Managed by

Meta-Brokering Solutions for Expanding Grid Middleware Limitations

A. Kertész[1] , I. Rodero[2] and F. Guim[2]

[1]MTA SZTAKI

attila.kertesz@sztaki.hu

[2]Barcelona Supercomputing Center

{irodero, francesc.guim}@bsc.es

Contents

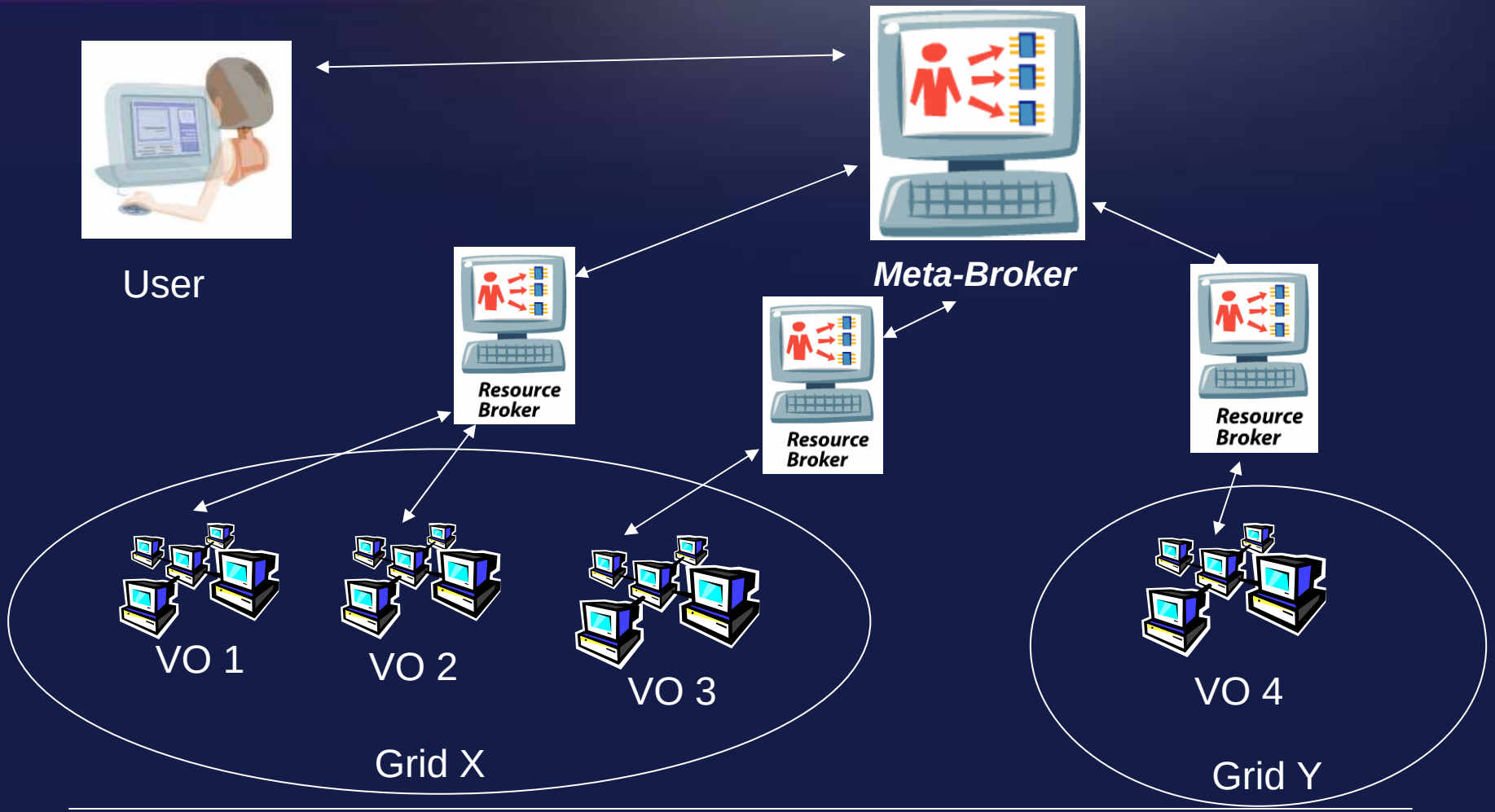
- Introduction
- General Meta-Broker Architecture
- Meta-Broker realizations
- Evaluation
- Future directions
- Conclusions

Introduction

- Grid resource management is probably the research field most affected by user demands
- Though well-designed, evaluated and widely used resource brokers, meta-schedulers have been developed, new capabilities are still required
- We examined and compared different research directions followed by researchers regarding meta-level brokering approaches
- Our proposed meta-brokering approach means a higher level resource management by enabling communication among existing Grid Brokers and utilizing them

Current situation in Grids

Meta-brokering in Grids

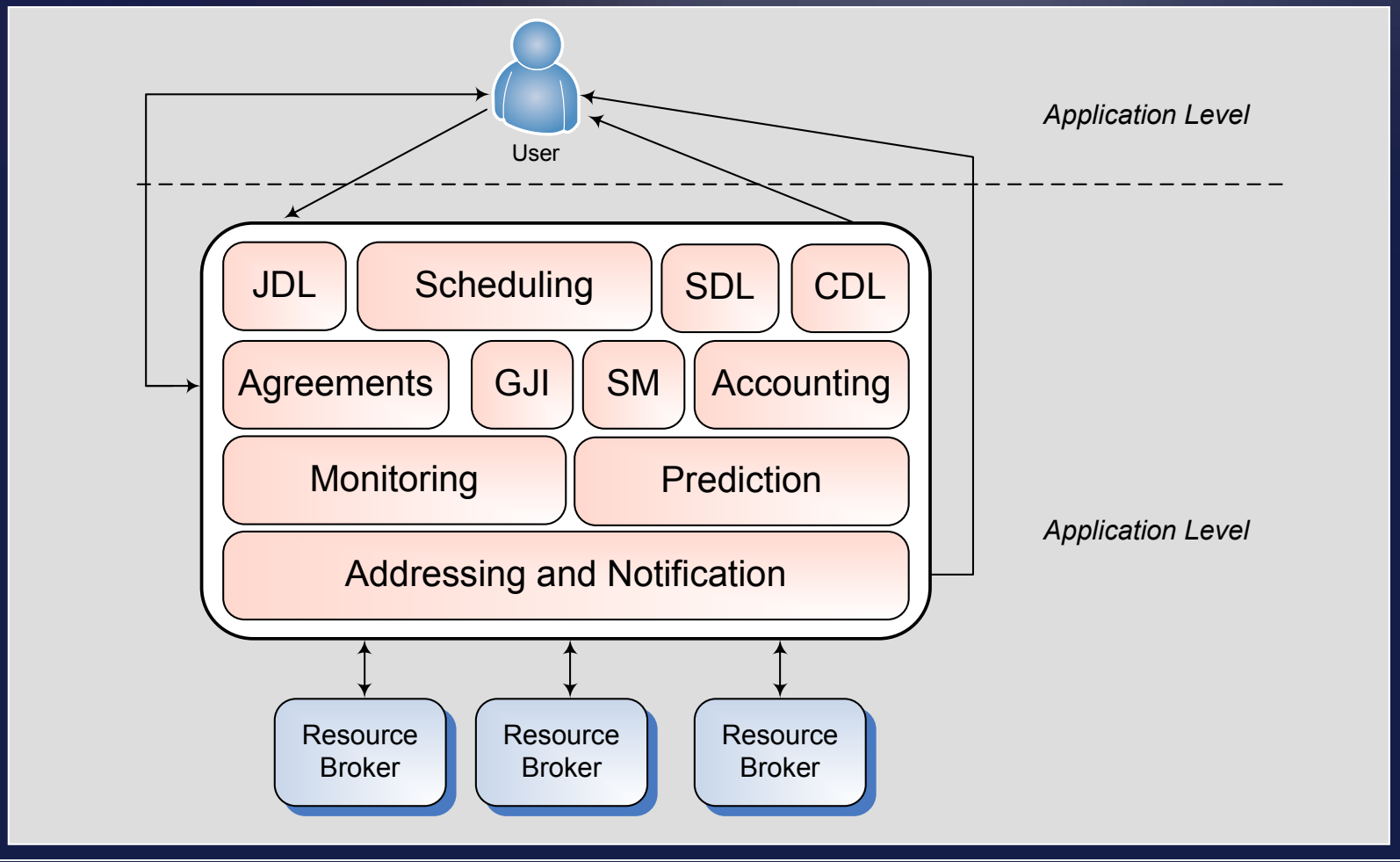


Related work

Meta-brokering by inter-broker communication:

- **GSA of OGF**
- **LA grid broker instances in domains – aggregated resource information share**
- **Koala – DMM for saturated grid domains**
- **Gridway instances communicating through GridGateways**

General Meta-Broker Architecture



General Meta-Broker Architecture

JDL - Job Description Language: a unified description to specify user requirements

CDL - Capability Description Language: a unified description to specify broker functionalities, service capabilities

SDL - Scheduling Description Language: Besides CDL and JDL the scheduling requirements and policies also need to be stored

Scheduling: broker selection with the help of meta-data

GJI - Global Job Identifiers: unique job identification within the meta-brokering scope

General Meta-Broker Architecture

SM - Security Management: to secure access to the interconnected domains

Accounting Mechanism: user access by pre-defined policies

Agreements Mechanism: to negotiate user requirements, which can also affect scheduling policies

Monitoring Mechanism: reliable operation by global monitoring for self-aware and fault tolerant operation

Prediction Mechanism: to perform calculations of broker availability, service utilization and user request loads to cope with the highly dynamic nature of grids

Addressing and Notification Mechanism: to access the inter-connected resource brokers, and managing communication including local events and external job state notifications

Meta-Broker realizations

We used standards and widespread technologies, where applicable. (JSDL, WSDL)

Regarding CDL we previously developed a language called BPDL (Broker Property Description Language), which had also incorporated SDL:

- revised and modified BPDL
- gathered the scheduling-related attributes to MBSDL (Meta-Broker Scheduling Description Language)
- this separation contributes to the standardization process of the OGF-GSA research group to create a standard SDL

Implemented capability language

BPDL:

- BrokerID: unique identifier of a broker
- Interface: metadata about the accessibility and notification methods of the broker
- Monitoring: for specifying self, job or resource monitoring mechanisms
- Security: job and user authentication methods (MyProxy)
- PerformanceMetrics: Dynamic information about availability and reliability

MBSDL:

- Constraints: middleware, remote file access and job type constraints, processing time and budget cost requirements
- QoS (Quality of Service) requirements: agreements, job priorities, advance reservations, email notification or access controls, fault tolerance features
- Policy: customized scheduling policies, local scheduler capabilities

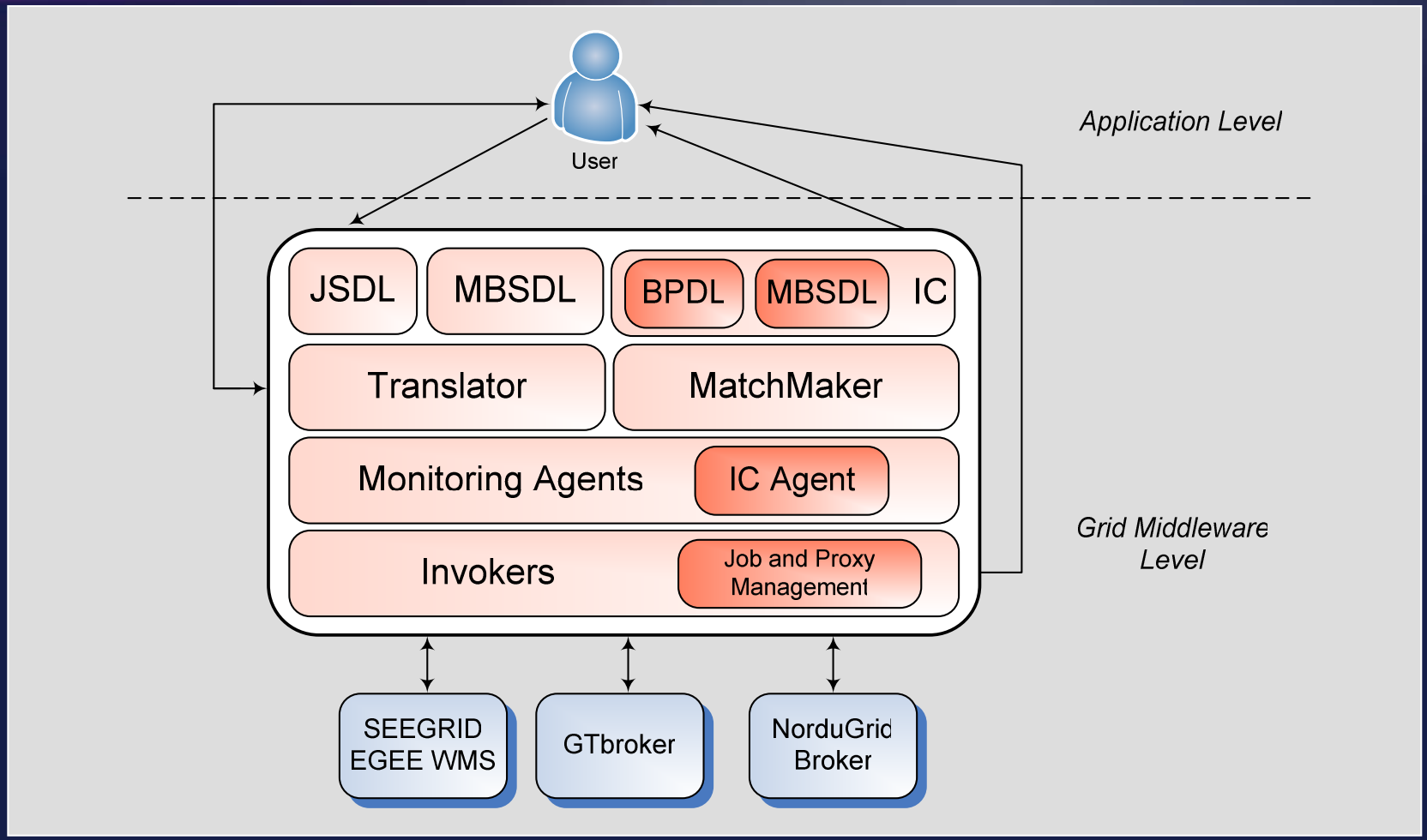
Scheduling policies

A general meta-brokering policy relies on the capabilities and measured performances of the utilized brokers

The selection of the appropriate brokering system can be done using a multi-criteria algorithm that can take into account the gathered and predicted metadata stored in SDL and CDL

In our realizations the scheduling policy examines the job description, which is stored in JSDL and MBSDL, and matches them with the metadata stored in BPDL and MBSDL

A.) Grid Meta-Broker in the P-GRADE Portal



Grid Meta-Broker in the P-GRADE Portal

It has been designed as a standalone, standards-base Web Service, therefore it is grid middleware and portal independent

The Translator component:

- translates the JSDL of the user job to the language of the appropriate broker/grid JDL

The Information Collector component:

- stores the data of the reachable brokers and historical data of the previous submissions in BPDL, the load of the resources behind the brokers is also stored here

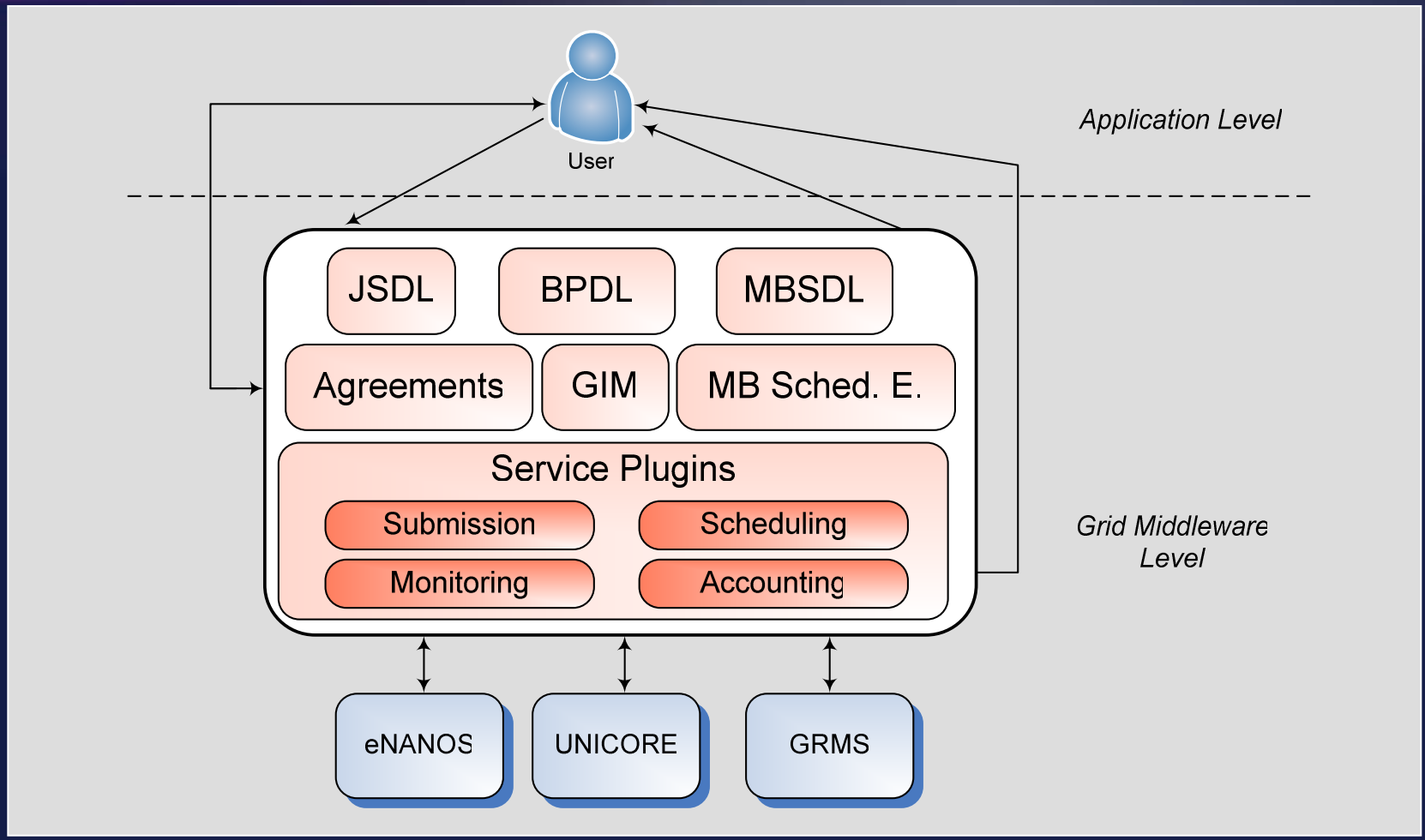
IS Agents:

- report to the IC, they monitor the load of the underlying grids of each connected resource broker

The Invokers:

- broker-specific components: they communicate with the interconnected brokers, invoking them with job requests and collecting the results

B.) Meta-Broker in the HPC-Europa SPA



Meta-Broker in the HPC-Europa SPA

This extended version includes new services to perform meta-brokering:

- new module for the broker scheduling: a scheduling plug-in for each center,
- a language to describe the scheduling capability of brokers (MBSDL)
- GIM: global identifiers management (managing jobs coming from all the centers)
- some other services, such as a predictor service or a historic data catalog to improve the scheduling techniques.

The meta-brokering scheduling engine (MB Sched. E.) is responsible for the broker selection:

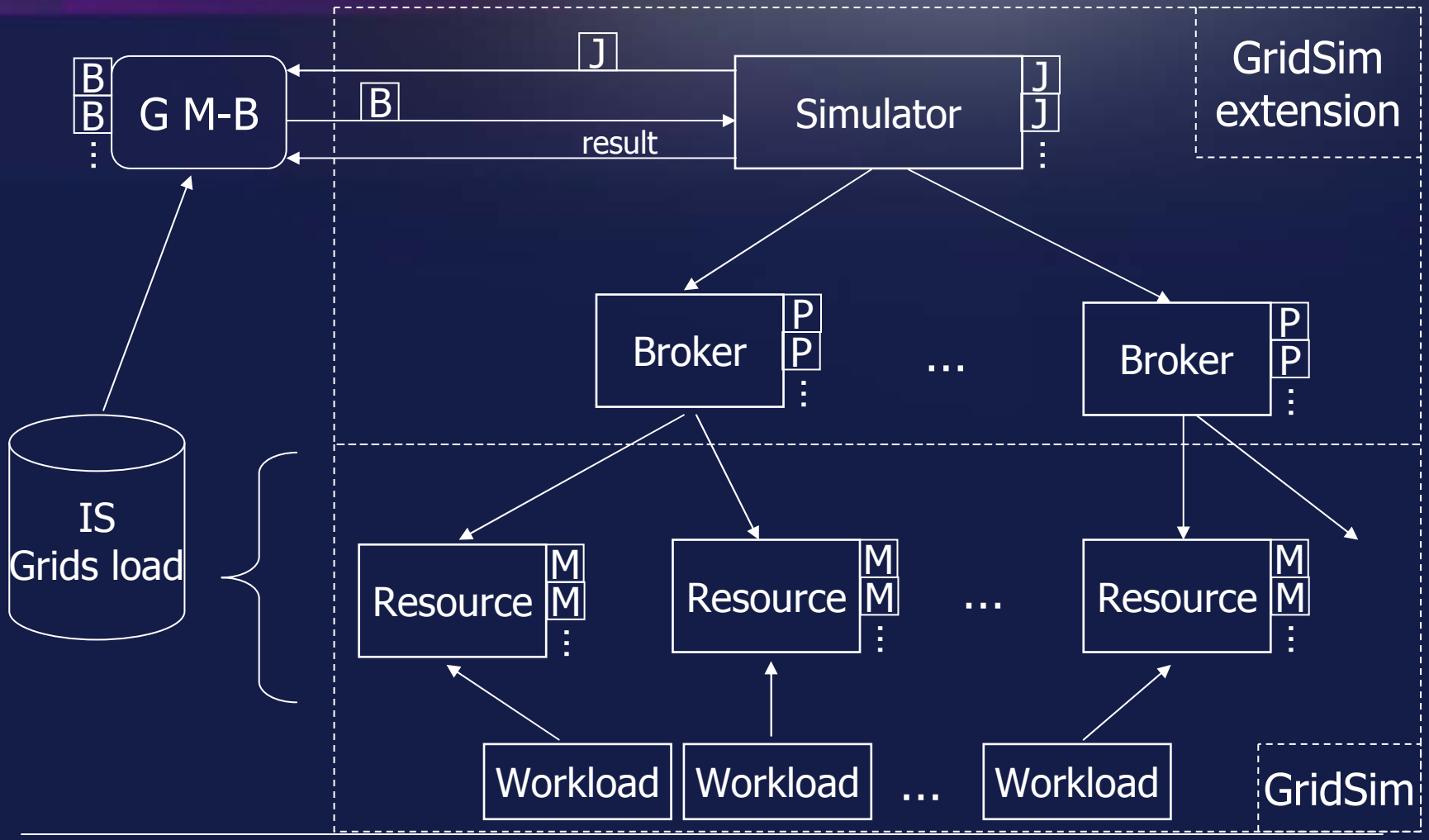
- at the portal level, it evaluates one of the meta-brokering policies available in the framework

Different brokers need to support these new functionalities via their plug-ins

A.) Evaluation with GridSim

- The GridSim toolkit is a fully extendable, widely used and accepted grid simulation tool
- It can be used for evaluating VO-based resource allocation and dynamic resource provisioning techniques in global grids
- It supports modeling and simulation of heterogeneous grid resources, users, applications, brokers and schedulers in a grid computing environment
- It provides primitives for the creation of jobs, mapping of them to resources, and managing the whole job life-cycle

Grid Meta-Broker with GridSim



Evaluation results

Brokers	Resources	Jobs	Work-load	AVG time Rnd	AVG time MB
3/X-3/Y (rnd)	6X4(X4)	50	10X(6X4)	572666.30	67327.74 **11.8 %**
3/X-3/Y (fcpu)	6X4(X4)	50	10X(6X4)	145002.44	33117.03 **22.8 %**
3/X-3/Y (rnd)	6X4(X4)	100	20X(6X4)	1086140.78	93959.86 **8.7 %**
3/X-3/Y (fcpu)	6X4(X4)	100	20X(6X4)	255944.12	18469.28 **7.2 %**

- 10 times shorter makespan

Evaluation results

Brokers	Resources	Jobs	Work-load	AVG time Rnd	AVG time MB
8/X (fcpu)	8X4(X4)	50	10X(8X4)	125167.43	65427.31 – **52%**
8/X (fcpu)	8X4(X4)	50	10X(8X4)	236322.30	117563.16 – **49%** 14318.02 - **6%**
8/X (rnd)	8X4(X4)	100	20X(8X4)	1320141.79	226344.34 - **17%** 22304.95 - **2%** 7971.54 - **1%**

- Less execution time
- Much shorter makespan, after trained values

B.) Evaluation with Alvio

C++ event driven simulator

Implements scheduling policies at different scheduling layers:

- Local HPC scenarios
- Grid systems (various local HPC systems)
- Meta-brokering scenarios (various Grid systems)

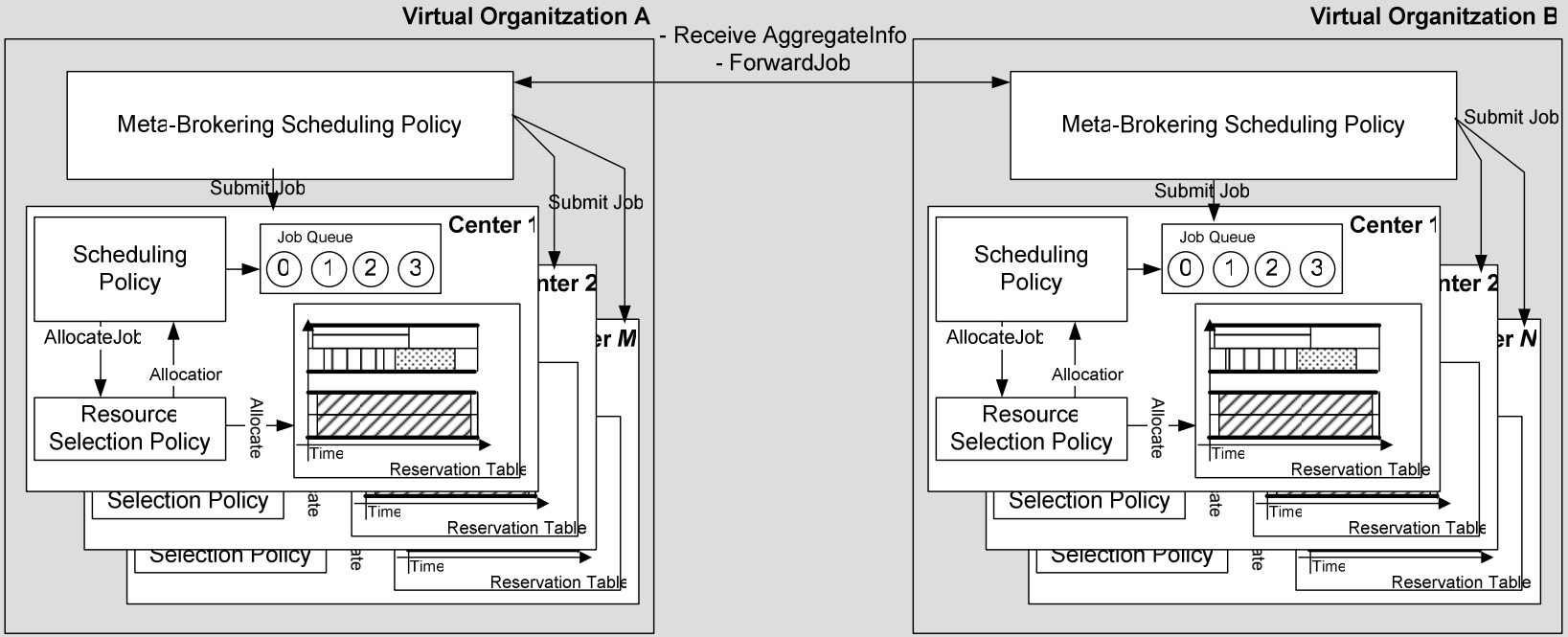
We use real systems traces from 'Grid Workload Archive'

- DAS-2
- Grid5000
- Sharcnet

Meta-brokering in a P2P fashion

Job forwarding between brokers (VO's) using also aggregated resource data for broker selection

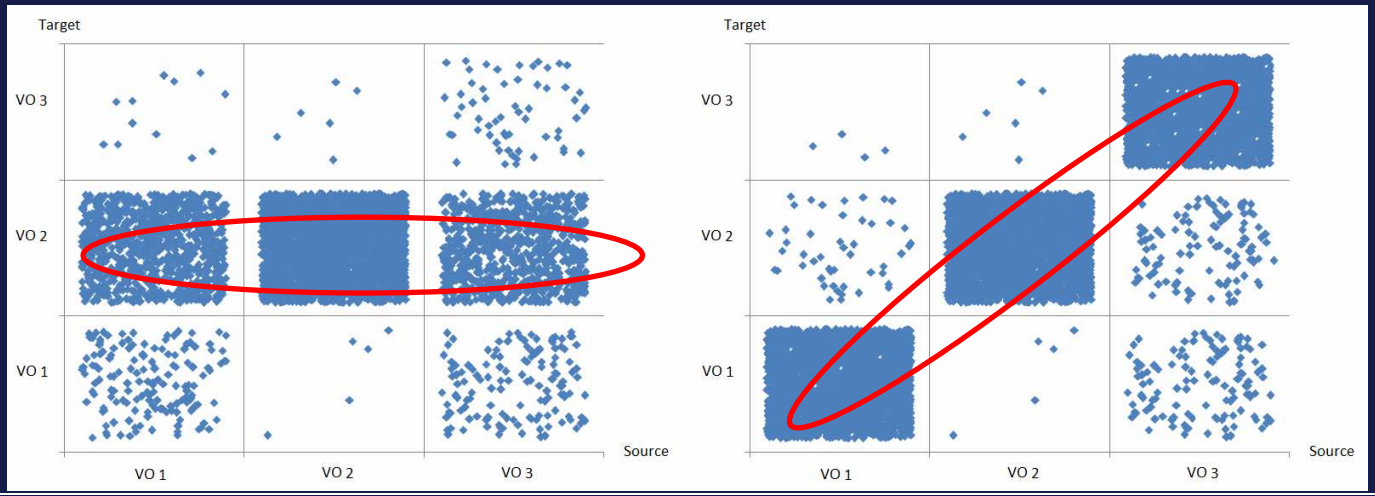
Meta-brokering with Alvio



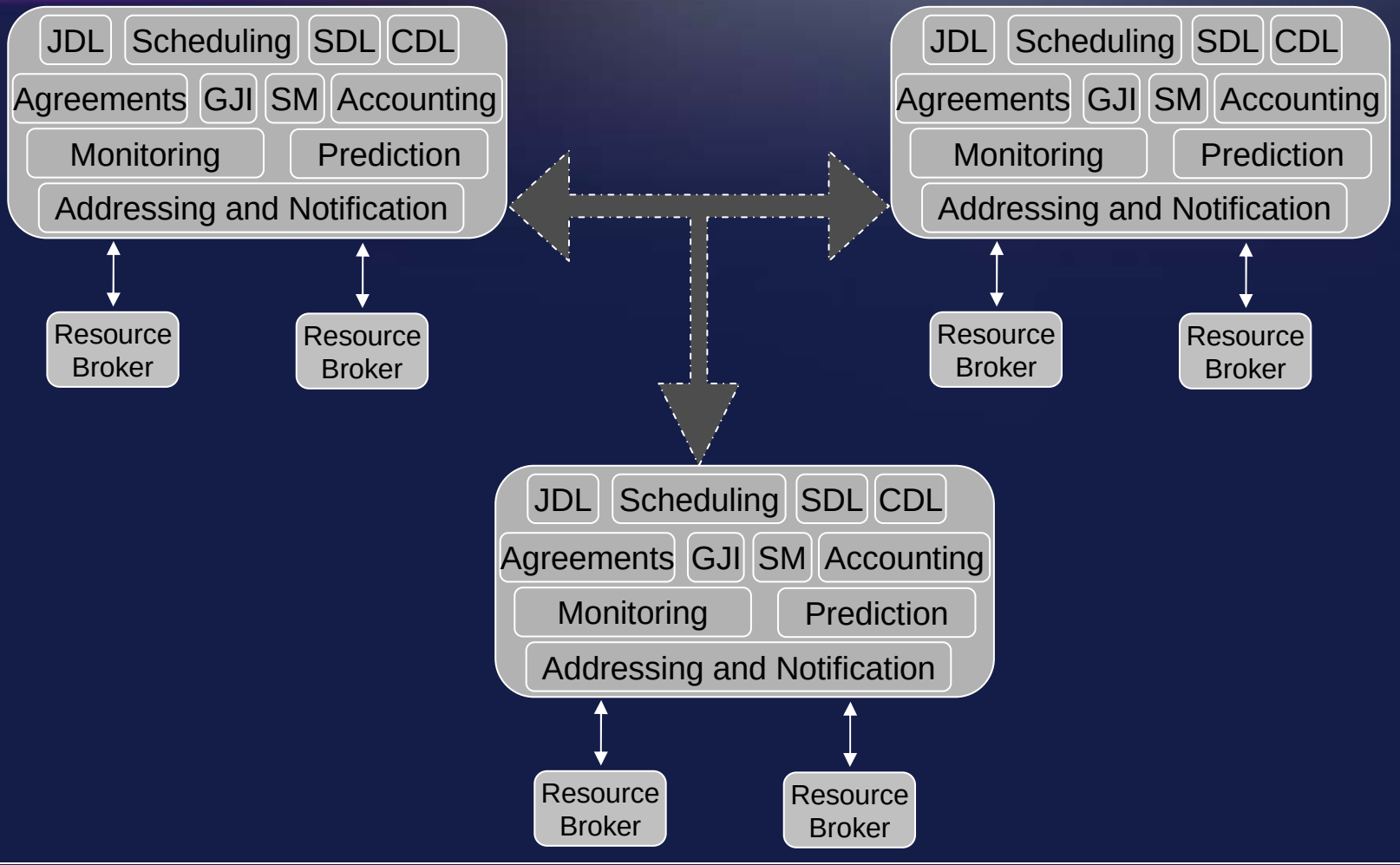
Evaluation with Alvio

Workload	Exec Time (h)	Avg Wait (s)	Avg SLD	CPU Util
DAS-2	1,486	1,789	11,26	28 (8,33%)
Grid5000	1,565	4,850	13,06	39 (6,79%)
Sharcnet	1,596	3,011	15,78	200 (6,25%)
Overall	1,596	3,216	13,36	267 (7,12%)
Meta-brokering	1,410	274	1,34	314 (7,63%)

Job Forwarding (with 2 different policies)



Future directions: distributed operation



Conclusions

- We have examined and compared different research directions followed by researchers in the field of Grid Resource Management regarding higher level brokering approaches
- We defined the essential requirements of a novel middleware tool called Meta-Broker, proposed a general architecture and have shown how it can be implemented in two different grid portal environments
- We derived two language schemas for describing resource brokers and scheduling policies, using meta-information about resource brokers and user requests we proposed a general matchmaking algorithm for meta-brokering scheduling policies
- After finalizing the presented two meta-brokering solutions, we have carried out performance measurements in two different portal environments
- **The evaluations show better resource utilization and shorter execution times**

Information Society Technologies

CoreGRID

Thank You!

Questions, remarks, suggestions?