EuroPVM/MPI 2006
Bonn, September 20, 2006

Orchestration of distributed MPI-Applications in a UNICORE-based Grid with MetaMPICH and MetaScheduling

Boris Bierbaum, Carsten Clauss, Thomas Eickermann, Lidia Kirtchakova, Arnold Krechel,

Stephan Springstubbe, Oliver Wäldrich, Wolfgang Ziegler

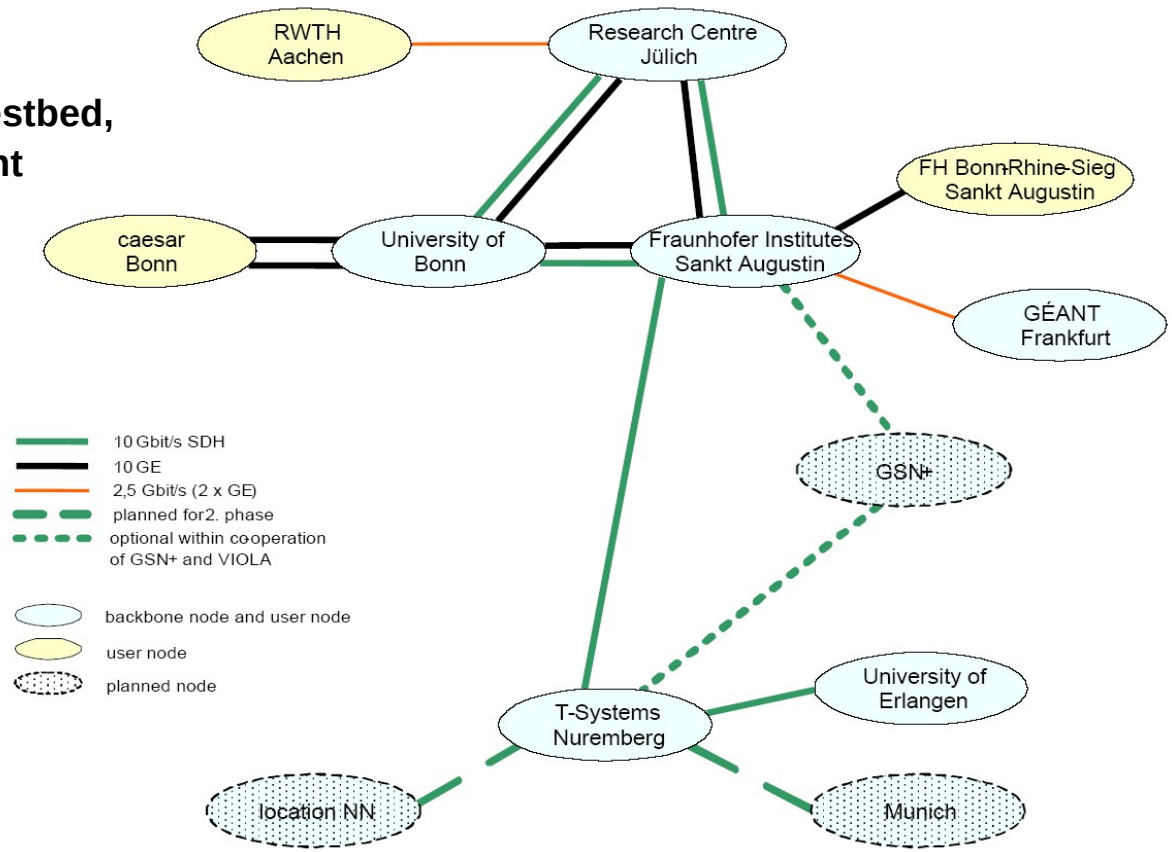
Fraunhofer SCAI Institut Algorithmen und Wissenschaftliches Rechnen

Forschungszentrum Jülich in der Helmholtz-Gemeinschaft

VIOLA

VIOLA-Networking

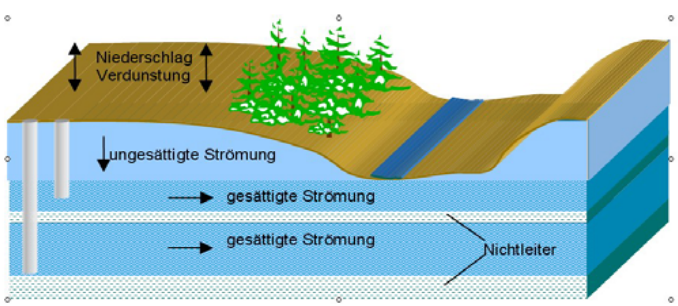
- **Deployment and operation of the testbed, test of advanced network equipment**
- **Signaling and reservation**
  - **bandwidth- and QoS-reservations in the network**
  - **interfaces for user-driven reservation: immediate and in advance**

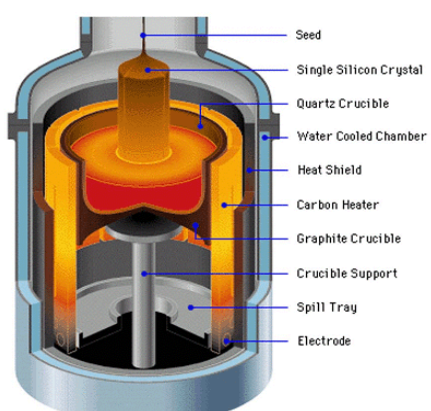


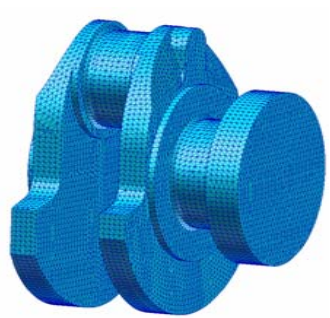
VIOLA Subprojects: Distributed Parallel Simulations

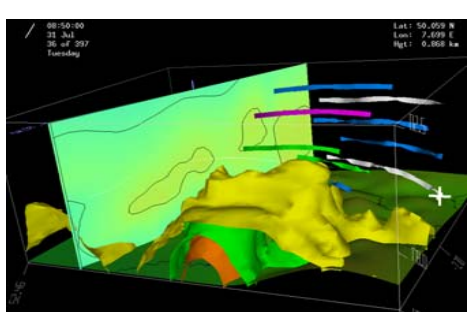
VIOLA-Applications (Multi-physics, Tele-collaboration)

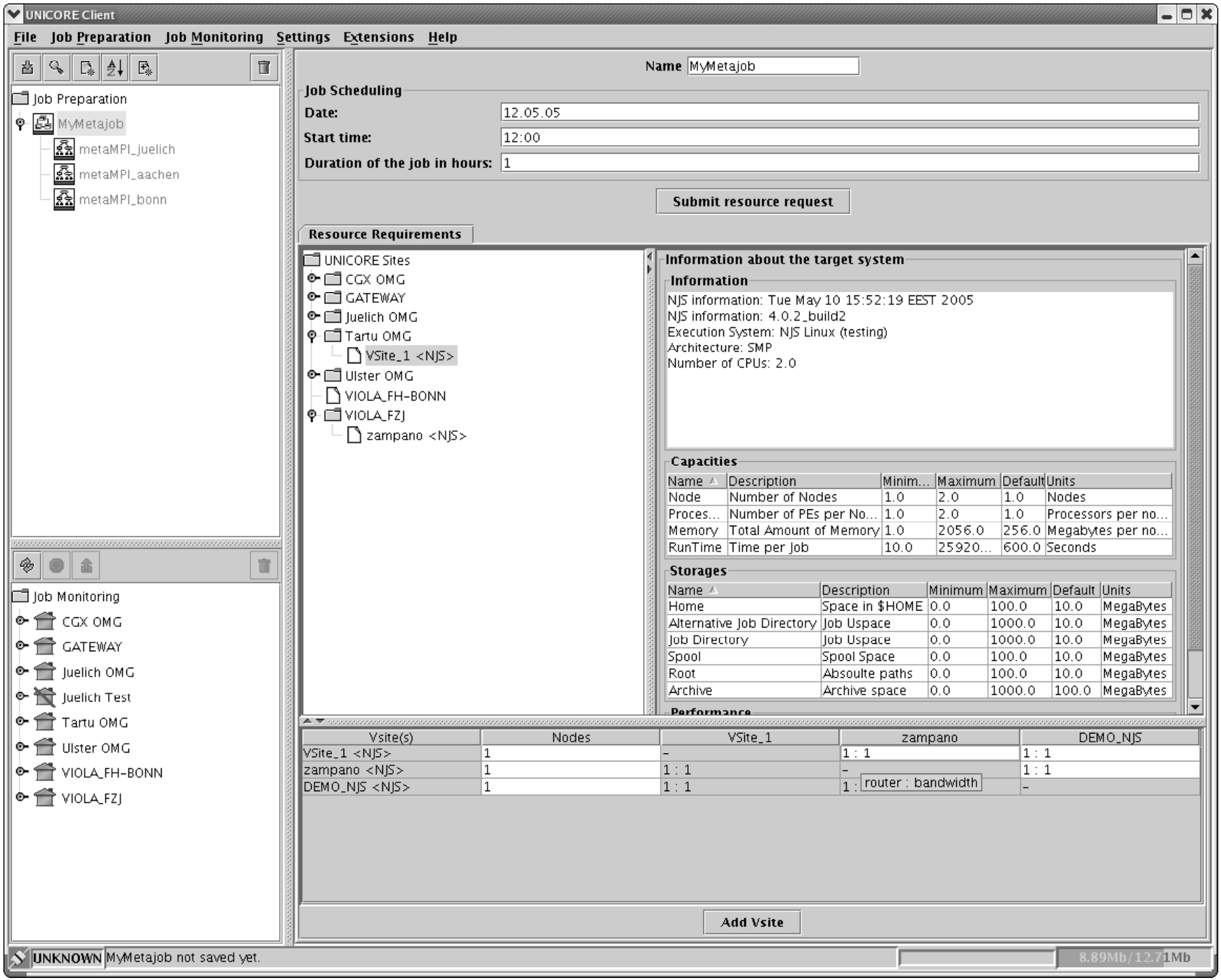
- **MetaTrace: Simulation of pollutant transport in groundwater with distributed SMP-Clusters (FZJ)**
- **TechSim: Distributed simulation of crystal growth and biosensors (Caesar)**
- **AMG-OPT: Parameter optimization and optimal algebraic solvers – Mechanical structure (SCAI)**
- **KoDaVis-Atmo: Collaborative visualization of huge atmospheric datasets (FZJ)**



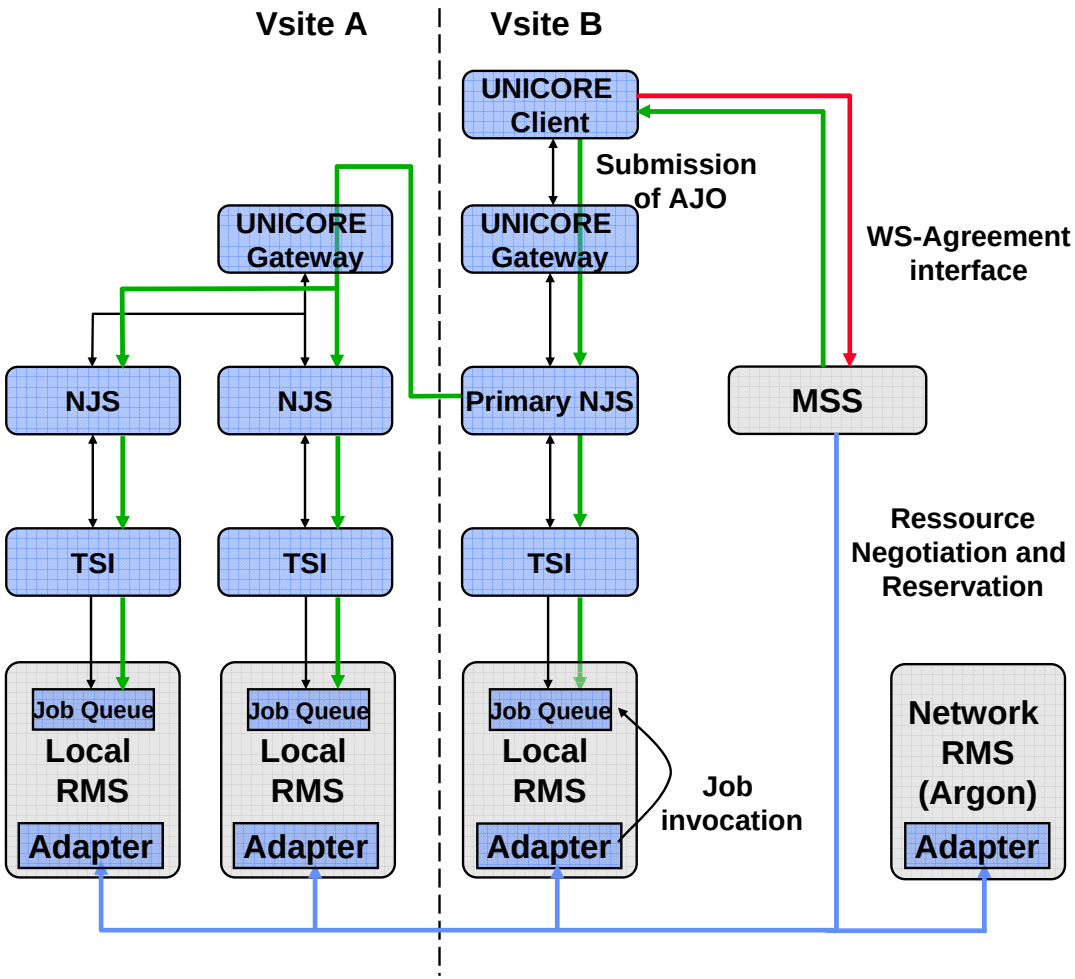






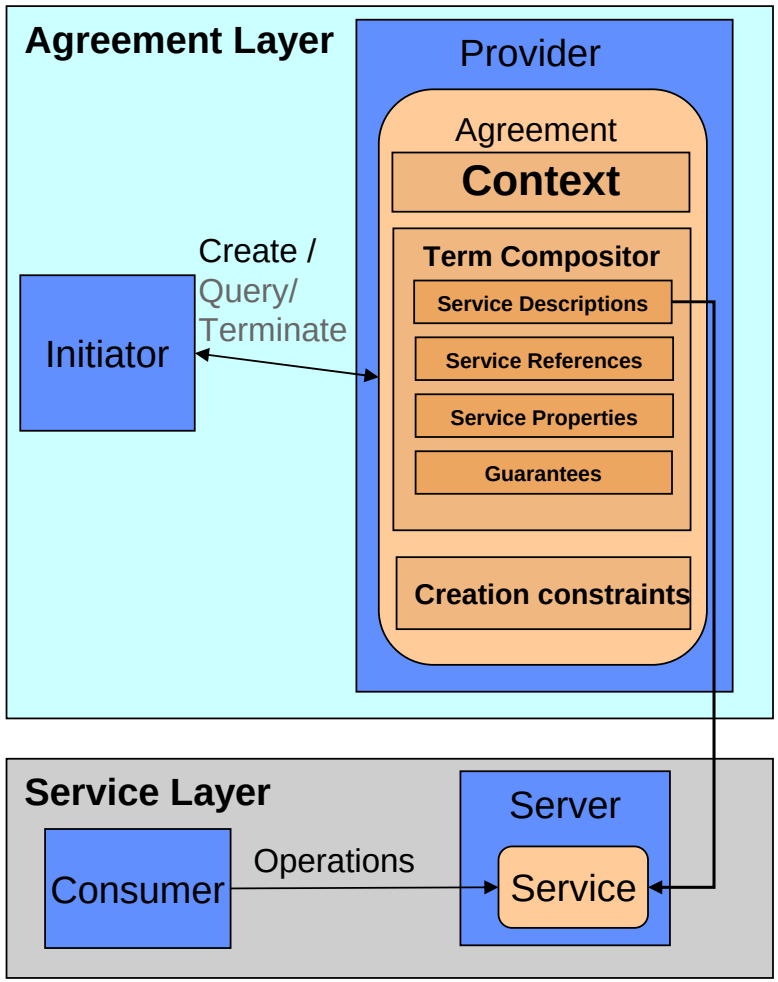

UNICORE Client
File Job Preparation Job Monitoring Settings Extensions Help
Job Preparation
MyMetajob
metaMPI_juelich
metaMPI_aachen
metaMPI_bonn
Job Monitoring
CGX OMG
GATEWAY
Juelich OMG
Juelich Test
Tartu OMG
Ulster OMG
VIOLA_FH-BONN
VIOLA_FZJ
Name MyMetajob
Job Scheduling
Date: 12.05.05
Start time: 12:00
Duration of the job in hours: 1
Submit resource request
Resource Requirements
UNICORE Sites
CGX OMG
GATEWAY
Juelich OMG
Tartu OMG
VSite_1 <NJS>
Ulster OMG
VIOLA_FH-BONN
VIOLA_FZJ
zampano <NJS>
Information about the target system
Information
NJS information: Tue May 10 15:52:19 EEST 2005
NJS information: 4.0.2_build2
Execution System: NJS Linux (testing)
Architecture: SMP
Number of CPUs: 2.0
Capacities
Name | Description | Minim... | Maximum | Default | Units
Node | Number of Nodes | 1.0 | 2.0 | 1.0 | Nodes
Proces... | Number of PEs per No... | 1.0 | 2.0 | 1.0 | Processors per no...
Memory | Total Amount of Memory | 1.0 | 2056.0 | 256.0 | Megabytes per no...
RunTime | Time per Job | 10.0 | 25920... | 600.0 | Seconds
Storages
Name | Description | Minimum | Maximum | Default | Units
Home | Space in $HOME | 0.0 | 100.0 | 10.0 | MegaBytes
Alternative Job Directory | Job Uspace | 0.0 | 1000.0 | 10.0 | MegaBytes
Job Directory | Job Uspace | 0.0 | 1000.0 | 10.0 | MegaBytes
Spool | Spool Space | 0.0 | 100.0 | 10.0 | MegaBytes
Root | Absoulte paths | 0.0 | 100.0 | 10.0 | MegaBytes
Archive | Archive space | 0.0 | 1000.0 | 100.0 | MegaBytes
Performance
Vsite(s) | Nodes | VSite_1 | zampano | DEMO_NJS
VSite_1 <NJS> | 1 | - | 1 : 1 | 1 : 1
zampano <NJS> | 1 | 1 : 1 | - | 1 : 1
DEMO_NJS <NJS> | 1 | 1 : 1 | 1 : router : bandwidth | -
Add Vsite
UNKNOWN MyMetajob not saved yet.
8.89Mb/12.71Mb

MetaScheduler - Integration in UNICORE



- UNICORE Client sends request to MetaScheduler (WS-Agreement)
- MetaScheduler negotiates earliest time to run this job, requests the reservation of the requested resources and returns the WS-Agreement with additional Status, ID
- UNICORE Client creates Abstract Job Object (AJO) and sends it to the Primary Network Job Supervisor (NJS)
- NJS incarnates the AJO according to the information in the AJO and the UIDB, forwards it to the local Target System Interface (TSI) and sends the AJO to all other NJSs
- TSI creates the entry for the Meta-Job in the UNICORE Job Queue, and stores the job data in the User-directory
- Scheduler triggers job at start time

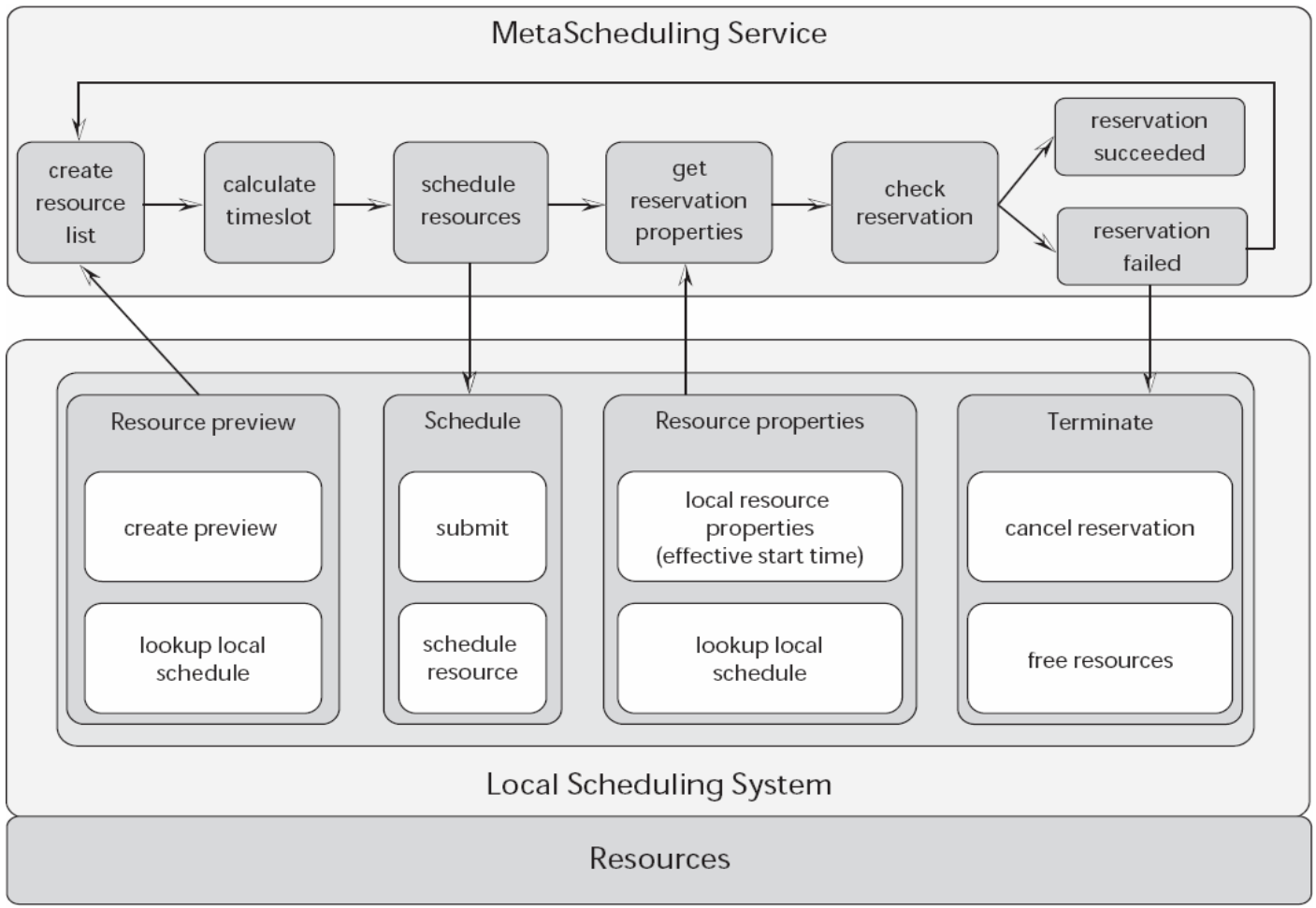
WS-Agreement Structure



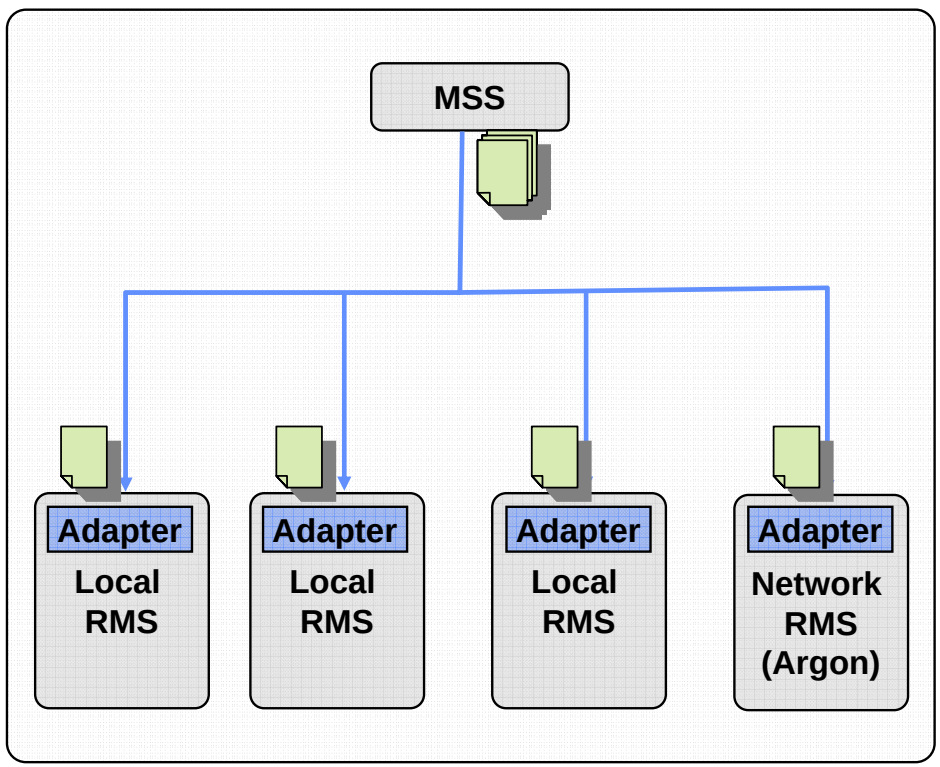
WS-Agreement Main points

- Protocol for dynamic agreement management
- Terms can relate to:
  - functional description
  - non-functional properties
- WS-Agreement is domain-agnostic
- An agreement can involve four conceptual parties:
  - agreement initiator and provider and
  - service consumer and provider
- State can be published and monitored via agreement properties allowing notification of updates
- Can be chained or nested to represent complex relationships

MSS Allocation Protocol

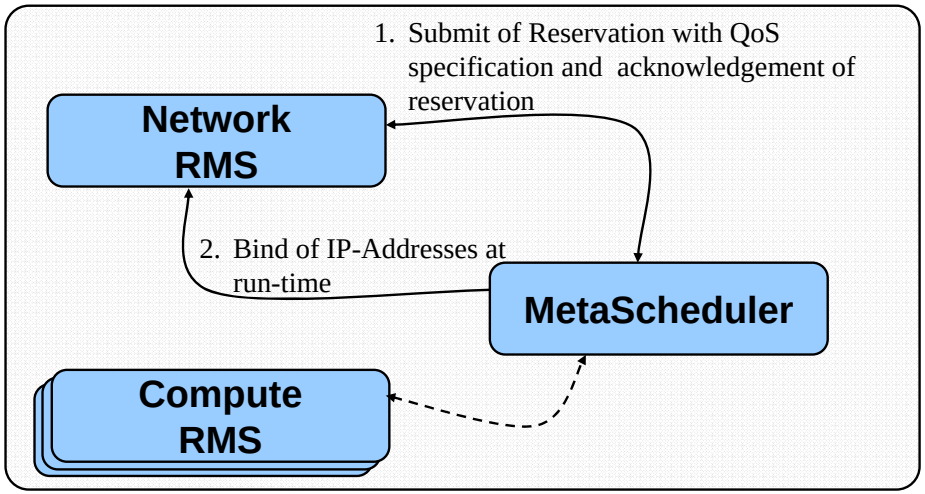


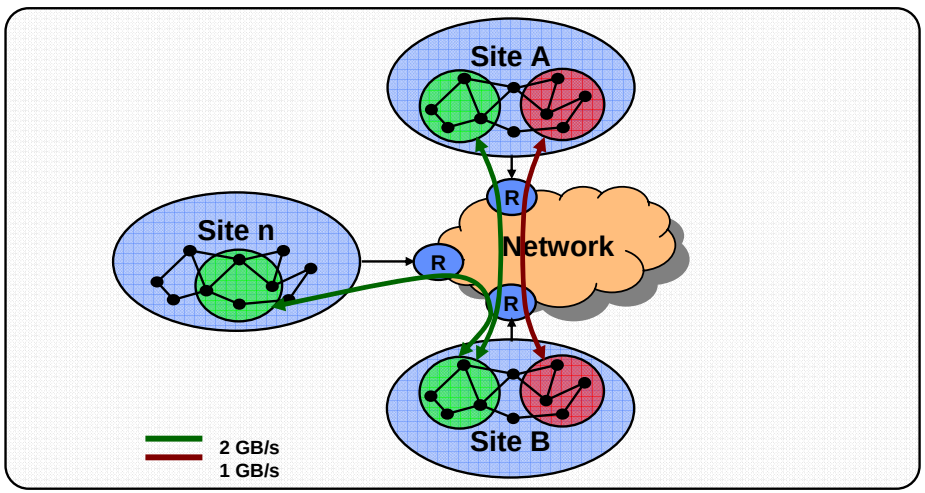
Collecting runtime information



- After job submission the MSS monitors the state of all reservations
- When all reservations changed to ***active*** the MSS gathers the IP addresses of finally assigned nodes
- This information is aggregated by the MSS
- The aggregated runtime information is sent to the subsystems
- Compute RMS use this information to build the MetaMPICH configuration and start the application
- The network RMS completes the network configuration based on this information

Allocate and Configure the Network Resources





1.) Reservation of required Resources

- Submit of a Reservation to the Network Resource Manager
- Acknowledgement of Reservation

2.) Bind of IP-Addresses at Run-time

- IP-Addresses are published at run-time of the job through the local Adapter
- Bind of the IP-Addresses by the Network Resource Manager
- Without explicit Bind the QoS Parameters for the Site-to-Site Interconnection are used

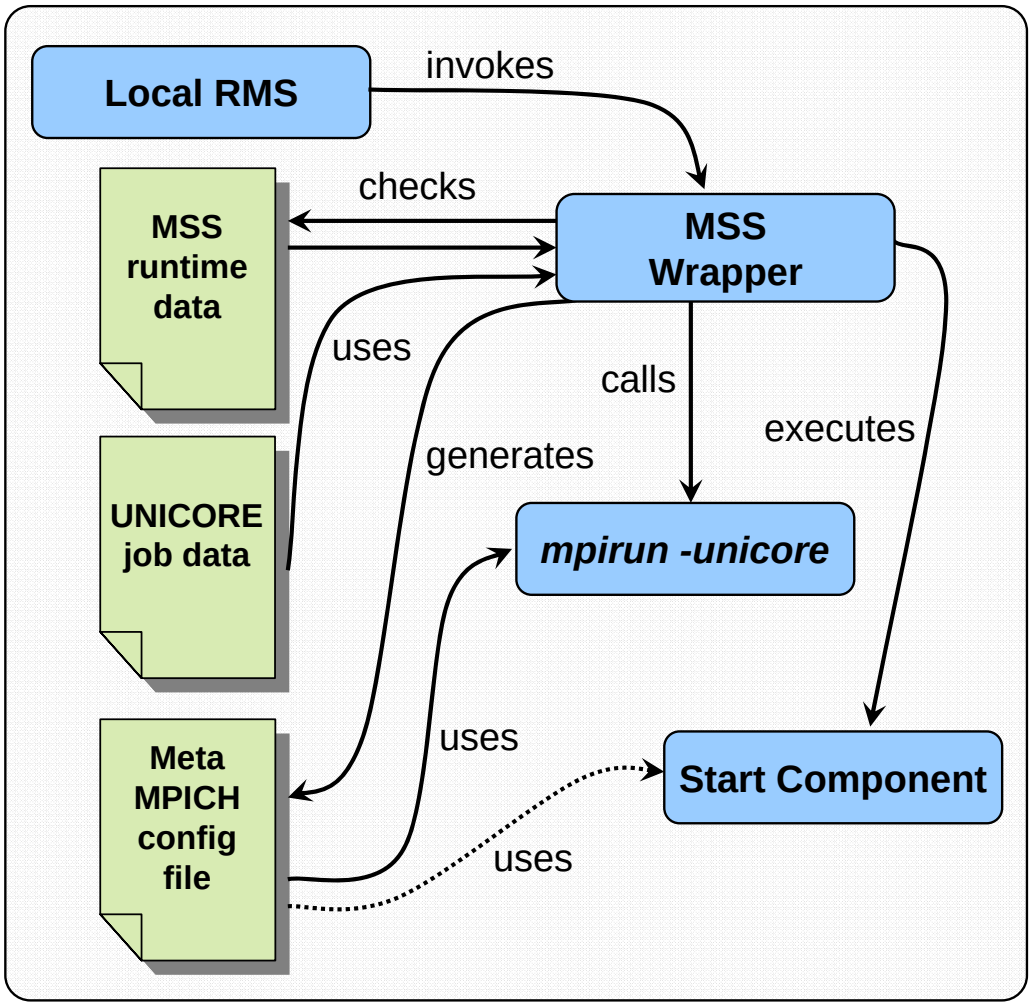
Fraunhofer SCAI Institut Algorithmen und Wissenschaftliches Rechnen

Forschungszentrum Jülich in der Helmholtz-Gemeinschaft

CoreGRID

VIOLA

Application Startup



- At the time a reservation becomes active MSS wrapper script is called
- The wrapper checks periodically if the MSS runtime configuration file is present
- When the MSS runtime was found the MetaMPICH configuration file is generated
- Afterwards it calls the mpirun command with the UNICORE flag (-unicore) using the generated configuration file
- This prints out the ssh calls to start up the MetaMPICH components
- The wrapper only executes the ssh call for the local site in order to start the application component at the local site

MetaMPICH sample configuration

```
NUMHOSTS 2          # MSS information
SCAI_PACK 2         # MSS information
CAESAR 2            # MSS information

OPTIONS
SECONDARY_DEVICE ch_usock (PORTBASE=35021, NETMASK=194.94.198.0/24)

METAHOST SCAI_PACK {
   TYPE=ch_usock;                                  # UNICORE job data
   FRONTEND=packcs-g0.viola-testbed.de;            # MSS information
   MPIROOT=/usr/local/viola/mp-mpich-md;           # UNICORE job data
   EXECPATH=/home/oliver;                          # UNICORE job data
   NODES=pack00.viola-testbed.de 1 (194.94.198.200),pack01.viola-testbed.de 1 (194.94.198.201);      # MSS info
}

METAHOST CAESAR {
   TYPE=ch_usock;
   FRONTEND=pcc3001.viola-testbed.de;
   MPIROOT=/usr/local/viola/mp-mpich-md;
   EXECPATH=/home/waeldrich;
   NODES=pcc3010.viola-testbed.de 1 (194.94.198.10),pcc3011.viola-testbed.de 1 (194.94.198.11);
}

CONNECTIONS
PAIR SCAI_PACK CAESAR 0 -
PAIR CAESAR SCAI_PACK 0 -
```

Future Work

- Integration in GT4
- Integration in UNICORE/GS
- Co-allocation of additional resource types (e.g. licenses)
- Implementation of the full WS-Agreement
- Resource negotiation using on WS-Agreement
- Interoperability with other brokers
- Automatic resource selection based on application monitoring data (VIOLA MSS/ISS integration)