

## **CIM Database Model for Data Access and Integration Services: Scenarios**

### Status of This Memo

This memo provides information to the Grid community on the potential use of the DMTF CIM Database model to support the deployment and use of Data Access and Integration Services. It does not define any standards or technical recommendations. Distribution is unlimited.

### Copyright Notice

Copyright © Global Grid Forum (2004). All Rights Reserved.

### **Abstract**

The DMTF Common Information Model Database Model [1] is a conceptual information model that describes management aspects of a database environment. This includes the characteristics of the software that is supporting database management, and statistical information on the behavior of a database. Such models are potentially useful for the discovery, management and use of databases accessed through data service interfaces. This document introduces some scenarios that motivate the deployment and further development of the CIM database model for use in a service-based environment.

### Contents

Abstract .....	1
1. Introduction .....	2
2. Data Service Discovery Scenarios .....	2
2.1 Discovery for Access .....	2
2.2 Discovery for Administration .....	3
2.3 Discovery for Federation .....	3
2.4 Discovery for Replication .....	4
2.5 Discovery for Evolution .....	5
3. Data Service Deployment Scenarios .....	6
3.1 Implementing Access .....	6
3.2 Implementing Administration .....	7
3.3 Implementing Federation .....	8
3.4 Implementing Replication .....	8
3.5 Implementing Evolution .....	8
4. Descriptions of Relevant Information Model Concepts not in CIM Version 2.8 .....	9
4.1 Logical Schema of the Database .....	9
4.2 Physical Schema of the Database .....	10
4.3 Access Control Definitions of the Database .....	10
4.4 Database System Capabilities .....	10
5. Summary list of database related classes in CIM Version 2.8 .....	11
6. Security Considerations .....	11
Author Information .....	11
Intellectual Property Statement .....	11
Full Copyright Notice .....	12
References .....	12
Other Useful Documents and Web-Sites .....	13

## 1. Introduction

The DMTF Common Information Model Database Model [1] is a conceptual information model that describes management aspects of a database environment. This includes the characteristics of the software that is supporting database management, and statistical information on the behavior of a database. Such models are potentially useful for the discovery, management and use of databases accessed through data service interfaces.

The goal of this document is to define scenarios to drive the GGF CGS-WG activity to extend the CIM data model to take into account grid and DAIS-WG requirements. This document should support this activity by mapping DAIS concepts and management related requirements to CIM:

- Where specific mappings exist, reusing the CIM classes.
- Where mappings do not exist, extending the CIM models.

In order to succeed in setting up and running a complete data or database system, administration and management scenarios have to be considered, in addition to those focusing on application access and integration, which is the focus of the DAIS-WG. Where possible, common constructs should be modeled in the same way, and similar (if not the same) terms should be used across management (which DMTF and CGS-WG are focused on) and DAIS-WG data access activities.

The document is structured as follows. Section 2 describes scenarios relating to the discovery of a database for subsequent use through a data service interface. Section 3 describes scenarios relating to the deployment of a database that may subsequently be accessed through a data service interface. Section 4 describes categories of information that are not currently included in CIM that are potentially relevant to DAIS-WG activities.

## 2. Data Service Discovery Scenarios

In Grid environments, software and hardware resources must be described in a precise and systematic manner if they are to be able to be discovered for subsequent management or use. This section discusses different discovery scenarios, in particular, discovery to support the access, evolution, federation and replication of a database.

### 2.1 Discovery for Access

Access to a database is considered here to involve any use of a database by an application or user. The DAIS-WG is developing specifications for services that allow a database to be accessed using the Data Manipulation Language (DML) of the database. A database could be selected for use on the basis of the following existing CIM Database Model classes:

1. *CIM\_Product*: Items such as the *Name*, *Vendor* or *Version* could be relevant to the selection of a database for wider use. For example, a user may be familiar (or not) with the language features of *MyDB*.
2. *CIM\_DatabaseSystem*: Items such as the *Name*, *PrimaryOwnerName* and *Description*, could be relevant to the selection of a database for wider use. For example, a user may know that the database owned by *Cynthia Smith* contains relevant information.
3. *CIM\_CommonDatabase*: Items such as the *OperationalStatus* could be useful, for example, in selecting an installation of a database for use where several replicas are described in a registry, and the *Caption* could provide helpful initial information on contents. For example, a user may have a requirement to obtain some information within a strict time period, and thus need to know if a particular database is currently operational.

A database could also be selected for use based on characteristics of the database that are not currently in the CIM Database Model. Examples include the following:

1. *The Logical Schema of the Database*: For details see Section 4.1. For example, a user may select a database for use on the basis of the names of the tables in the database, and may infer properties of the tables from details of the columns associated with the tables.

## 2.2 Discovery for Administration

Database administration involves a wide range of activities, such as creating and destroying databases, configuring the database server, managing users, and importing or exporting data. The DAIS-WG is defining specifications for services that allow access using the full Data Definition Language (DDL) facilities of SQL, which includes the ability to create, modify or delete tables and other database objects (triggers, stored procedures, etc), manipulate the physical schema of a database, and manage authentication. A database might be selected for such administrative tasks on the basis of the following existing CIM Database Model classes:

1. *CIM\_Product*: Items such as the *Name*, *Vendor* or *Version* could be relevant to the selection of a database for wider use. For example, different database administrators may only be qualified or permitted to conduct administrative tasks on specific platforms.
2. *CIM\_DatabaseSystem*: Items such as the *Name*, *PrimaryOwnerName* and *Description*, could be relevant to the selection of a database for administrative activity. For example, an administrator may know the name of a database for which administrative tasks are required, or may know that the database owned by *Cynthia Smith* requires certain administrative tasks to be conducted.
3. *CIM\_CommonDatabase*: Items such as the *SizeAllocated* could be useful to a database administrator. For example, an administrator could be looking for a database that can accommodate the storage of data for a rapidly growing data collection.
4. *CIM\_DatabaseService*: Items such as the *ConnectionLimit* could be relevant. For example, a database administrator might need to know such limits before moving additional data into a database, or broadening access permissions.
5. *CIM\_DatabaseServiceStatistics*: Items relating to resource statistic, space usage and loads are relevant to many administrative tasks. For example, loading information could be useful in estimating the likelihood that a service can satisfy quality of service requirements if extensions are made to the range of data being stored in the database.

A database could also be selected for administrative tasks based on characteristics of the database that are not currently in the CIM Database Model. Examples include the following:

1. *The Logical Schema of the Database*: For details see Section 4.1. For example, as an administrator can change the logical schema, it is likely to be important to know the current logical schema.
2. *The Physical Schema of the Database*: For details see Section 4.2. For example, as the administrator may be seeking to improve the performance provided by a database, it is likely to be important to know the current physical schema.

## 2.3 Discovery for Federation

Federation of a collection of databases is considered here to involve the use of a database integration middleware that allows a query to be evaluated over data that resides in more than one database. The DAIS-WG is developing specifications for services that allow a database to be accessed using the Data Manipulation Language (DML) of the database; such services can be used by a federating middleware to provide consistent access to diverse data resources. A database could be selected for use on the basis of the following existing CIM Database Model classes:

1. *CIM\_Product*: Items such as the *Name*, *Vendor* or *Version* could be relevant to the selection of a database for wider use. For example, a middleware may be able (or not) to express requests using the language features of *MyDB*.
2. *CIM\_DatabaseSystem*: Items such as the *Name*, *PrimaryOwnerName* and *Description*, could be relevant to the selection of a database for wider use. For example, a user constructing a federation may know that the database owned by *Cynthia Smith* contains relevant information.
3. *CIM\_CommonDatabase*: Items such as the *OperationalStatus* or *Version* could be useful, for example, in selecting an installation of a database for inclusion in a federation where several mirrors are described in a registry, and the *Caption* could provide helpful initial information on contents. For example, a federation could not readily be constructed where a participating database is currently operational.
4. *CIM\_DatabaseService*: Items such as the *ConnectionLimit* could be relevant. For example, a federation may be being constructed for use by collections of users that require concurrent access to databases in the federation.
5. *CIM\_DatabaseServiceStatistics*: Items such as *LastActivity*, *ActiveConnections* and *DiskReads* could be useful when identifying the suitability of a database for inclusion in a federation. For example, such loading information could be useful in estimating the likelihood of a service satisfying quality of service requirements of the federation.

A database could also be selected for use based on characteristics of a database that are not currently in the CIM Database Model. Examples include the following:

1. *The Logical Schema of the Database*: For details see Section 4.1. For example, a federating middleware is likely to import much of the logical schemas of the databases to be federated, for use validating queries submitted to the federation.
2. *The Physical Schema of the Database*: For details see Section 4.2. For example, a federating middleware is likely to use some sort of cost model when optimizing queries submitted to the federation.

## 2.4 Discovery for Replication

Replication of a database is considered here to mean the copying of the contents of a database to another location, for performance or resilience reasons, whether using the mechanisms provided by a DBMS or using some other middleware. This is potentially relevant to the DAIS-WG, as DAIS services could be used by a middleware that supports database replication. A database could be selected for use on the basis of the following existing CIM Database Model classes:

1. *CIM\_Product*: Items such as the *Name*, *Vendor* or *Version* could be relevant to the selection of a database for wider use. For example, a specific version of a database product may provide built-in facilities to support replication.
2. *CIM\_DatabaseSystem*: Items such as the *Name*, *PrimaryOwnerName* and *Description*, could be relevant to the selection of a database for replication use. For example a user selecting databases for replication may know that the database owned by *Cynthia Smith* contains relevant information.
3. *CIM\_CommonDatabase*: Items such as the *OperationalStatus* could be useful. For example, replication cannot be conducted unless the source database is operational.
4. *CIM\_DatabaseService*: Items such as the *ConnectionLimit* could be relevant. For example, regular propagation of updates from a database to a replica will impose additional load on the source, so known limits could affect quality of service.
5. *CIM\_DatabaseServiceStatistics*: Items such as *LastActivity*, *ActiveConnections* and *DiskReads* could be useful when identifying the suitability of a database for replication. For example, such loading information could be useful in establishing the need to create a replica for performance reasons.

A database could also be selected for use based on characteristics of a database that are not currently in the CIM Database Model. Examples include the following:

1. *The Logical Schema of the Database*: For details see Section 4.1. For example, a replicating middleware is likely to transport the logical schemas of the databases to be replicated prior to replicating the data.
2. *The Physical Schema of the Database*: For details see Section 4.2. For example, decisions on replication are likely to depend on the physical properties of the data that is under consideration for replication.

After replication has been conducted, information is required to track replicas (a replica catalogue); such information is not considered in the report, although such information could certainly be relevant to a grid context. The GGF O-REP working group has specified a replica catalog mechanism. A BOF is proposed at GGF10 for an Information Dissemination working group that may also be relevant to this section.

## 2.5 Discovery for Evolution

Evolution is considered here to involve any change to the configuration of previously established federation or replicated databases. There are two obvious subtasks relating to evolution, namely the discovery of issues with an existing setup, and the identification of resources that might be useful for resolving the issues. This section does not consider access to information describing federation or replication activities, as this is considered beyond the scope of the DAIS-WG.

### 2.5.1 Issue Identification

Discovery tasks could be conducted to identify issues with databases participating in federation or replication activities on the basis of the following existing CIM Database Model classes:

1. *CIM\_Product*: Items such as the *Name*, *Vendor* or *Version* could be relevant to the selection of a database for wider use. For example, changes to licensing arrangements may affect the way a product is to be used in future.
2. *CIM\_DatabaseSystem*: Items such as the *Name* or *PrimaryOwnerName* could be relevant to the selection of a database for wider use. For example, a specific database, or the databases owned by a specific individual may no longer be being made available for participation in a specific federation.
3. *CIM\_CommonDatabase*: Items such as the *OperationalStatus* could be useful. For example, a prolonged period in which a database is not operation could lead to it being removed from a federation or as a source of data from replication.
4. *CIM\_DatabaseService*: Items such as the *ConnectionLimit* or the *ServiceAvailableToDatabase* could be relevant. For example, difficulties could be being encountered with the connection limit, or a database may no longer be made available through a service, with consequences for federation and replication activities.
5. *CIM\_DatabaseServiceStatistics*: Items such as *LastActivity*, *ActiveConnections* and *DiskReads* could be useful for comparing the loads on databases that might participate in federation or replication activities. For example, data might be offloaded from a specific database, or the level of replication increased for a data collection.

A database could also be selected for evolution based on characteristics of the database that are not currently in the CIM Database Model. Examples include the following:

1. *The Logical Schema of the Database*: For details see Section 4.1. For example, the demand for specific kinds of data may be anticipated to increase, giving rise to the need for additional replication of some data.

2. *The Physical Schema of the Database*: For details see Section 4.2. For example, changes to the size of some data collections may give rise to a decision to place data differently.

### 2.5.2 Issue Resolution

The resolution of issues with existing federations or replication configurations may involve a search for databases that are suitable to replace an existing database, take some of the load of an existing database, etc. Such discovery tasks are analogous to those required to discover databases for use in federation and replication activities, as described in Sections 2.3 and 2.4.

## 3. Data Service Deployment Scenarios

In Grid environments, software and hardware resources must be described in a precise and systematic way if they are to be able to be accessed and managed in a systematic way. This section describes scenarios for deploying access and management services. Deployment can be classified into the following high-level scenarios:

- Implementing Access.
- Implementing Administration (Logical, Physical and Access Control).
- Implementing Federation.
- Implementing Replication.
- Implementing Evolution.

Typically these deployment scenarios follow on from a scenario that identifies the appropriate data resource (to be deployed) either through discovery or through prior knowledge. The deployment scenarios can themselves be combined. For example, implementing federation could be followed by implementing access where an application needs to access data through a federation of a number of sources. A composite scenario for accessing a data resource through a federation could be constructed as follows:

- Discover a Data Resource.
- Implement Federation on the Data Resource.
- Place the Accessing Application in an appropriate location (this portion is out of scope for this document).
- Implement Access to the Data Resource.

Another composite scenario for accessing a data resource through a replica could be constructed as follows:

- Discover a Data Resource.
- Implement Replication on the Data Resource near the Accessing Application.
- Implement Access to the Data Resource.

### 3.1 Implementing Access

After identifying a database, e.g., through discovery or prior knowledge, the database may then be accessed. The DAIS working group describes how access to a database is performed in a grid environment using DML. Access includes both query and update. When accessing a database, it may be helpful to find out the following information that already exists in the CIM model:

1. *CIM\_Product*: Items such as the *Name*, *Vendor* or *Version* could be relevant when accessing a database. For example, a particular access feature included in a recent version may be required.

2. *CIM\_DatabaseSystem*: Items such as the *Name*, *PrimaryOwnerName* and *Description*, could be relevant when accessing a database. For example, a user may know that the database owned by *Cynthia Smith* contains the most up date information in a particular domain of interest.
3. *CIM\_CommonDatabase*: Items such as the *OperationalStatus* could be useful. For example, access cannot be conducted unless the source database is operational.

Other items that may be very useful when implementing access do not appear in the current CIM model but do appear in JDBC metadata [3]. Examples include:

1. *The Logical Schema of the Database*: For details see Section 4.1. For example, a user may formulate queries without prior knowledge of specific table and column names.
2. *Database System Capabilities*: There are many pieces of information that assist the creation of general-purpose software to access databases in a grid environment that fall into this category. For example, a user may need to know the level of support for SQL before accessing a database. For details of database system capabilities, see Section 4.4.

### 3.2 Implementing Administration

After identifying a database, e.g., through discovery or prior knowledge, the database may then be administered. The DAIS working group describes how administration of a database is performed in a grid environment using DDL, for which the following existing CIM Database Model classes could be useful:

1. *CIM\_Product*: Items such as the *Name*, *Vendor* or *Version* could be relevant when administering a database. For example, a particular administration feature included in a recent version may be required.
2. *CIM\_CommonDatabase*: Items such as the *OperationalStatus* could be useful. For example, the database may require administration (such as restarting) if it is currently not operational.
3. *CIM\_DatabaseServiceStatistics*: Items such as *LastActivity*, *ActiveConnections* and *DiskReads* could be useful for an administrator trying to improve the performance of the database system.

Database administration through DDL can be classified into logical schema administration, physical schema administration and access control administration. Classes associated with all three categories do not appear in the current CIM data model. It would be helpful for administering databases in a heterogeneous grid environment if these pieces of information are available:

1. *The Logical Schema of the Database*: For details see Section 4.1. For example the administrator or administration software may require to add or to modify some table definitions due to a new application being developed.
2. *The Physical Schema of the Database*: For details see Section 4.2. For example, as the administrator or administration software may be seeking to improve the performance provided by a database, it is likely to be important to be able to modify physical schema.
3. *Access Control Definitions of the Database*: For details see Section 4.3. For example, the administrator or administration software may wish to add more users to access the system, or to modify the privileges of existing users.
4. *Database System Capabilities*: For details see Section 4.4. For example, the administrator or administration software may need to know if this database supports stored procedures in order to ensure adequate memory is available.

### 3.3 Implementing Federation

After identifying a database, e.g., through discovery or prior knowledge, the database may then be included in a database federation. Alternatively, multiple databases could be identified, and a data federation could be created. There are a number of pieces of information needed to perform a database federation that are in the CIM database model:

1. *CIM\_Product*: Items such as the *Name*, *Vendor* or *Version* could be relevant when administering a database. For example, a particular federation feature included in a recent version may be required.

*CIM\_DatabaseServiceStatistics*: Items such as *LastActivity*, *ActiveConnections* and *DiskReads* could be useful for an administrator trying to configure a database federation.

Essential to the creation or modification of a data federation, are the following pieces of information that are not currently part of the CIM data model:

1. *The Logical Schema of the Database*: For details see Section 4.1. For example, a federating middleware is likely to need access to, or even to import, the logical schemas of the databases to be federated, for use when executing queries submitted to the federation.
2. *The Physical Schema of the Database*: For details see Section 4.2. For example, a federating middleware is likely to use some sort of cost model if it optimizes queries submitted to the federation.
3. *Database System Capabilities*: For details see Section 4.4. For example, the federating software may need to know the kinds of queries this database supports, before constructing queries to access the data.

### 3.4 Implementing Replication

Having identified a database for replication, database replication can be performed. There are a number of pieces of information needed to perform a database replication, e.g., to identify the replica source, to identify the replica target, to identify the portion of the data to be replicated, the characteristics of the replica, e.g., master slave, read only, synchronized at most 3 minutes behind. We exclude identifying characteristics of replicas from the CIM model as the characteristics are not particularly pertinent to the DAIS-WG, although the characteristics would be relevant for databases in a grid environment in general.

In the CIM model, the following information could assist in implementing replication:

1. *CIM\_Product*: Items such as the *Name*, *Vendor* or *Version* could be relevant to the accessing a database. For example, a particular replication feature included in a recent version may be used.

Useful to the creation of a replica are the following pieces of information that do not appear in the CIM model:

1. *Access Control Definitions of the Database*: For details see Section 4.3. For example, the administrator or administration software may wish to ensure the same access control definitions exist in the replica.
2. *Database System Capabilities*: For details see Section 4.4. For example, the administrator or administration software may need to know if this database supports a particular kind of query, to generate the data for the replica in the most efficient way.

### 3.5 Implementing Evolution

Issue identification relating to evolution may take place through discovery, prior knowledge, or through explicit notification. Having identified issues with one or more databases, in a database

replication or in a database federation or in a stand-alone environment, it may then be necessary to implement changes, e.g., to restore consistency or availability, and then to check that the implementation was effective.

Implementing changes as a result of an evolution in one or more databases in a system or as a result of an environmental change, would be made up of executing a combination of steps outlined in sections 2.1-2.4 and 3.1-3.4 above. Checking the restoration of consistency or availability after implementing the changes may require access to a variety of pieces of information. Here are some examples that include information that is already in the CIM model:

1. *CIM\_Product*: Items such as the *Name*, *Vendor* or *Version* could be relevant, e.g., to ensure that all systems are now at the same version.
2. *CIM\_DatabaseSystem*: Items such as the *PrimaryOwnerName* could be relevant, e.g., if the primary owner of the database has changed, and it has been necessary to modify the security scheme and privileges of the users. It may be necessary to ensure that all the owners of a collection of databases are now compatible or from the same organization.

*CIM\_CommonDatabase*: Items such as the *OperationalStatus* could be useful. For example, if a database becomes non-operational in a federation, it may be appropriate to bring in another database into the federation, e.g., a replica. It may be necessary to ensure that all databases in a system are now operational. -

After an evolution, it may be necessary to check characteristics of the database that are not currently in the CIM Database Model. Examples include the following:

1. *The Logical Schema of the Database*: For details see Section 4.1. For example, to ensure the table structures across a number of databases are compatible.
2. *The Physical Schema of the Database*: For details see Section 4.2. For example, to ensure that after implementing a series of changes in the number of disks available to the system, there is enough storage for the anticipated database index growth.
3. *Access Control Definitions of the Database*: For details see Section 4.3. For example, to ensure that after adding a new role to one database in a federation, it may be necessary to ensure that the role is consistently represented across a database federation.

#### 4. Descriptions of Relevant Information Model Concepts not in CIM Version 2.8

##### 4.1 Logical Schema of the Database

The logical schema of a relational database contains the information required to allow a user to issue SQL against that database, and is mentioned in [1] as a possible area for future extension of the CIM Database Model. Such information includes: the names of tables, the names and types of columns, the integrity constraints applying to the tables, the views, the packages and stored procedures and the triggers. Vendor neutral descriptions of logical schemas are provided by the ISO/ANSI SQL standard and as part of JDBC in [2] and [3]. Here is a list derived from [2] which could provide a starting point for reviewing the possible content for database logical schemas in the CIM model. The list requires prioritization, and includes typical operation performed on the descriptions through DDL.

- **Schema (schema definition)**: drop schema statement.
- **Table (table definition)**: alter table statement, drop table statement, table constraint definition, add table constraint definition, drop table constraint definition.
- **Constraint (default clause definition, unique constraint definition, referential constraint definition, check constraint definition)**.

- **Column (column definition):** add column definition, alter column definition, set column default clause, drop column default clause, add column scope clause, drop column scope clause, alter identity column specification, drop column definition.
- **View (view definition):** drop view statement.
- **Domain (domain definition):** alter domain statement, set domain default clause, drop domain default clause, add domain constraint definition, drop domain constraint definition, drop domain statement.
- **Character set (character set definition):** drop character set statement, collation definition, drop collation statement, transliteration definition, drop transliteration statement.
- **Trigger (assertion definition, trigger definition):** drop assertion statement, trigger definition, drop trigger statement.
- **User Defined Type (user-defined type definition, attribute definition, method specification, user-defined transform definition, user-defined ordering definition, user-defined cast definition):** alter type statement, drop data type statement, add attribute definition, drop attribute definition, add original method specification, add overriding method specification, drop method specification, drop user-defined cast statement, drop user-defined ordering statement, transform definition, alter transform statement, add transform element list, drop transform element list, drop transform statement.
- **SQL-invoked routine (SQL-invoked routine definition):** alter routine statement, drop routine statement.
- **Sequence Generator (sequence generator definition):** alter sequence generator statement, drop sequence generator statement.

#### 4.2 Physical Schema of the Database

The physical schema of a relational database contains information about the physical organization of the data in the database (e.g., the presence of indexes), and statistics describing the current database extent (e.g., sizes, cardinalities and column value distributions). Vendor neutral descriptions of physical schemas are provided by the ISO/ANSI SQL standard and as part of JDBC in [2] and [3]

#### 4.3 Access Control Definitions of the Database

The database can contain information about access control. Here is a list derived from [2] that includes the access control related definitions and associated operations that can be performed through DDL:

- **User:** grant statement, revoke statement.
- **Privilege:** grant privilege, revoke privilege.
- **Role:** grant role, drop role.

#### 4.4 Database System Capabilities

There are many database capabilities that appear in JDBC Metadata [3], but that are not modeled in CIM. These capabilities are important for applications that are implementing access, federations and replications. Database System Capabilities may be considered to be out of scope for the CIM database model. However, the inclusion of some Database System Capabilities should be reviewed as they include very helpful pieces of information when building a flexible grid infrastructure. Here are just a few:

- Supports ANSI92 entry, intermediate or full, supports ANSI 2003 SQL/XML
- Supports multiple transactions, supports save-points, supports stored procedures

- Supports open cursors across commit, supports open cursors across rollback
- Supports union, supports union all, supports select for update

For more information, see [3]

## **5. Summary list of database related classes in CIM Version 2.8**

The following is a summary list of classes that are relevant to databases in CIM Version 2.8:

- CIM\_DatabaseSystem
- CIM\_AssociatedDatabaseSystem association
- CIM\_CommonDatabase
- CIM\_DatabaseAdministrator association
- CIM\_DatabaseStorageArea
- CIM\_DatabaseFile association
- CIM\_DatabaseControlFile association
- CIM\_DatabaseSegment
- CIM\_CommonDatabaseSettingData
- CIM\_DatabaseService
- CIM\_ServiceAvailableToDatabase association
- CIM\_DatabaseParameter class
- CIM\_DatabaseResourceStatistics
- CIM\_CommonDatabaseStatistics
- CIM\_DatabaseServiceStatistics

Other non-database specific classes in CIM 2.8 referred to in this document include:

- CIM\_Product

## **6. Security Considerations**

The ability to apply regular security mechanisms in all the scenarios listed above has been assumed. In addition, specific access control definitions and operations that can be performed through DDL have been included.

### **Author Information**

Susan Malaika  
IBM Silicon Valley Lab  
San Jose, CA, US  
malaika@us.ibm.com

Norman W Paton  
Department of Computer Science  
University of Manchester M13 9PL,  
UK.  
norm@cs.man.ac.uk

### **Intellectual Property Statement**

The GGF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in

this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the GGF Secretariat.

The GGF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this recommendation. Please address the information to the GGF Executive Director.

### **Full Copyright Notice**

Copyright (C) Global Grid Forum (2004). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the GGF or other organizations, except as needed for the purpose of developing Grid Recommendations in which case the procedures for copyrights defined in the GGF Document process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the GGF or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE GLOBAL GRID FORUM DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE."

### **References**

- [1] CIM Database Model White Paper, CIM Version 2.8, Version 1.3, November 03, 2003  
[http://www.dmtf.org/standards/published\\_documents/DSP0133.pdf](http://www.dmtf.org/standards/published_documents/DSP0133.pdf)
- [2] ISO International Standard (IS) Database Language SQL - Part 11: SQL/Schemata, ISO/IEC 9075-11:2003  
<http://www.iso.ch/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=34917&ICS1=35&ICS2=60&ICS3>
- [3] JDBC Metadata  
<http://java.sun.com/j2se/1.4.2/docs/api/java/sql/DatabaseMetaData.html>
- [4] GGF OGSA Data Services  
[https://forge.gridforum.org/projects/dais-wg/document/OGSA\\_Data\\_Services/en/1](https://forge.gridforum.org/projects/dais-wg/document/OGSA_Data_Services/en/1)
- [5] GGF DAIS Grid Data Service Specification  
[https://forge.gridforum.org/projects/dais-wg/document/Grid\\_Data\\_Service\\_Specification/en/1](https://forge.gridforum.org/projects/dais-wg/document/Grid_Data_Service_Specification/en/1)
- [6] GGF DAIS XML Realization  
[https://forge.gridforum.org/projects/dais-wg/document/Relational\\_Realisation/en/1](https://forge.gridforum.org/projects/dais-wg/document/Relational_Realisation/en/1)
- [7] GGF DAIS Relational Realization  
[https://forge.gridforum.org/projects/dais-wg/document/XML\\_Realisation/en/4](https://forge.gridforum.org/projects/dais-wg/document/XML_Realisation/en/4)

**Other Useful Documents and Web-Sites**

- CGS-WG <https://forge.gridforum.org/projects/cgs-wg>
- DMTF CIM <http://www.dmtf.org/standards/cim>
- DAIS-WG <https://forge.gridforum.org/projects/dais-wg>
- Andrea Westerinen GGF9 Charts
- <https://forge.gridforum.org/projects/cgs-wg/document/Current-CIM-DAIS-Support/en/1>
- JDBC API Tutorial and Reference, Third Edition, by [Maydene Fisher](#), [Jon Ellis](#), and [Jonathan Bruce](#)