grid job	Composition of Grid resources forming grid applications
Grid resource	Software, hardware, data, (people)
Atomic job	Single task, indivisible component of a Grid job Execution of a software component with input data Future: Invocation of a WebService method call (?)
Component Model	Loosely coupled software components Executables that read input files and write output files Communication via files and GridFTP
GJobDL	Description of Grid jobs on an abstract level Independent from Grid infrastructure Connecting software components and data Based on Petri Nets XML

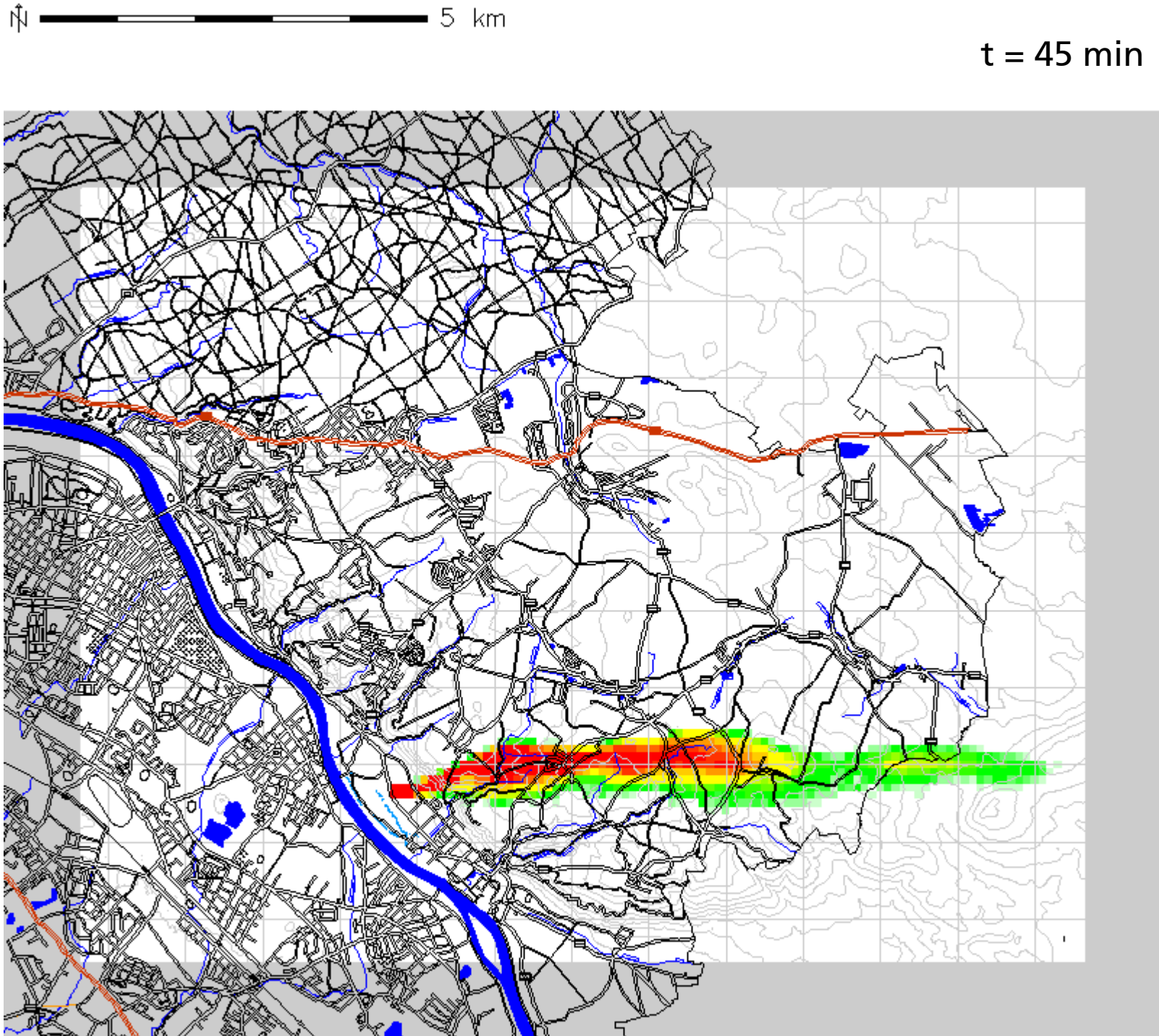
Example Grid job:

Environmental Risk Analysis and Management System (ERAMAS)


N
5 km
t = 0
Dresden

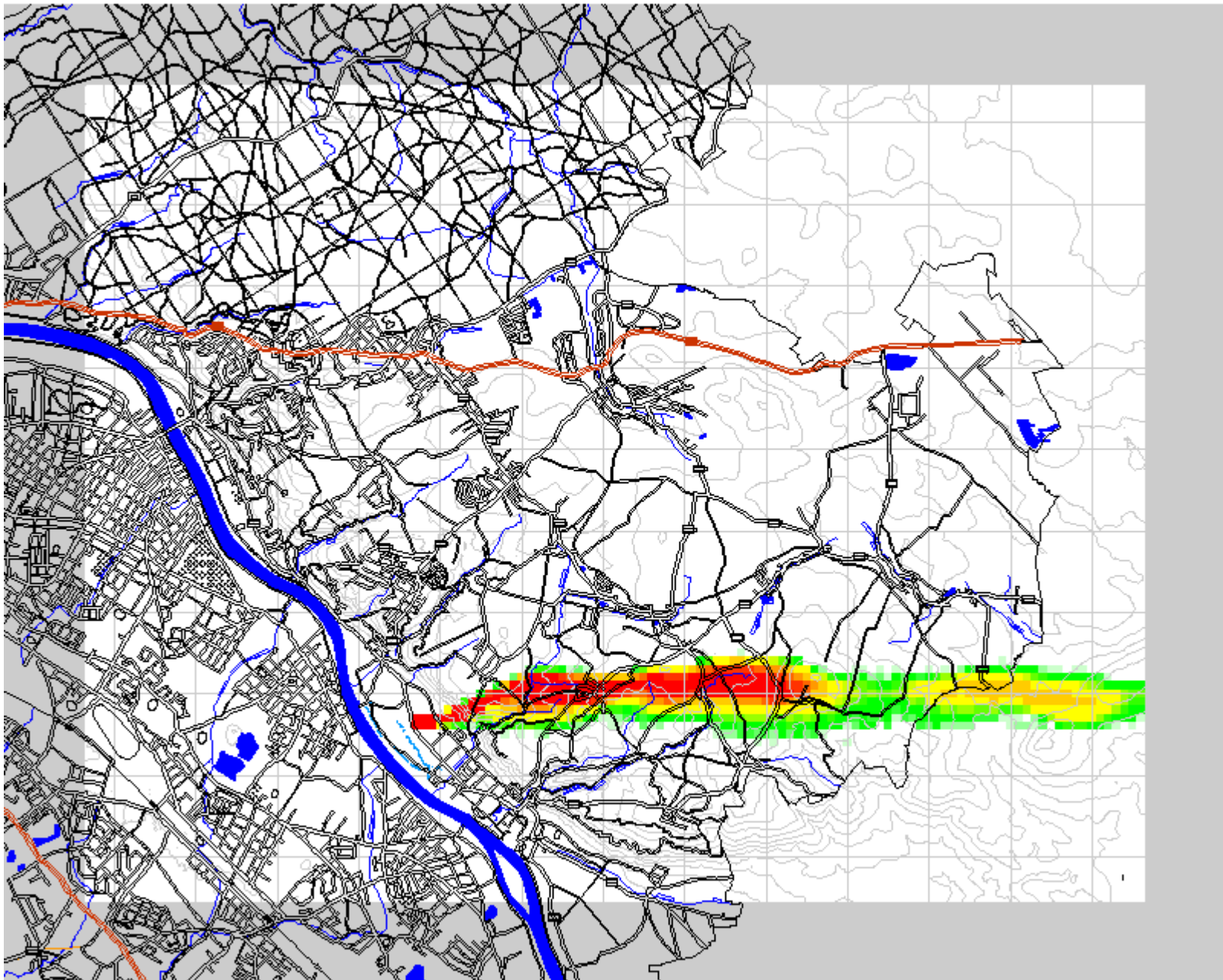

N
5 km
t = 15 min


N
5 km
t = 30 min


N
5 km
t = 45 min

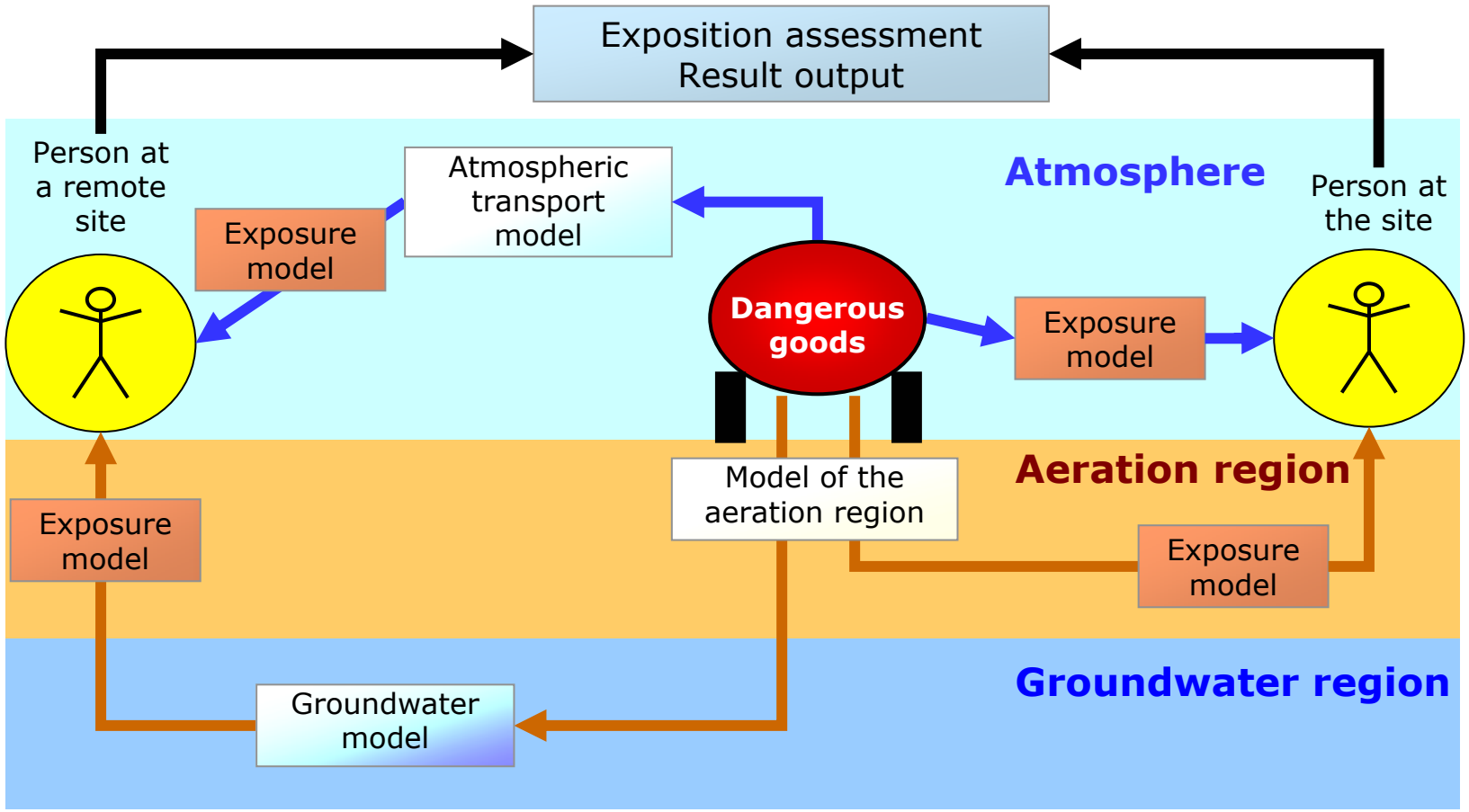
N 5 km

t = 60 min



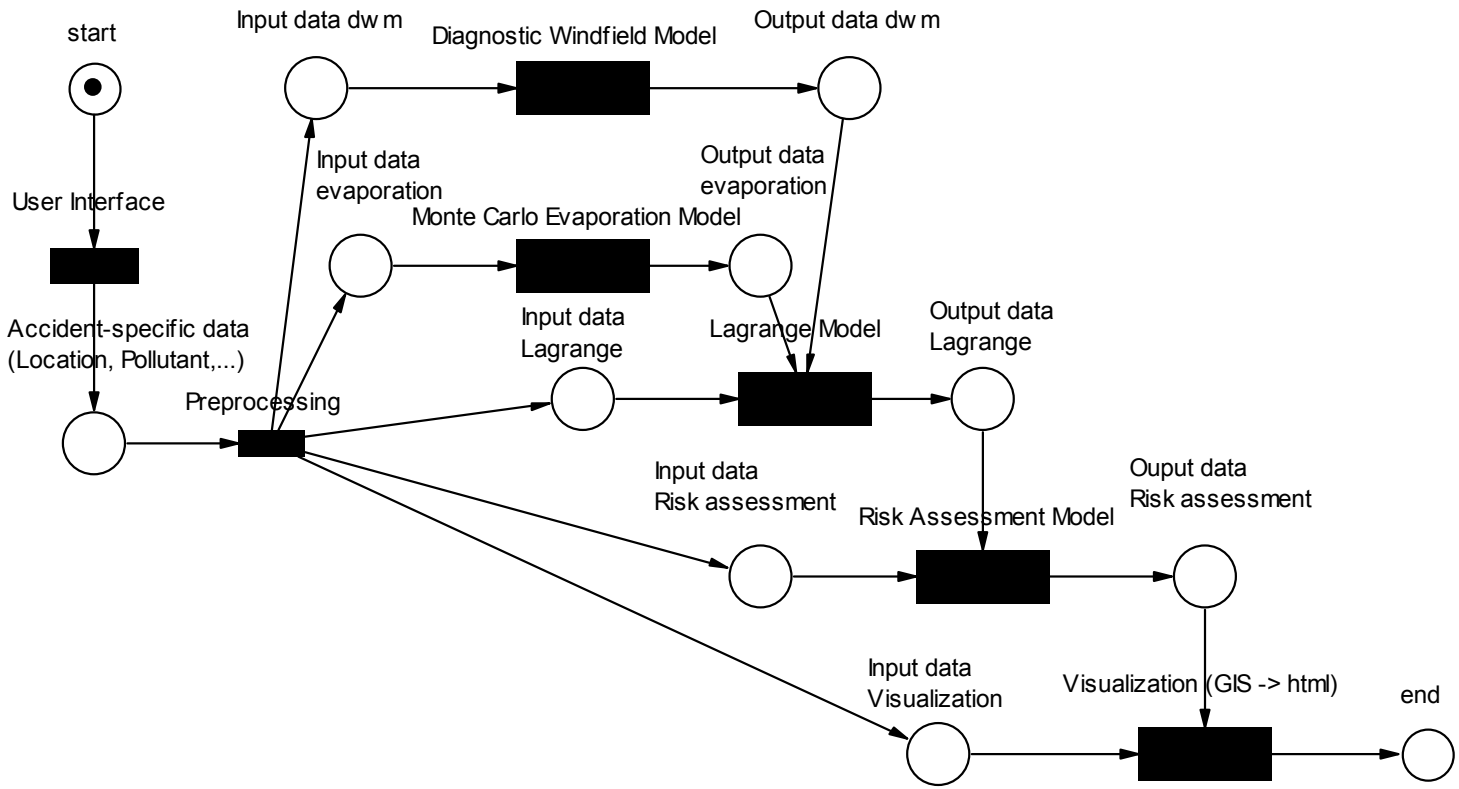
ERAMAS

ERAMAS	Environmental Risk Analysis and Management System
	Simulation-based analysis and management system for environmental risks caused by dangerous substances
Problem	Release of carcinogenic and chemically toxic substances
Scenario	Accidents in industrial installations
	Transport of dangerous goods
	Terrorist attacks



ERAMAS – Pollutant Transport in the Atmosphere:

Accident → Source → Atmospheric Transport → Exposure



Why Petri Nets?

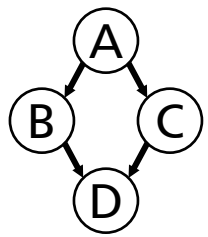
Problem

Description of complex workflows of grid jobs

DAG

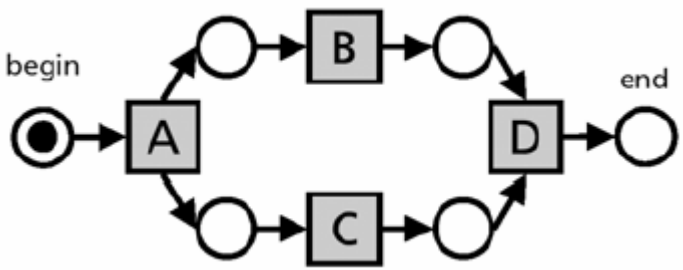
Directed Acyclic Graph (see Condor, Cactus, UNICORE)
no bidirectional coupling (interaction)
no loops

```
PARENT A CHILD B C
PARENT B C CHILD D
```



Petri Nets

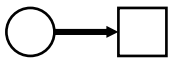
Graphical flow control of discrete systems



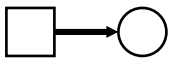
Petri Nets

Places Files, buffers, control places

Transitions Software components, control transitions



Arcs from places to transitions (Place is input place of transition)



Arcs from transitions to places (Place is output place of transition)

Tokens Data, State (done, failed)

Rule A transition is activated if all input places are filled with tokens and all output places have not reached their maximum capacity of tokens

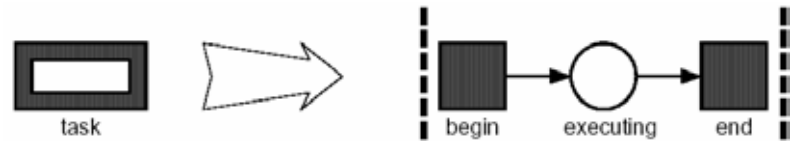
Refinement A single part of a Petri Net can be replaced by a sub Petri Net

Description of state A Petri Net describes workflow and state of a system

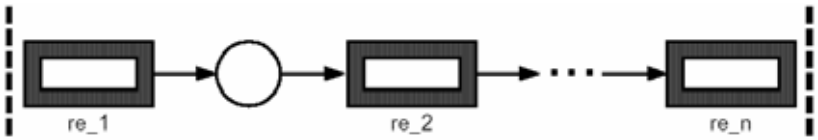
Petri Nets

(from *van der Aalst und Kumar, 2000*)

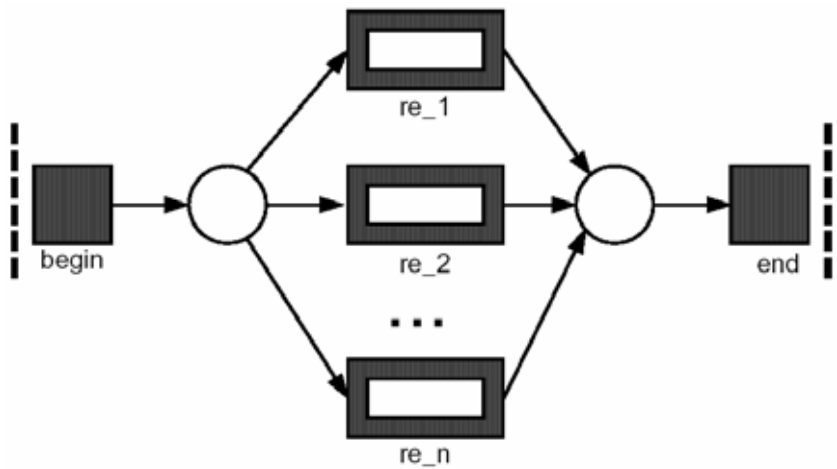
Task



Sequence

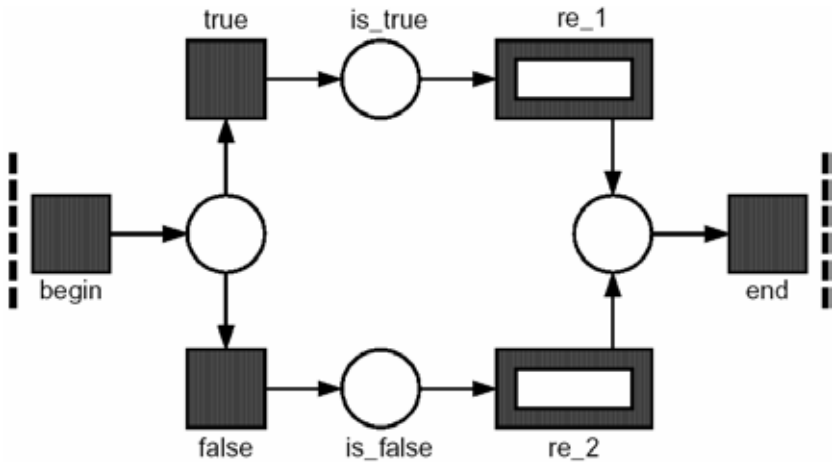


Choice



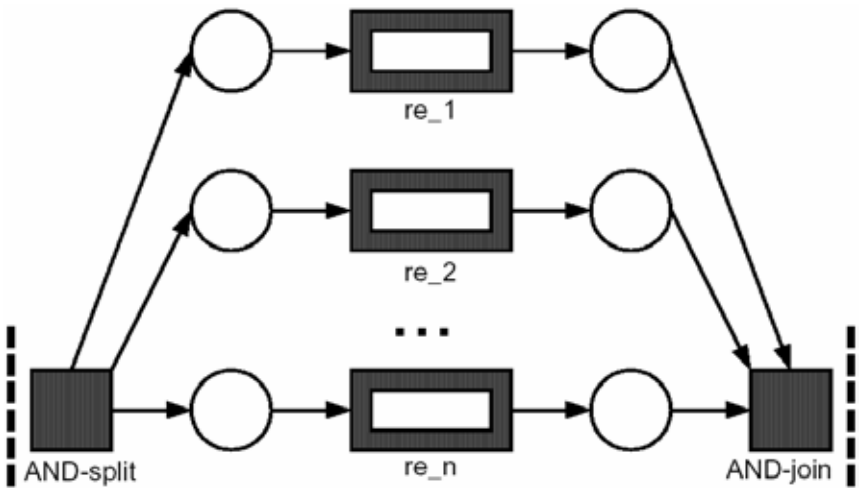
Petri Nets

Condition



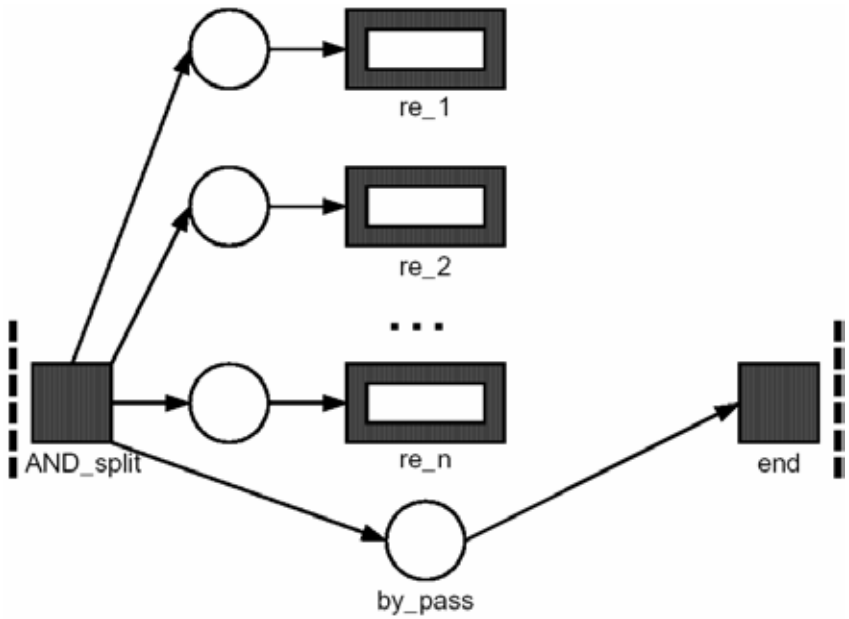
Petri Nets

Parallel execution with synchronization



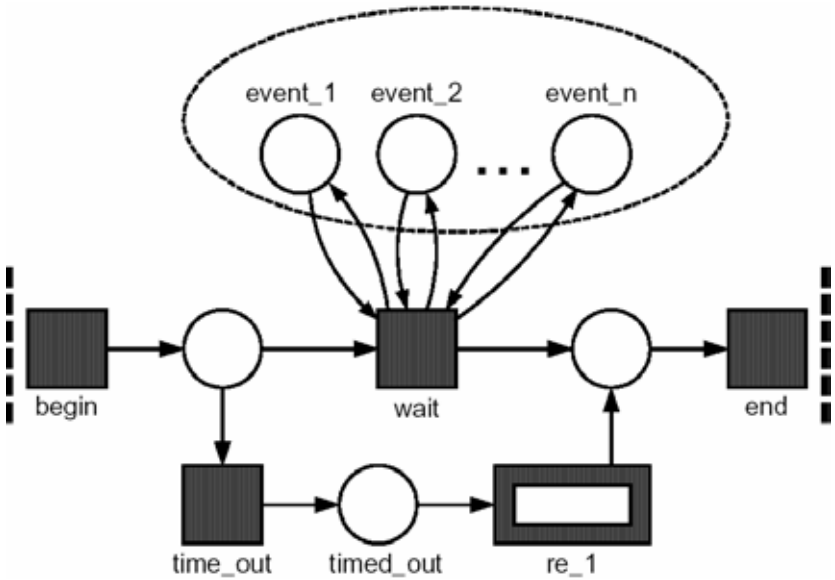
Petri Nets

Parallel execution without synchronization



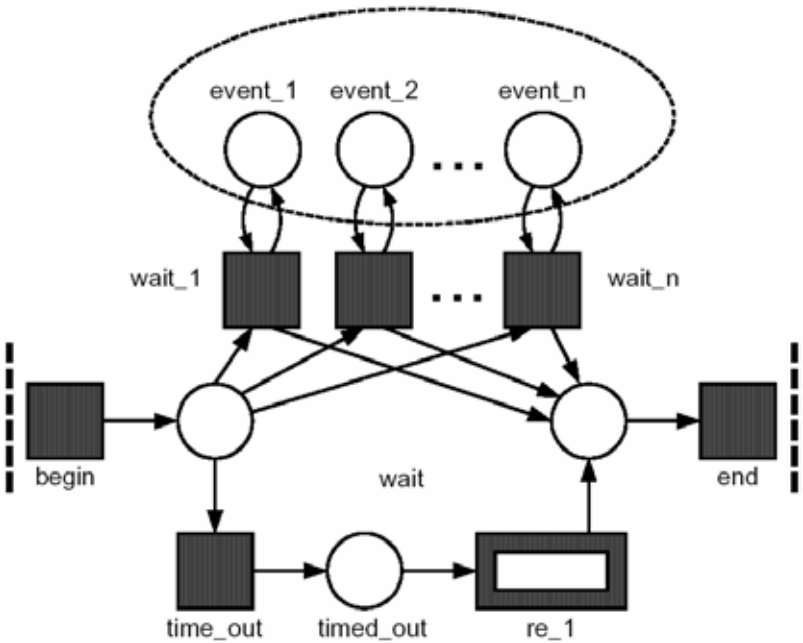
Petri Nets

Wait all with time out



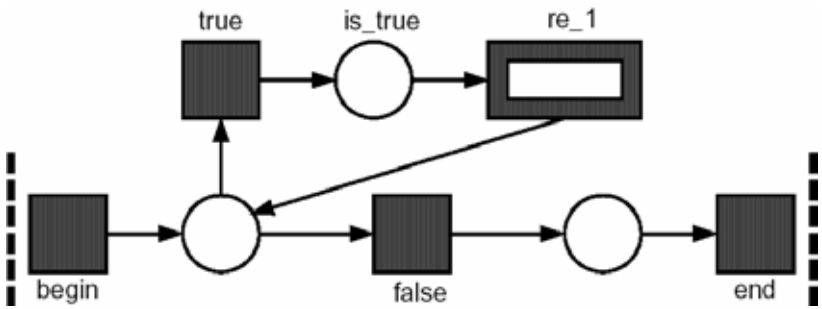
Petri Nets

Wait any with time out



Petri Nets

While … do



Dynamic Workflows

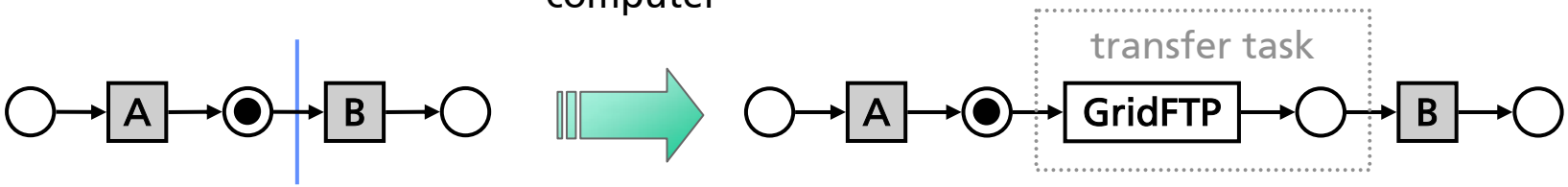
Dynamic Workflows

Refinement

The Grid Job Handler supplements the workflow during runtime by introducing additional tasks

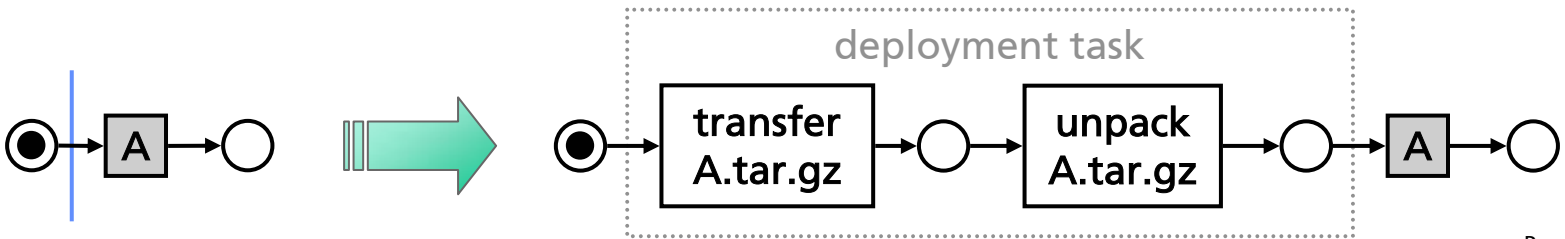
File transfer

A GridFTP task may be introduced automatically to transfer files that are not available on the remote computer



Software deployment

A software deployment task may be introduced to install software components on a remote computer

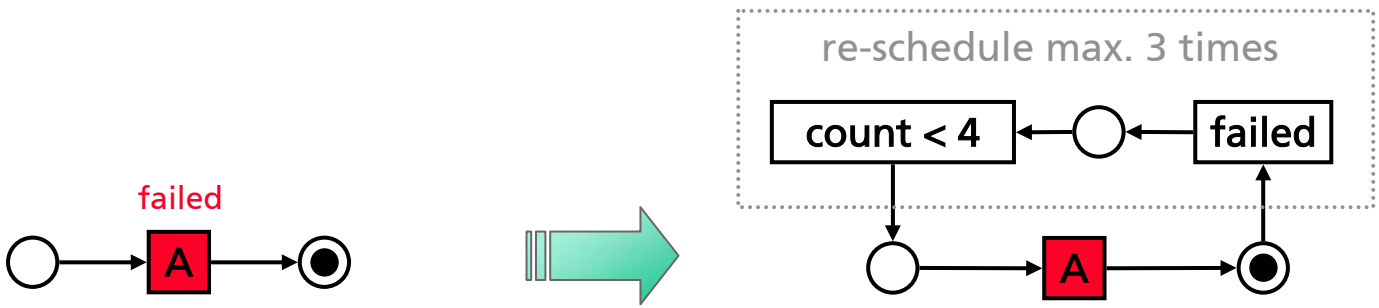


Fault Management

Implicit Fault Management

Grid middleware — Fault management that is included in the Grid middleware

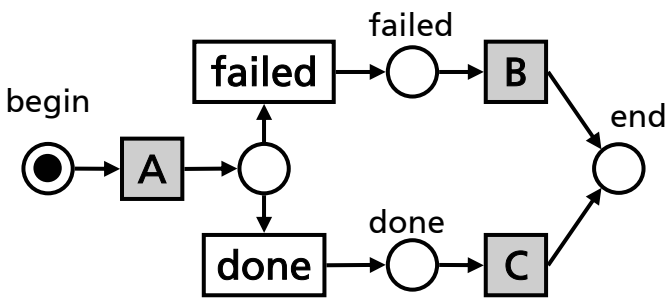
Petri Net refinement — Fault management tasks are introduced automatically if the submission or execution of a atomic task fails

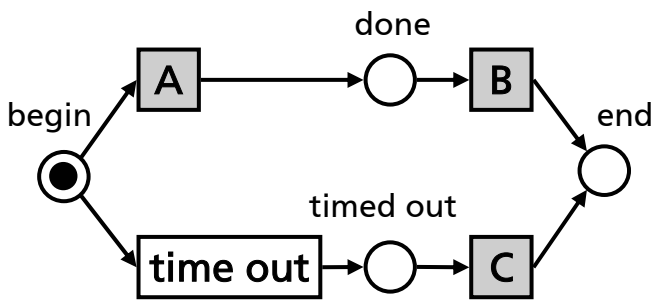


Explicit Fault Management

Petri Net workflow model

The user defines the fault management explicitly by including user-defined fault management tasks in the Petri Net of his Grid job





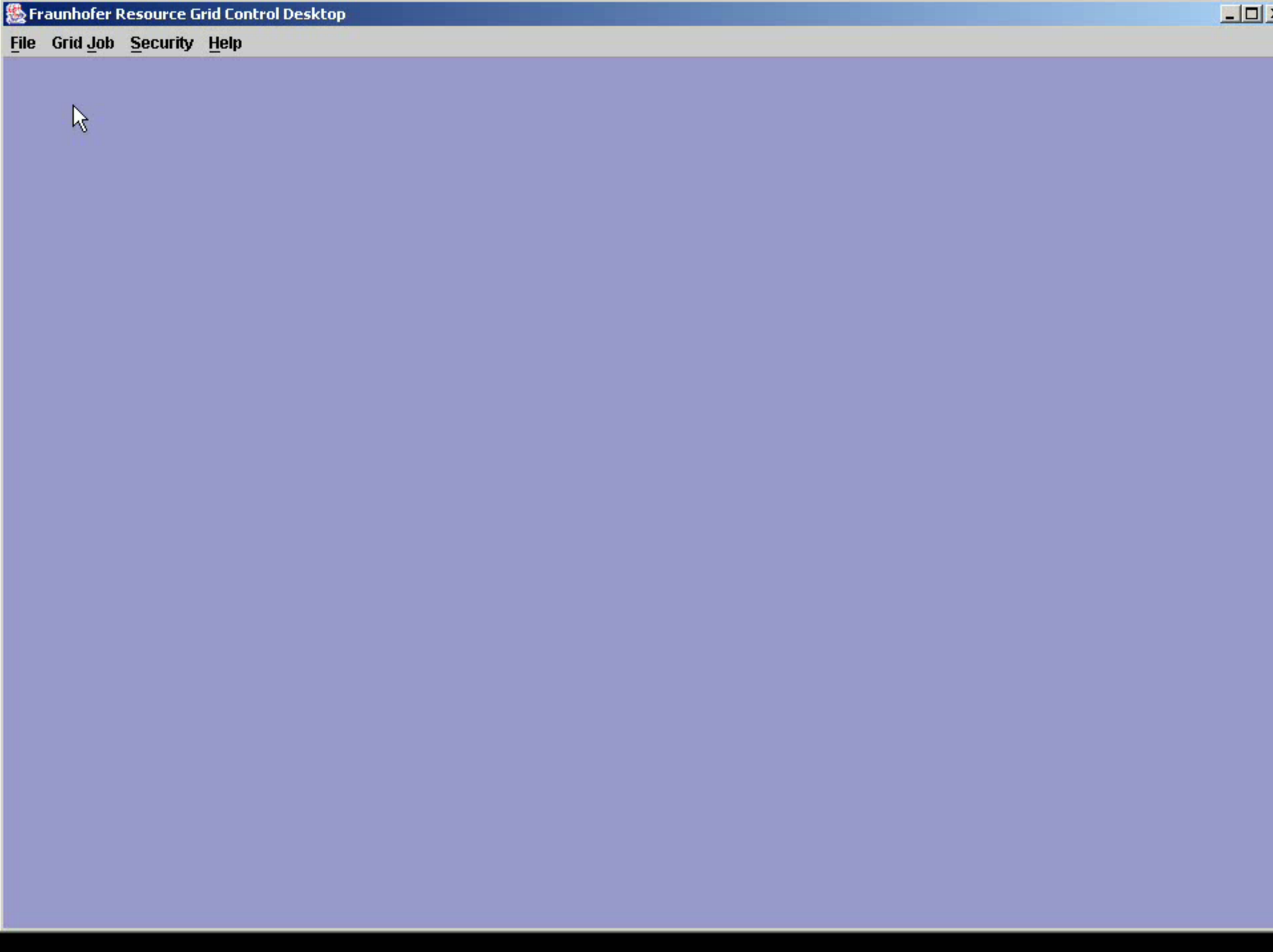
Grid Job Handler

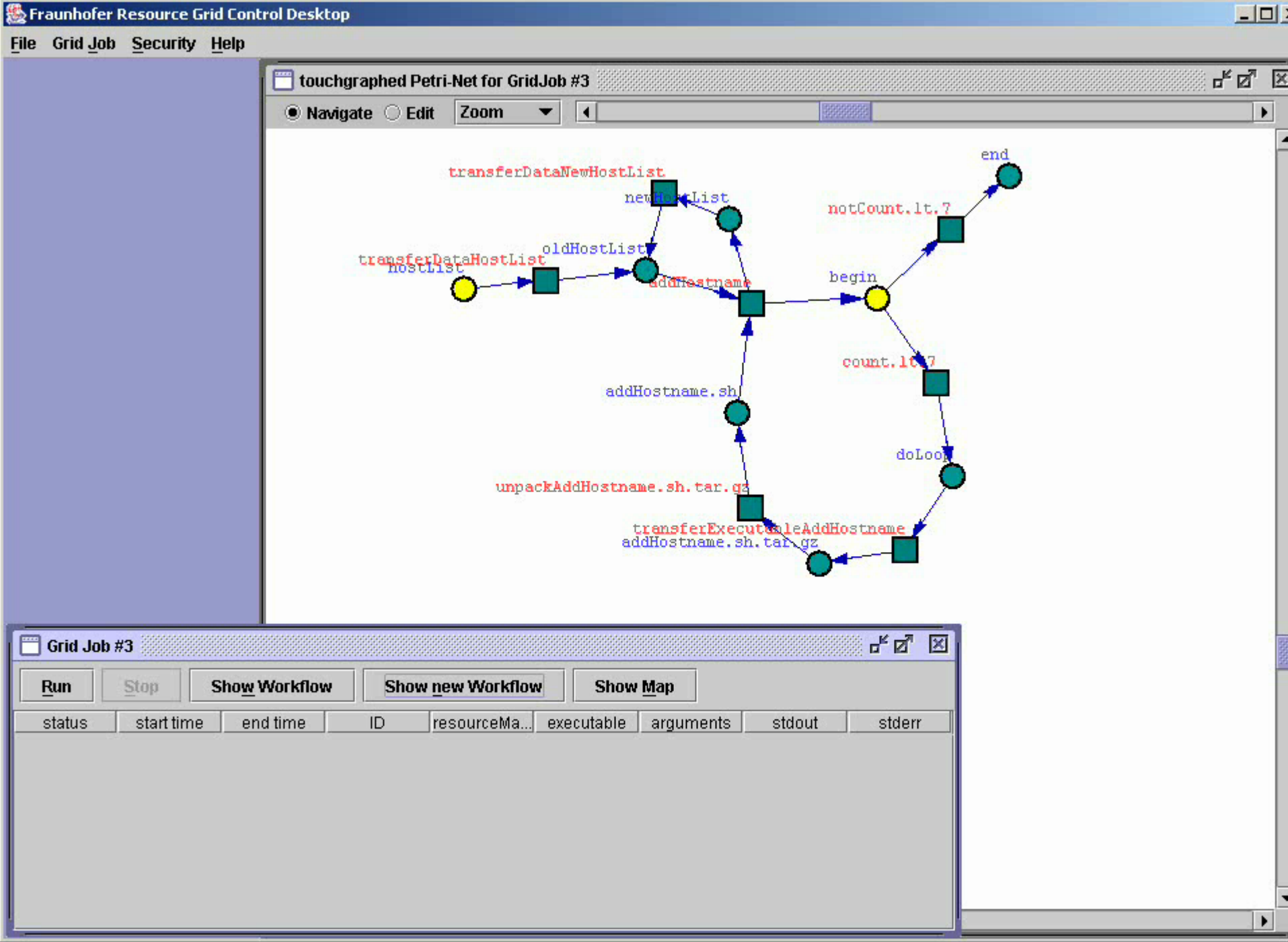
Grid Job Handler

Grid Job Handler	Software for executing and controlling coupled grid applications Workflow modeling Mapping of Grid Jobs onto appropriate distributed resources Execution of jobs using grid middleware (e.g. Globus → Java Commodity Grid Kit)
Component environment	Coupling of components using file input/output
Monitoring	Job status (pending, active, failed, done)
Integration into the FhRG	Communication with other FhRG services using SOAP (→ WebService)

Application flow of Grid Job Handler

- Read the GJobDL document
- Create Petri Net from this job description
- Verify the Petri Net (well-formedness, liveliness, deadlocks, pits, ...)
- Start the Grid Job (own thread)
- Collect all activated transitions
- Evaluate conditions
- Invoke resource mapping → repository, (meta-)scheduler
- **Refine the Petri Net**
  → insert GridFTPs, fault management, etc. if necessary
- Create and submit atomic jobs using grid middleware (e.g. GRAM)
- The transition fires, if atomic job is “done“ or has “failed“.


Fraunhofer Resource Grid Control Desktop
File
Grid Job
Security
Help

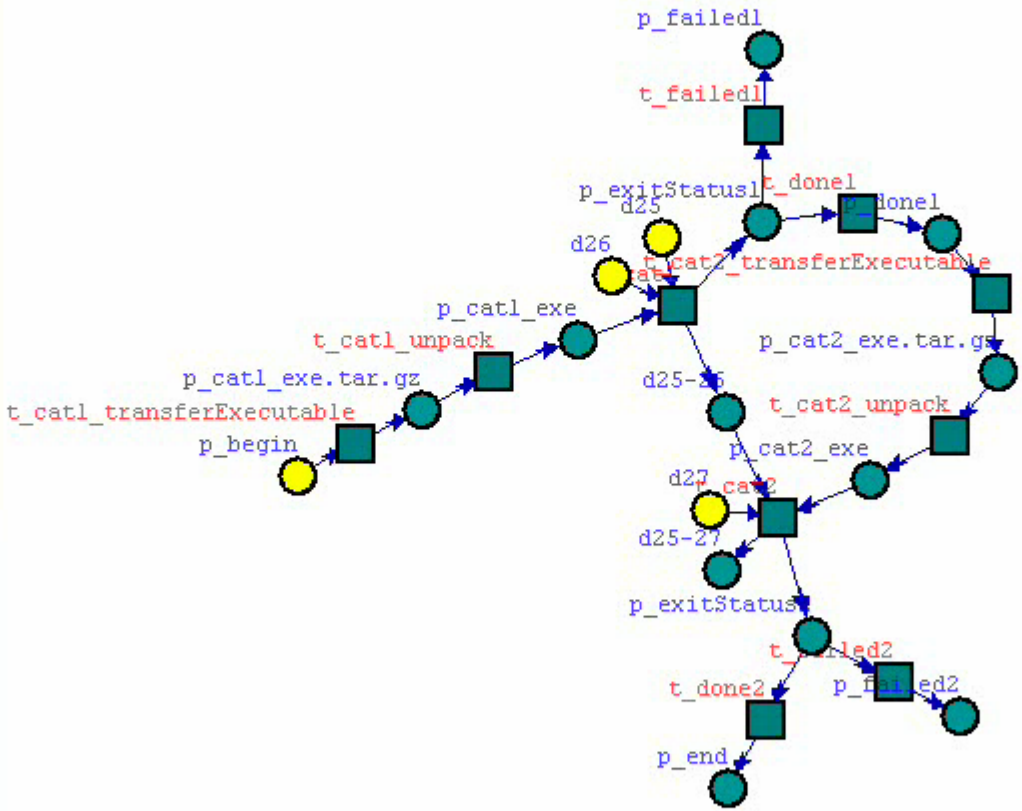

Fraunhofer Resource Grid Control Desktop
File
Grid Job
Security
Help
touchgraphed Petri-Net for GridJob #3
Navigate
Edit
Zoom
transferDataNewHostList
newHostList
oldHostList
transferDataHostList
hostList
addHostname
begin
notCount.lt.7
end
addHostname.sh
unpackAddHostname.sh.tar.gz
transferExecutableAddHostname
addHostname.sh.tar.gz
Grid Job #3
Run
Stop
Show Workflow
Show new Workflow
Show Map
status
start time
end time
ID
resourceMa...
executable
arguments
stdout
stderr

Fraunhofer Resource Grid Control Desktop

File Grid Job Security Help

touchgraphed Petri-Net for GridJob #5

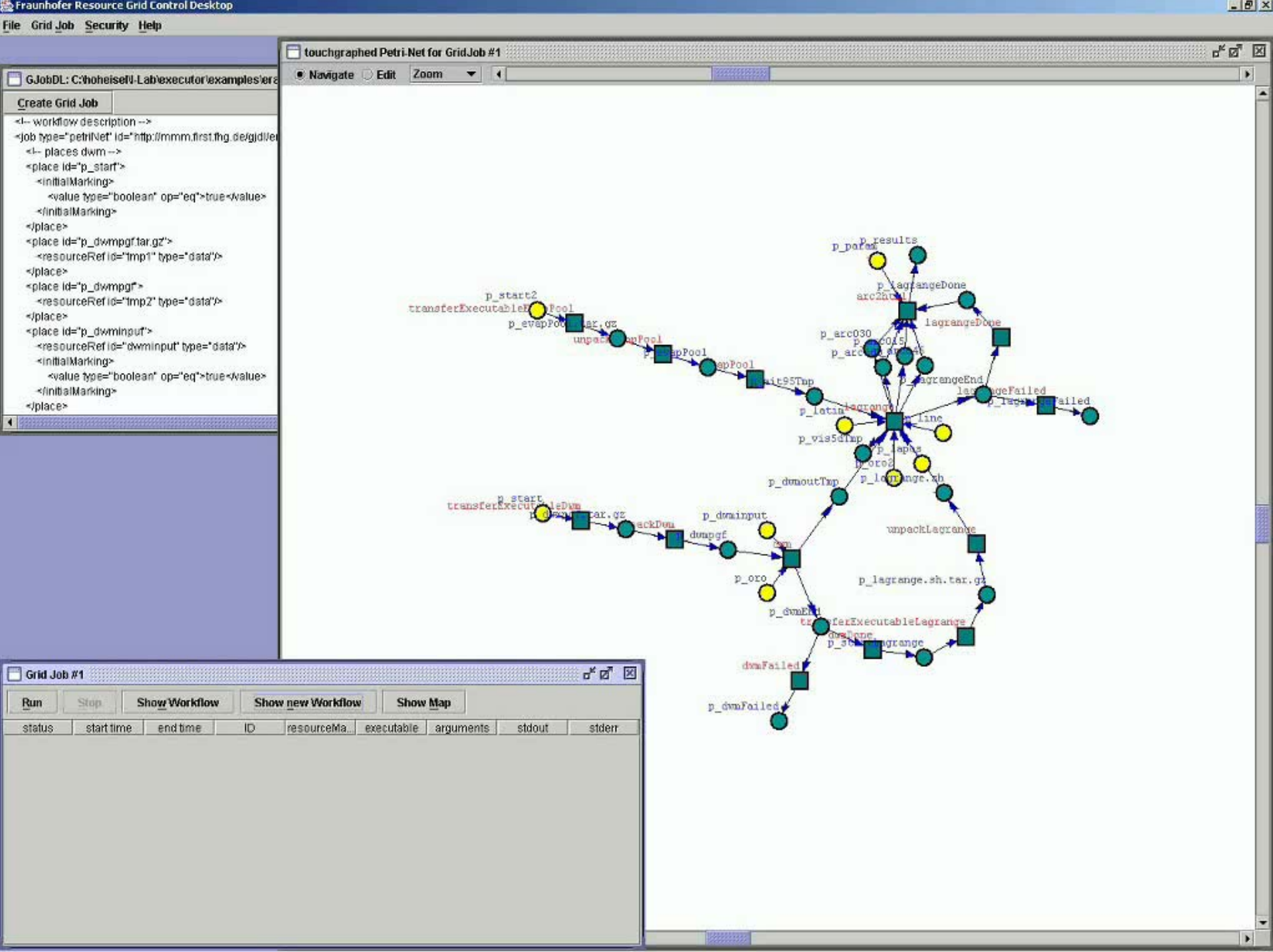
Navigate Edit Zoom



Grid Job #5

Run Stop Show W

status st... e... ID r... e... ... ...


Fraunhofer Resource Grid Control Desktop
File
Grid Job
Security
Help
GJobDL: C:\hoheisel\I-Lab\executor\examples\era
Create Grid Job
<!-- workflow description -->
<job type="petriNet" id="http://mmm.first.fhg.de/gridVer
<!-- places dwm -->
<place id="p_start">
<initialMarking>
<value type="boolean" op="eq">true</value>
</initialMarking>
</place>
<place id="p_dwmpgf.tar.gz">
<resourceRef id="tmp1" type="data"/>
</place>
<place id="p_dwmpgf">
<resourceRef id="tmp2" type="data"/>
</place>
<place id="p_dwminput">
<resourceRef id="dwminput" type="data"/>
<initialMarking>
<value type="boolean" op="eq">true</value>
</initialMarking>
</place>
touchgraphed Petri-Net for GridJob #1
Navigate
Edit
Zoom
p_results
p_lagrangeDone
p_arc030
p_lagrangeEnd
p_vis5dTmp
p_dumoutTmp
p_dwminput
unpackLagrange
p_lagrange.sh.tar.gz
p_oro
dumpgf
dwmFailed
Grid Job #1
Run
Stop
Show Workflow
Show new Workflow
Show Map
status
start time
end time
ID
resourceMa..
executable
arguments
stdout
stderr

Conclusions and Future Work

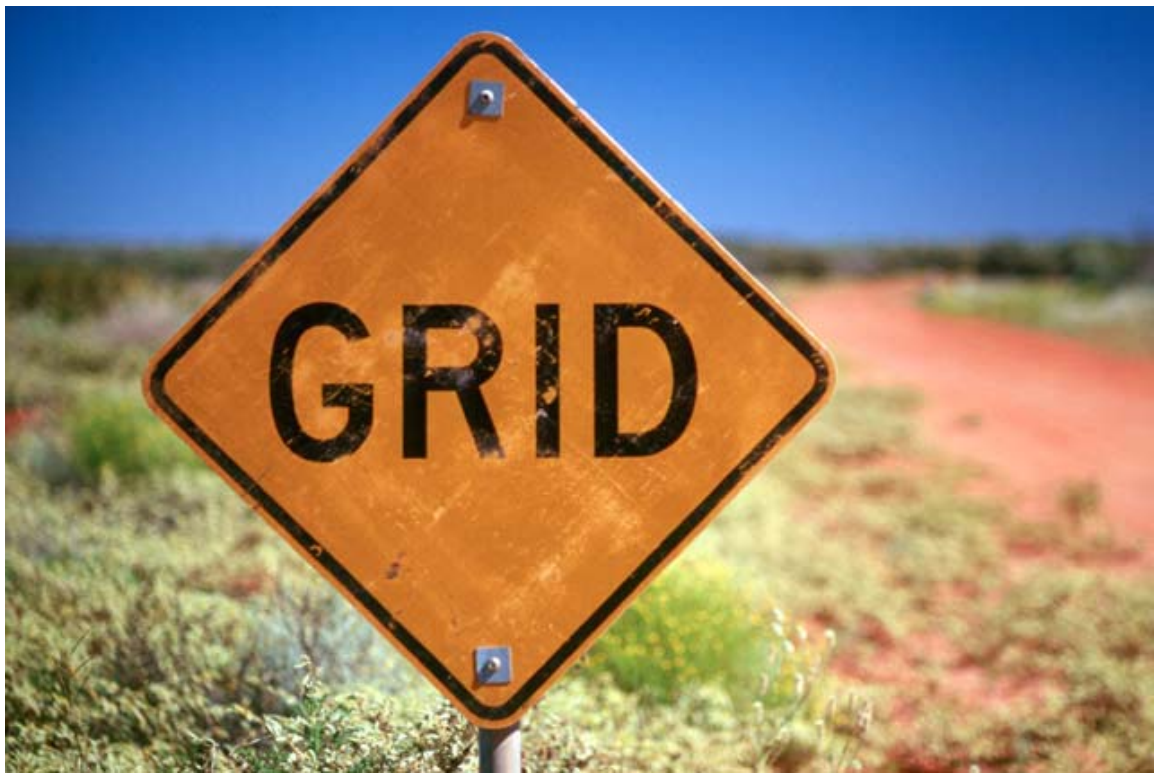
Conclusions

Description of workflow	GJobDL uses Petri Nets instead of directed acyclic graphs to model workflow of Grid jobs
Petri Nets	Easy orchestration of complex workflows, including conditions and loops
Dynamic workflow model	Petri Nets can be refined and modified during runtime Adding new tasks to the workflow, e.g.: transfer tasks software deployment tasks fault management tasks
Fault Management	implicit → automatic, included in Grid middleware explicit → user-defined, included in workflow model

Future Work

Petri Nets and OGSA?	How does workflow management with Petri Nets adapt to OGSA?
Tight coupling scheme?	Now: one transition → one executable
	Future: one transition → one method call (?) ➔ Grid Service, Web Service
Simulation of Petri Nets	Prediction of Petri Nets for advanced reservation of resources (→ scheduler) based on software und hardware benchmarks
Fault management	Workflow check pointing, recovery of grid jobs

More Information: http://www.fhrg.fhg.de/
http://www.andreas-hoheisel.de/
andreas.hoheisel@first.fraunhofer.de



FhRG Fraunhofer Resource Grid

Fraunhofer FIRST Institut Rechnerarchitektur und Softwaretechnik