

## **BPDL: A Data Model for Grid Resource Broker Capabilities**

*A. Kertész*

`attila.kertesz@sztaki.hu`

*MTA SZTAKI Computer and Automation Research Institute  
H-1518 Budapest, P.O. Box 63, Hungary*

*I. Rodero, F. Guim*

`{irodero, francesc.guim}@bsc.es`

*Barcelona Supercomputing Center  
Jordi Girona 29, 08034 Barcelona, Spain*



CoreGRID Technical Report  
Number TR-0074  
March 14, 2007

Institute on Resource Management and Scheduling

CoreGRID - Network of Excellence  
URL: <http://www.coregrid.net>

# BPDL: A Data Model for Grid Resource Broker Capabilities

A. Kertész

`attila.kertesz@sztaki.hu`

MTA SZTAKI Computer and Automation Research Institute  
H-1518 Budapest, P.O. Box 63, Hungary

I. Rodero, F. Guim

`{irodero, francesc.guim}@bsc.es`

Barcelona Supercomputing Center  
Jordi Girona 29, 08034 Barcelona, Spain

*CoreGRID TR-0074*

March 14, 2007

## Abstract

Since the management and the optimal utilization of the highly dynamic grid resources cannot be handled by the users themselves, various grid Resource Brokers have been developed, supporting different grids. To ease interoperability and the higher level utilization of different resource brokers, we introduce a meta-data model for storing broker capabilities and show how an implementation of this model can be realized. We believe that this abstraction will help standardizing inter-broker communication to enable more efficient grid resource utilization.

## 1 Introduction

The Grid was originally proposed as a global computational infrastructure to solve grand-challenge, computational intensive problems that cannot be handled within reasonable time even with state of the art supercomputers and computer clusters [1]. Grid computing tackles these tasks by aggregating geographically and architecturally dispersed hardware and software resources into large virtual super-resources. The first decade of grid research aimed at creating relatively reliable infrastructures to serve researchers and attract users. These attempts have led to the present grid middlewares, and now development is focusing on user requirements. End-users typically access grid resources through resource management systems. Unfortunately, these tools are typically tightly coupled to one specific grid environment and do not provide multi-grid support because each center has its own resource management and scheduling system. Even if a tool is connected to multiple grids, applications that utilize services from these grids simultaneously are rarely supported.

There have been several attempts to make existing production Grids and grid services interoperable. Grid researchers seem to follow two different ways in the area of resource management.

The first one is to extend existing resource brokers with multiple grid middleware support. The Gridbus Grid Service Broker [2], Gridway [3], JSS [4], GRMS [5], GTbroker [6] and eNANOS [7], all support accessing resources of different middlewares. The GRIP [8] broker is the one that tries to support interoperability with a semantic matching of the resource descriptions enabling job submissions to Globus [9],[10] and Unicore [11] sites. This summary shows that these tools are forming separate user groups, again. They use different job descriptions and do not communicate with each other: putting an end to this separation process would need high efforts by all parties.

---

This research work is carried out under the FP6 Network of Excellence CoreGRID funded by the European Commission (Contract IST-2002-004265).

The second approach is to provide a higher level tool that supports different middleware services. One possible instance of this approach is a meta-scheduler, which coordinates some communication process between existing schedulers. A part of the SPA in the HPC-Europa Project [12] was working on a similar topic. In this approach, each center implements a plug-in with its own set of supported capabilities. The user chooses manually the system to submit their jobs and the job scheduling policies are evaluated inside the context of each center. Thus, it does not take into account the broker scheduling properties; it rather operates as a job submitter.

The OGSA-RSS-WG of the OGF [13] devoted efforts to provide protocols and interface definitions for the resource selection services portion of the Execution Management Services (EMS) part of the Open Grid Services Architecture. The Resource Selection Services (RSS) consist of the Candidate Set Generator (CSG) and the Execution Planning System (EPS). The CSG can be used to generate a set of computational resources that are able to run a job in general, while the EPS uses this list to decide exactly what resources to run the job.

The GSA-RG of OGF [14] is currently working on a project enabling grid scheduler interaction. They try to define a common interface among schedulers enhancing interoperability. The greatest problem is that the existing schedulers/brokers need to support this common interface, so they need to be modified. In the following sections of this paper we introduce a meta-data model for describing various resource brokers to enable communication and interoperability among them. Furthermore, we formalize the proposed meta-data model and we specify a XML schema as a possible implementation for the model.

## 2 A Meta-Brokering Approach and Description Languages

Utilizing the existing, widely used and reliable resource brokers and managing interoperability among them could be new point of view in resource management. Users usually have certificates to access more VOs. A new problem arises in this situation: which VO, which broker to choose for my specific application? Just like users needed Resource Brokers to choose proper resources within a VO, now they need a meta-brokering service to decide, which broker (or VO) is the best for them and also to hide the differences of utilizing them. Therefore, a meta-broker can be defined as the middleware component that selects the most appropriate meta-scheduler/broker to submit a job following a particular policy. In this context, a part of the interoperability mechanisms, a new area on scheduling policies is opened to the research (e.g. global load-balancing).

Heterogeneity appeared not only in the fabric layer of Grids, but also in the middleware. Even components and services of the same middleware may support different ways for accessing them. After a point this variety makes the users and developers life miserable. Languages are one of the most important factors of communication. Different resource management systems use different resource specification languages like RSL [9], JDL [15],[16], etc. These documents need to be written by the users to specify all kinds of job-related requirements and data. The OGF has already started to take several steps towards interoperability among these coordinating components, and developed a resource specification languages standard called JSDL [17]. As the JSDL is general enough to describe jobs of different grids, this tool solves the interoperability problem regarding job descriptions. Besides describing user jobs, we also need to describe resource brokers in order to make difference among them and help the Meta-broker decide, to which broker should be the job submitted. These brokers have various features for supporting different user needs and to implement scheduling policies. These needs should be able to be expressed in the users' JSDL, and identified by the Meta-Broker for each corresponding broker. Therefore we propose a Data Model for describing Resource Broker capabilities, to store metadata about brokers. We use a broker taxonomy [18] to identify the relevant properties, where various, widely used grid brokers are gathered and analyzed. These two kinds of languages are used by the Meta-Broker to communicate with the inner and outer world (Figure 1).

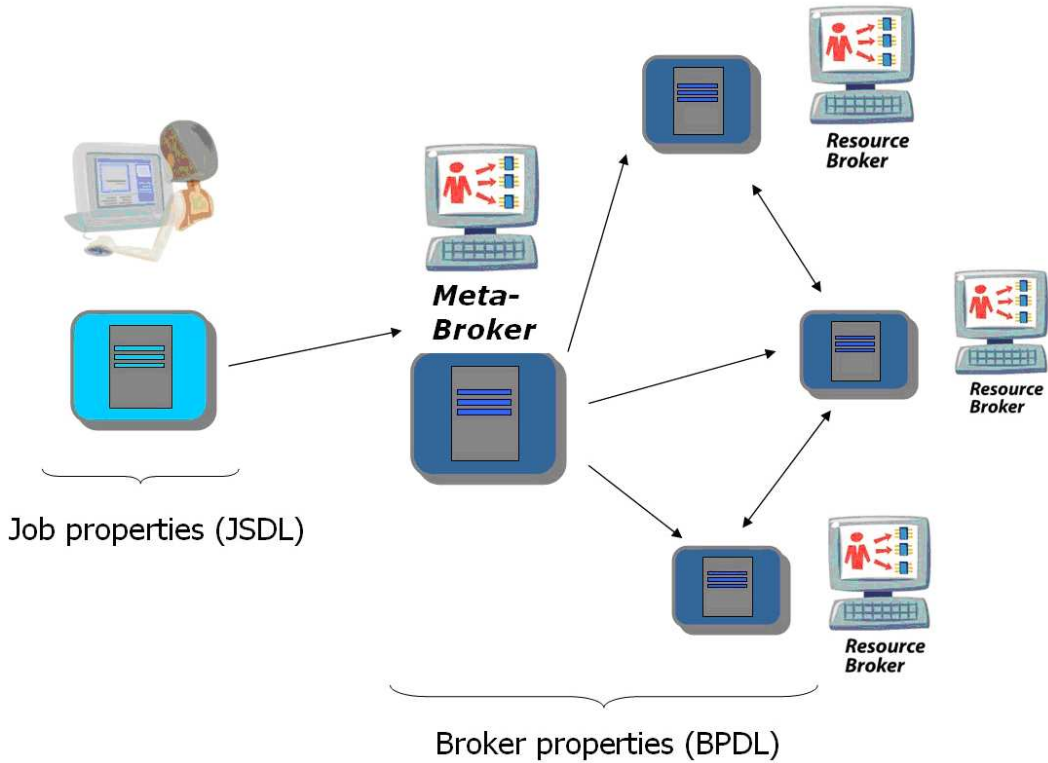


Figure 1: Languages of the Meta-Broker

### 3 Data Model for Broker Capabilities

#### 3.1 Formal definition of the data model

For describing Grid Resource Broker capabilities, we introduce an extensible meta-data model. Our model can be taken as an extension of the general scheduling model presented in [19]. Beside their resource and job model, there is a need for a model describing broker characteristics in order to compare, interoperate and manage different resource brokers, schedulers. We use the same notations for building up the model.

The meta-data about resource brokers are expressed through  $\langle \text{attribute}, \text{value} \rangle$  pairs – we denote with  $\mathcal{P}$  the set of all possible such pairs. A broker denoted by  $\mathcal{B} \subseteq \mathcal{P}$  is modeled as a pair:

$$\langle \text{brokerID}, \text{description} \rangle,$$

where  $\text{brokerID}$  is a unique identifier, and  $\text{description} \subseteq \mathcal{P}$  is a set of attribute/value pairs, which contains metadata of basic and special properties. Figure 2 shows the tree of pairs in  $\mathcal{P}$ , which defines the whole model.

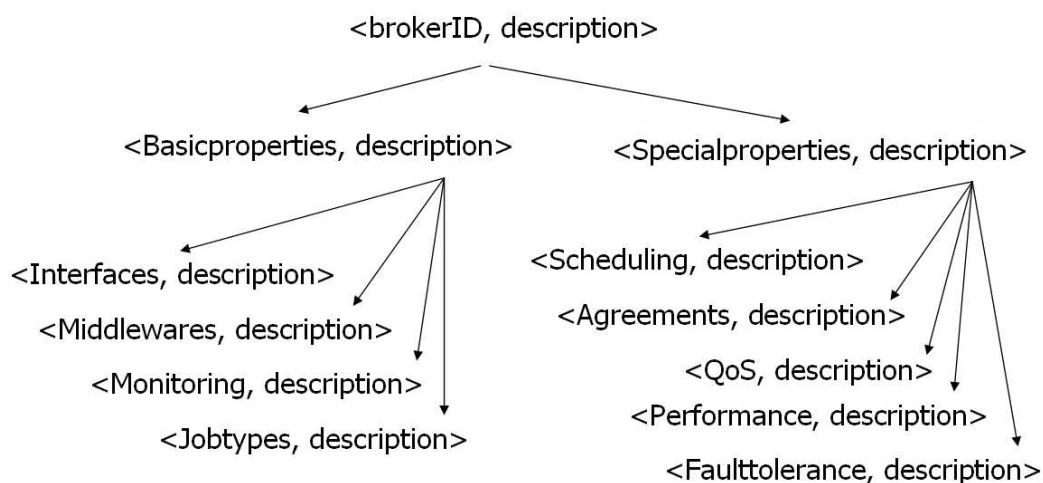


Figure 2: Structure of the Data Model for resource broker capabilities

### 3.2 Matchmaking function

In order to present a usage scenario we define a function over this model with the following structure:

- $\mu: \mathcal{T} \times \mathcal{B}^i \rightarrow \mathcal{B}$ , where  $\mathcal{T}$  is a set of tasks [19] (here: jobs) and  $\mathcal{B}$  is a set of brokers.

For  $t \in \mathcal{T}$ ,  $b_0, \dots, b_n \in \mathcal{B}$ ,  $n \geq 0$ :

- $\mu(t, (b_1, b_2, b_3)) = b_2$  means that for a job denoted by  $t$  matched with brokers denoted by  $b_1$ ,  $b_2$  and  $b_3$  the matchmaking function returns  $b_2$ , which is the fittest broker for the job. That means the returned broker can most efficiently execute the job. (Note that  $b_0$  can be a special element, which is an empty description. This is the return value, when no broker fits the job requirements.)

In our scenario shown in the following section a JSDL of the job is denoted by  $t$ , and a BPDFL of a broker by  $b_i$ .

## 4 BPDFL: One possible implementation of the data model

### 4.1 The Broker Property Description Language

Based on the data model introduced in the previous section we have created an XML-based language called BPDFL (Broker Property Description Language). The common subset of the individual broker properties are the basic properties: the supported middlewares, job types, certificates, interfaces, and monitoring issues. (See the representation of this schema in Figures 3 and 4) And there are special ones, such as remote file handling, fault tolerant features, agreement support, QoS support, performance metrics, and various scheduling policies. The union of these properties forms a complete broker description document that can be filled out and regularly updated for each utilized resource broker. The special any##other type describes a mechanism that can be used to extend the schema with custom elements and attributes.

Notice that these languages can also be used for peer-to-peer communication and identification in a decentralized architecture. In particular, the agreements are another mechanism typically used in this kind of architectures to broaden a domain or as a communication mechanism during the negotiation process.

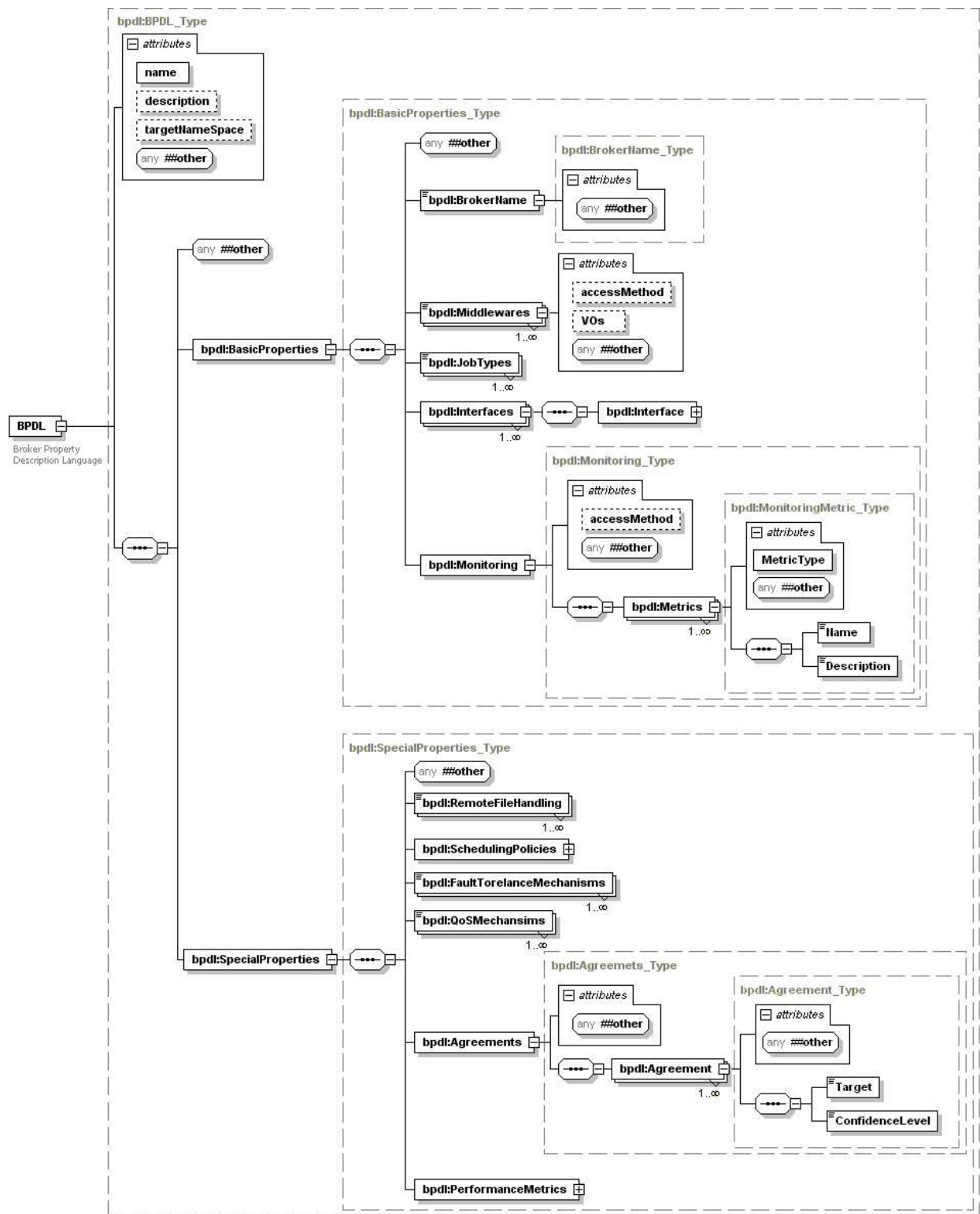


Figure 3: General XML schema of the Broker Property Description Language

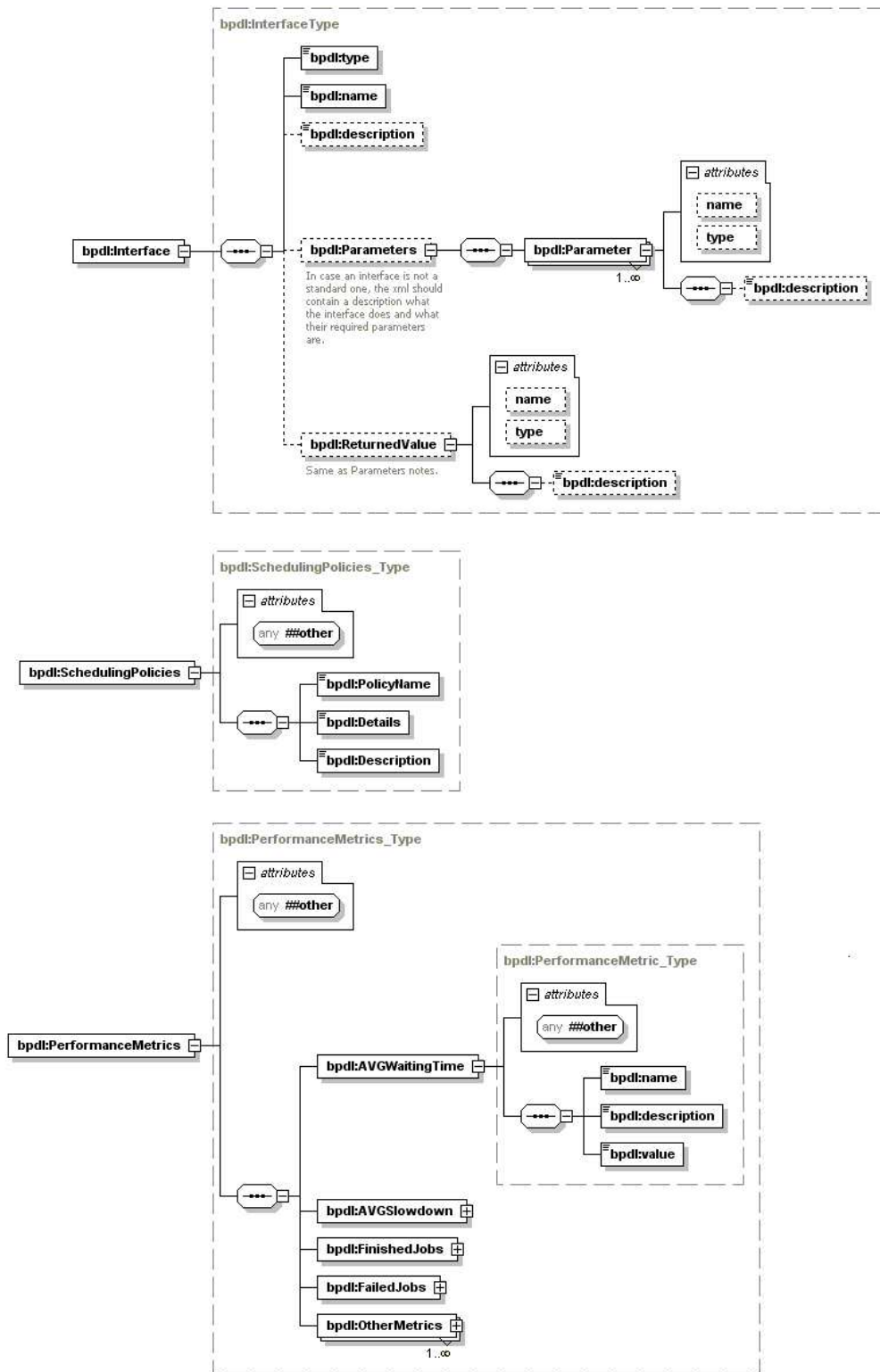


Figure 4: Elements of the BPD XML schema describing interfaces, scheduling policies and performance metrics

## 4.2 Scenario for BPDFL utilization in the Meta-Broker

When some years ago grid developers began to implement Grid Resource Brokers to solve the problem of resource management and scheduling, the first challenge was matchmaking: matching user jobs to grid resources. Now we are facing the same challenge at a higher level: we need to automate broker selection for job requests. The matchmaking process of the Meta-Broker uses the previously introduced languages for matching the user requests to the description of the interconnected brokers. During broker utilization the successful submissions and failures should be tracked, and regarding these events a rank should be modified for each special attribute in the BPDFL of the appropriate broker. The JSDL contains the user request this supposed to be an exact specification of the user's job. During matchmaking the following steps should be taken to find the fittest broker:

1. The Matchmaker compares the JSDL of the actual job to the BPDFL of the registered resource brokers. First the basic attributes are matched against the basic properties: this selection determines a group of brokers that are able to submit the job.
2. In the second phase those brokers are kept, which are able to fulfill the special requirement attributes of the job (these capabilities are looked up from the BPDFL).
3. Finally a priority list of the remaining brokers is created taking into account the ranks (stored for the requested features). The first resource broker is chosen from the list.

Since meta-brokering is in a premature state yet, during design and development we need to take into account the existing standards, data models, etc.; furthermore we need to create the missing ones. These tools should be general enough to allow the researchers to implement new grid scheduling and resource management policies based on complex criteria. Therefore we designed this data model incorporating metrics that can be used by future research (not only in matchmaking):

- Broker performance metrics (e.g. average waiting time, throughput),
- Broker historic data,
- Reputation of brokers (e.g. achieved QoS),
- Level of availability/reliability (of brokers and resources behind the brokers).

## 5 Conclusions

The introduced meta-brokering approach opens a new way for interoperability support. Creating such a Meta-Broker, standardized and extensible description languages are needed.

In this paper we proposed a formalization of a data model for storing grid resource broker capabilities and showed how an implementation of this model can be realized and used in a matchmaking scenario. We have presented an XML schema for the Broker Property Description Language that implements the data model. Following our approach, the current Grid Brokers should implement the BDPL to become a member of federation of Grid Brokers managed by a global meta-brokering system.

This work enhances establishing better interoperability among the current production grids and user groups; therefore it enables more beneficial resource utilization and collaboration by global resource management and scheduling. It also considers other mechanisms to achieve collaborative communities in a decentralized environment such as agreements mechanism. Research in this area should focus on the development of infrastructures enabling interoperability achieved by defining new standards, creating sophisticated scheduling policies in terms of global resource usage (e.g. global load balancing) to allow future grids to be more transparent and efficient.

## References

- [1] I. Foster, C. Kesselman, "Computational Grids, The Grid: Blueprint for a New Computing Infrastructure", Morgan Kaufmann, 1998. pp. 15-52.
- [2] S. Venugopal, R. Buyya, L. Winton, "A Grid Service Broker for Scheduling e-Science Applications on Global Data Grids", *Concurrency and Computation: Practice and Experience*, Volume 18, Issue 6, 2006, pp. 685-699.
- [3] E. Huedo, R. S. Montero, I. M. Llorente, "A framework for adaptive execution in grids", *Software: Practice and Experience*. vol. 34, 7, 2004, pp. 631-651.
- [4] E. Elmroth and J. Tordsson, "An Interoperable Standards-based Grid Resource Broker and Job Submission Service", *First IEEE Conference on e-Science and Grid Computing*, IEEE Computer Society Press, 2005, pp. 212-220.
- [5] GridLab Grid Resource Management System (GRMS), [http://www.gridlab.org/WorkPackages/wp-9/res/docs/GRMS\\_1.9.3\\_UserGuide.pdf](http://www.gridlab.org/WorkPackages/wp-9/res/docs/GRMS_1.9.3_UserGuide.pdf)
- [6] A. Kertesz, G. Sipos and P. Kacsuk, "Multi-Grid Broker Utilization with the P-GRADE Portal", *Post-Proceedings of the Austrian Grid Symposium 2006*, OCG Verlag, Austria, 2007.
- [7] I. Rodero, J. Corbalan, R.M. Badia, J. Labarta, "eNANOS Grid Resource Broker", P.M.A. Sloot et al. (Eds.): *EGC 2005*, LNCS 3470, pp. 111-121, ISBN: 3-540-26918-5, Amsterdam, The Netherlands, 14-16 February, 2005.
- [8] The GRIP project web site, <http://www.grid-interoperability.org/grip-workpackages.htm>
- [9] I. Foster C. Kesselman, "The Globus project: A status report", in *Proc. of the Heterogeneous Computing Workshop*, IEEE Computer Society Press, 1998, pp. 4-18.
- [10] Globus Team, *Globus Toolkit 4.0 Release Manuals*, <http://www.globus.org/toolkit/docs/4.0/>
- [11] D. W. Erwin and D. F. Snelling., "UNICORE: A Grid Computing Environment", In *Lecture Notes in Computer Science*, volume 2150, Springer, 2001, pp. 825-834.
- [12] The HPC-Europa Project website, <http://www.hpc-europa.org>
- [13] OGF OGSA Resource Selection Services WG. <https://forge.gridforum.org/sf/projects/ogsa-rss-wg>
- [14] OGF Grid Scheduling Architecture Research Group: <https://forge.gridforum.org/sf/projects/gsa-rg>
- [15] LCG-2 User Guide, 4 August, 2005: <https://edms.cern.ch/file/454439/2/LCG-2-User-Guide.html>
- [16] The gLite website, <http://glite.web.cern.ch/glite/>
- [17] Job Submission Description Language: <http://www.ggf.org/documents/GFD.56.pdf>
- [18] A. Kertesz, P. Kacsuk, "A Taxonomy of Grid Resource Brokers", 6th Austrian-Hungarian Workshop on Distributed and Parallel Systems (DAPSYS), Innsbruck, Austria, 2006.
- [19] A. Pugliese, D. Talia and R. Yahyapour, "Modeling and Supporting Grid Scheduling", *CoreGrid Technical Report no. 56*, August 2006.

## 6 Appendix: BPD XML Schema

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:bpdl="uri:BrokerPropertyDescriptionLanguage"
  targetNamespace="uri:BrokerPropertyDescriptionLanguage" elementFormDefault="qualified"
  attributeFormDefault="unqualified">
3   <xsd:element name="BPD" type="bpdl:BPD_Type">
4     <xsd:annotation>
5       <xsd:documentation>Broker Property Description Language</xsd:documentation>
6     </xsd:annotation>
7   </xsd:element>
8   <xsd:complexType name="BPD_Type">
9     <xsd:sequence>
10      <xsd:any namespace="##other" processContents="lax"/>
11      <xsd:element name="BasicProperties" type="bpdl:BasicProperties_Type"/>
12      <xsd:element name="SpecialProperties" type="bpdl:SpecialProperties_Type"/>
13    </xsd:sequence>
14    <xsd:attribute name="name" type="xsd:NCName" use="required"/>
15    <xsd:attribute name="description" type="xsd:string" use="optional"/>
16    <xsd:attribute name="targetNameSpace" type="xsd:anyURI" use="optional"/>
17    <xsd:anyAttribute namespace="##other" processContents="lax"/>
18  </xsd:complexType>
19  <xsd:complexType name="BasicProperties_Type">
20    <xsd:sequence>
21      <xsd:any namespace="##other"/>
22      <xsd:element name="BrokerName" type="bpdl:BrokerName_Type"/>
23      <xsd:element name="Middlewares" maxOccurs="unbounded">
24        <xsd:complexType>
25          <xsd:simpleContent>
26            <xsd:extension base="bpdl:MiddlewareEnumeration">
27              <xsd:attribute name="accessMethod" type="xsd:string" use="optional"/>
28              <xsd:attribute name="VOs" type="xsd:string" use="optional"/>
29              <xsd:anyAttribute namespace="##other"/>
30            </xsd:extension>
31          </xsd:simpleContent>
32        </xsd:complexType>
33      </xsd:element>
34      <xsd:element name="JobTypes" type="bpdl:JobTypeEnumeration" maxOccurs="unbounded"/>
35      <xsd:element name="Interfaces" maxOccurs="unbounded">
36        <xsd:complexType>
37          <xsd:sequence>
38            <xsd:element name="Interface" type="bpdl:InterfaceType"/>
39          </xsd:sequence>
40        </xsd:complexType>
41      </xsd:element>
42      <xsd:element name="Monitoring" type="bpdl:Monitoring_Type"/>
43    </xsd:sequence>
44  </xsd:complexType>
45  <xsd:complexType name="SpecialProperties_Type">
46    <xsd:sequence>
47      <xsd:any namespace="##other"/>
48      <xsd:element name="RemoteFileHandling" type="bpdl:RemoteFileHandlingEnumeration"
49        maxOccurs="unbounded"/>
50      <xsd:element name="SchedulingPolicies" type="bpdl:SchedulingPolicies_Type"/>
51      <xsd:element name="FaultToleranceMechanisms"
52        type="bpdl:FaultToleranceMechanismsEnumeration" maxOccurs="unbounded"/>
53      <xsd:element name="QoSMechansims" type="bpdl:QoSMechanismsEnumeration"
54        maxOccurs="unbounded"/>
55      <xsd:element name="Agreements" type="bpdl:Agreemets_Type"/>
56      <xsd:element name="PerformanceMetrics" type="bpdl:PerformanceMetrics_Type"/>
57    </xsd:sequence>
58  </xsd:complexType>
59  <xsd:complexType name="BrokerName_Type">
60    <xsd:simpleContent>
61      <xsd:extension base="xsd:string">
62        <xsd:anyAttribute namespace="##other" processContents="lax"/>
63      </xsd:extension>
64    </xsd:simpleContent>
65  </xsd:complexType>
```

```

60     </xsd:extension>
61     </xsd:simpleContent>
62 </xsd:complexType>
63 <xsd:complexType name="SchedulingPolicies_Type">
64     <xsd:sequence>
65         <xsd:element name="PolicyName" type="xsd:string"/>
66         <xsd:element name="Details" type="xsd:string"/>
67         <xsd:element name="Description" type="xsd:string"/>
68     </xsd:sequence>
69     <xsd:anyAttribute namespace="##other"/>
70 </xsd:complexType>
71 <xsd:complexType name="InterfaceType">
72     <xsd:sequence>
73         <xsd:element name="type" type="bpd:InterfacesEnumeration"/>
74         <xsd:element name="name" type="xsd:string"/>
75         <xsd:element name="description" type="xsd:string" minOccurs="0"/>
76         <xsd:element name="Parameters" minOccurs="0">
77             <xsd:annotation>
78                 <xsd:documentation>In case an interface is not a standard one, the xml should
                    contain a description what the interface does and what their required
                    parameters are.</xsd:documentation>
79             </xsd:annotation>
80         <xsd:complexType>
81             <xsd:sequence>
82                 <xsd:element name="Parameter" maxOccurs="unbounded">
83                     <xsd:complexType>
84                         <xsd:sequence>
85                             <xsd:element name="description" minOccurs="0"/>
86                         </xsd:sequence>
87                         <xsd:attribute name="name" type="xsd:NCName"/>
88                         <xsd:attribute name="type" type="xsd:NCName"/>
89                     </xsd:complexType>
90                 </xsd:element>
91             </xsd:sequence>
92         </xsd:complexType>
93     </xsd:element>
94     <xsd:element name="ReturnedValue" minOccurs="0">
95         <xsd:annotation>
96             <xsd:documentation>Same as Parameters notes.</xsd:documentation>
97         </xsd:annotation>
98     <xsd:complexType>
99         <xsd:sequence>
100             <xsd:element name="description" minOccurs="0"/>
101         </xsd:sequence>
102         <xsd:attribute name="name" type="xsd:NCName"/>
103         <xsd:attribute name="type" type="xsd:NCName"/>
104     </xsd:complexType>
105 </xsd:element>
106 </xsd:sequence>
107 </xsd:complexType>
108 <xsd:complexType name="MonitoringMetric_Type">
109     <xsd:sequence>
110         <xsd:element name="Name" type="xsd:string"/>
111         <xsd:element name="Description" type="xsd:string"/>
112     </xsd:sequence>
113     <xsd:attribute name="MetricType" type="bpd:MonitoringInfoEnumeration" use="required"/>
114     <xsd:anyAttribute namespace="##other"/>
115 </xsd:complexType>
116 <xsd:complexType name="Monitoring_Type">
117     <xsd:sequence>
118         <xsd:element name="Metrics" type="bpd:MonitoringMetric_Type" maxOccurs="unbounded"/>
119     </xsd:sequence>
120     <xsd:attribute name="accessMethod" type="xsd:string" use="optional"/>
121     <xsd:anyAttribute namespace="##other"/>
122 </xsd:complexType>
123 <xsd:complexType name="Agreemets_Type">
124     <xsd:sequence>
125         <xsd:element name="Agreement" type="bpd:Agreement_Type" maxOccurs="unbounded"/>

```

```

126     </xsd:sequence>
127     <xsd:anyAttribute namespace="##other" />
128 </xsd:complexType>
129 <xsd:complexType name="Agreement_Type">
130     <xsd:sequence>
131         <xsd:element name="Target" type="xsd:anyURI" />
132         <xsd:element name="ConfidenceLevel" type="xsd:positiveInteger" />
133     </xsd:sequence>
134     <xsd:anyAttribute namespace="##other" />
135 </xsd:complexType>
136 <xsd:complexType name="PerformanceMetrics_Type">
137     <xsd:sequence>
138         <xsd:element name="AVGWaitingTime" type="bpd:PerformanceMetric_Type" />
139         <xsd:element name="AVGSlowdown" type="bpd:PerformanceMetric_Type" />
140         <xsd:element name="FinishedJobs" type="bpd:PerformanceMetric_Type" />
141         <xsd:element name="FailedJobs" type="bpd:PerformanceMetric_Type" />
142         <xsd:element name="OtherMetrics" type="bpd:PerformanceMetric_Type"
143             maxOccurs="unbounded" />
144     </xsd:sequence>
145     <xsd:anyAttribute namespace="##other" />
146 </xsd:complexType>
147 <xsd:complexType name="PerformanceMetric_Type">
148     <xsd:sequence>
149         <xsd:element name="name" type="xsd:string" />
150         <xsd:element name="description" type="xsd:string" />
151         <xsd:element name="value" type="xsd:string" />
152     </xsd:sequence>
153     <xsd:anyAttribute namespace="##other" />
154 </xsd:complexType>
155 <xsd:simpleType name="MiddlewareEnumeration">
156     <xsd:restriction base="xsd:string">
157         <xsd:enumeration value="GT2" />
158         <xsd:enumeration value="GT3" />
159         <xsd:enumeration value="GT4" />
160         <xsd:enumeration value="EGEE-LCG-2" />
161         <xsd:enumeration value="EGEE-gLite" />
162         <xsd:enumeration value="Nordugrid" />
163         <xsd:enumeration value="Unicore" />
164         <xsd:enumeration value="other" />
165     </xsd:restriction>
166 </xsd:simpleType>
167 <xsd:simpleType name="JobTypeEnumeration">
168     <xsd:restriction base="xsd:string">
169         <xsd:enumeration value="Serial" />
170         <xsd:enumeration value="Mpi" />
171         <xsd:enumeration value="Pvm" />
172         <xsd:enumeration value="Checkpointable" />
173         <xsd:enumeration value="Interactive" />
174         <xsd:enumeration value="Threads" />
175         <xsd:enumeration value="OpenMP" />
176         <xsd:enumeration value="Mpi+OpenMP" />
177         <xsd:enumeration value="Caf" />
178         <xsd:enumeration value="Upc" />
179         <xsd:enumeration value="other" />
180     </xsd:restriction>
181 </xsd:simpleType>
182 <xsd:simpleType name="InterfacesEnumeration">
183     <xsd:restriction base="xsd:string">
184         <xsd:enumeration value="Submit" />
185         <xsd:enumeration value="Cancel" />
186         <xsd:enumeration value="Suspend" />
187         <xsd:enumeration value="Resume" />
188         <xsd:enumeration value="Migrate" />
189         <xsd:enumeration value="other" />
190     </xsd:restriction>
191 </xsd:simpleType>
192 <xsd:simpleType name="MonitoringInfoEnumeration">
193     <xsd:restriction base="xsd:string">

```

```

193     <xsd:enumeration value="StaticInfo" />
194     <xsd:enumeration value="DynamicInfo" />
195     <xsd:enumeration value="AggregatedInfo" />
196     <xsd:enumeration value="other" />
197   </xsd:restriction>
198 </xsd:simpleType>
199 <xsd:simpleType name="RemoteFileHandlingEnumeration">
200   <xsd:restriction base="xsd:string">
201     <xsd:enumeration value="GridFTP" />
202     <xsd:enumeration value="RFT" />
203     <xsd:enumeration value="GASS" />
204     <xsd:enumeration value="Unicore" />
205     <xsd:enumeration value="SRB" />
206     <xsd:enumeration value="other" />
207   </xsd:restriction>
208 </xsd:simpleType>
209 <xsd:simpleType name="FaultToleranceMechanismsEnumeration">
210   <xsd:restriction base="xsd:string">
211     <xsd:enumeration value="Checkpointing" />
212     <xsd:enumeration value="Rescheduling" />
213     <xsd:enumeration value="Replication" />
214     <xsd:enumeration value="other" />
215   </xsd:restriction>
216 </xsd:simpleType>
217 <xsd:simpleType name="QoSMechanismsEnumeration">
218   <xsd:restriction base="xsd:string">
219     <xsd:enumeration value="AdvancedReservations" />
220     <xsd:enumeration value="Priorities" />
221     <xsd:enumeration value="Budget" />
222     <xsd:enumeration value="Timeouts" />
223     <xsd:enumeration value="other" />
224   </xsd:restriction>
225 </xsd:simpleType>
226 </xsd:schema>

```