CMDB Federation (CMDBf) Frequently Asked Questions (FAQ) White Paper

Version: 1.0.0
Status: DMTF Informational
Publication Date: 2010-05-10
Document Number: DSP2024

Abstract

This white paper describes the key concepts of Configuration Management Database Federation (CMDBf) in a question-and-answer format. Topics include explanations of CMDB, CMDBf, and their components and services, as well as recommendations for implementing them. The questions and answers are organized by role (end user, implementers of repositories and applications to be incorporated in a CMDB federation, and Federating CMDB implementer). The FAQ is intended to provide a simple “on-ramp” for those who want to implement and use the CMDBf specification.

CONTENTS

Figures

1 Introduction

This white paper is intended for use by people trying to apply the *Configuration Management Database Federation (CMDBf) Specification* (DSP0252) in real-world business scenarios. The content of this white paper is organized into sets of questions and answers much like Frequently Asked Questions (FAQs).

This white paper is intended to address the following audiences:

- end users, such as IT managers and IT administrators
- MDR (Management Data Repository) implementers
- Federating CMDB implementers

To make it easier to find content of interest to a particular audience, this document is organized by audience type. Questions are not repeated from section to section. If your question is not covered in a particular section, check the other sections.

2 End Users

Q: What is a CMDB?

A: CMDB is an acronym that stands for Configuration Management Database. It is derived from the Information Technology Infrastructure Library (ITIL) best practices. The ITIL® V3 Glossary defines CMDB as “A database used to store Configuration Records throughout their Lifecycle. The Configuration Management System maintains one or more CMDBs, and each CMDB stores Attributes of CIs, and Relationships with other CIs.” (See ITIL® V3 Glossary, pages not numbered, and DSP0252, clauses 1 and 3.)

Q: What is a CI?

A: CI is an acronym that stands for Configuration Item. The ITIL® V3 Glossary defines a CI as “Any Component that needs to be managed in order to deliver an IT Service. Information about each CI is recorded in a Configuration Record within the Configuration Management System and is maintained throughout its Lifecycle by Configuration Management. CIs are under the control of Change Management. CIs typically include IT Services, hardware, software, buildings, people, and formal documentation such as Process documentation and SLAs.” (See ITIL® V3 Glossary, pages not numbered.)

Exactly which components and attributes are tracked in a CMDB varies from organization to organization depending on the level of control exercised by configuration management and the needs of the organization. (See DSP0252, clauses 1 and 3.)

Q: What type of data is in a CMDB?

A: ITIL practice recommends that a CMDB focus on the data used to manage the configuration of IT resources that support services, including changes to the configuration of these resources. ITIL V3, Service Transition (pp. 92-93), suggests six general categories of data:

- Service lifecycle CIs, which include CIs such as business cases, service lifecycle plans and test plans that show a provider’s services, how they will be delivered, benefits expected, costs, and so on
- Service CIs that relate to specific service processes such as capabilities, models, packages, and acceptance criteria

- Organization CIs, which define organization constraints such as regulatory requirements and business strategies
- Internal CIs, such as data center assets and software
- External CIs, such as customer requirements and agreements
- Interface CIs that are required to deliver service over a service-provider interface

In practice, most organizations do not track all these CI categories in the CMDB. Rather, they pick those most relevant to their business. Of these categories, “Internal CIs” usually represent the core of the CMDB.

ITIL V3 introduces the concept of a comprehensively integrated system for managing services. Instead of a single database for managing configuration, many federated CMDBs fit together in the Configuration Management System (CMS), Service Asset and Configuration Management (SACM) and the Service Knowledge Management System (SKMS). In CMDBf, a Federated CMDB is designed to support this view of service management as an integrated system. To support the wide range of entities that appear in CMS, SACM, and SKMS, the items that CMDBf refers to are not limited to traditional CIs.

In addition to CIs, CMDBs also contain relationships between CIs. A relationship is a link between two CIs that identifies a dependency or connection between them. For example, applications may be linked to the servers they run on. Typically, IT services have many links to all the CIs that contribute to them. (See DSP0252, clauses 1 and 3.)

Q: What is federation?

A: Federation means some or all data from a number of sources can be viewed as coming from a single virtual source. The component databases of a federated database usually maintain independent control of the data they contain.

Q: Why does one federate data? What are the advantages and disadvantages?

A: Usually data is federated to provide a comprehensive view of an object or set of objects from a single access point. In the case of a CMDB, CIs that are desirable to appear in the CMDB are often managed by other applications with their own store of data about the CI. These applications are often called Management Data Repositories (MDRs). One of the goals of the CMDB is to consolidate data from various MDRs into the CMDB to provide a single virtual repository for managing configuration. This is an ideal opportunity for federation because federation allows the MDRs to maintain control of the data on the CIs they manage, but make the data accessible from the CMDB. (See DSP0252, clause 4.)

Q: What are the alternatives to federation?

A: In a limited way, CMDB centralization can be achieved by periodically loading data from the MDRs to the CMDB. This is often called Extract, Transform, and Load (ETL). ETL presents some problems because data stored in multiple places can become stale and out of synch between data transfers. Increasing the frequency of transfers or even initiating a transfer in real time with every change may diminish the stale data problem, but the overhead for transfer can be considerable. ETL systems often transfer data from the MDR that is never accessed from the CMDB. In this case, the system expense incurred to transfer data that is not used is wasteful. In a federated system, the CMDB queries the MDR for data when it is needed by the CMDB. (See DSP0252, clause 4.)

Q: When should federation be used?

A: Because federation avoids stale data and frequent ETL operations, federation is most suitable when the data in the MDRs changes rapidly and the CMDB must have access to the current values. When the MDR contains data on vast numbers of CIs but the probability that the CMDB will need to access any individual CI is low, federation avoids transferring large quantities of data that is never accessed.

Nevertheless, loading data may be desirable in some situations. Loading data incurs the overhead of the load into the CMDB, but the overhead for query to the CMDB is reduced because queries are always local. In situations where the data in the MDR remains relatively unchanged over time and the CMDB must access data from the MDR frequently, loading the MDR data may yield better performance with fewer resources. Loaded data will also perform faster when many complex queries involving comparison of data from several MDRs must be executed. A narrow bandwidth network connection between the MDR and the CMDB may favor periodic loads over federation, although this approach usually requires careful analysis of the circumstances. (See DSP0252, clause 4.)

Q: How much data should I federate?

A: The amount of data to federate depends on the services to be supported and the level of configuration control desired. ITIL documentation uses the example of computer keyboards. Most of the time, keyboards are a commodity that is not tracked. But at the United Nations, where the language of keyboards affects the business of the organization, keyboards are tracked.

The only guideline for what data should be included is business relevance. Usually, a CMDB contains only CIs that are subject to the change control process. For example, servers in the datacenter almost always appear in the CMDB because they are under change control. If workstations and laptops are not change controlled, they often do not appear in the CMDB. Placing a CI into the CMDB causes additional cost. If there is no return on that cost in the form of more efficient IT services, there is little reason to include the CI in the CMDB.

When deciding whether or not to federate an MDR, the same considerations apply. If the data in the MDR is likely to affect IT services, the MDR is a good candidate for federation. (See DSP0252, clause 4.)

Q: What is CMDBf?

A: CMDBf is an acronym that stands for Configuration Management Database Federation. The CMDBf was an industry consortium formed in 2006 to develop a specification for CMDB federation. The consortium was not affiliated with any standards organization. The CMDBf published a specification in October 2007. The CMDBf consortium then donated the specification to the DMTF, and the CMDBf consortium was officially dissolved. The DMTF CMDB Federation Working Group was then formed to shepherd the consortium specification through the DMTF standards acceptance process. The DMTF published the standard as DSP0252 in June 2009. Since the dissolution of the consortium, CMDBf is often used as an abbreviation for “CMDB Federation” and DSP0252 is often referred to as the “CMDBf.”

Q: What problem is the CMDBf trying to solve?

A: The specification (DSP0252) is meant to support a federated CMDB that provides a single aggregate view of an IT resource, even if the data is from heterogeneous sources. Such a federated CMDB can support many scenarios, including

- Provide an accurate description of IT inventory from a combination of configuration and asset information
- Reflect changes to IT resources across diverse repositories and data sources
- Compare expected configuration versus actual configuration
- Enable version awareness, such as in the following examples:
  - Coordinate planned configuration changes
  - Track change history
- Relate configuration and asset data to other data and data sources, such as incident, problem, and service-level records. This category includes:

- Integration of change management and incident management with actual CI status and configuration information
- Analysis of root causes of incidents, problems, and SLA breaches and assessment of impacts of planned changes based on dependency analysis of incident, problem, change, and configuration records.

Q: Is CMDBf an official standard?

A: CMDBf, *Configuration Management Database (CMDB) Federation Specification* (DSP0252), is an official standard of the DMTF.

Q: What are the architectural components of CMDBf?

A: The main architectural components of a CMDBf implementation are the Federating CMDB and MDRs, as shown in Figure 1. The MDRs and the Federating CMDB communicate through Registration and Query web services. These services are defined in DSP0252. A Federating CMDB can itself be an MDR, which allows for hierarchical federations of federated CMDBs. In addition to the Federating CMDB and MDR, the architecture describes a Client, an entity that can query MDRs and Federating CMDBs. (See DSP0252, clause 5.)

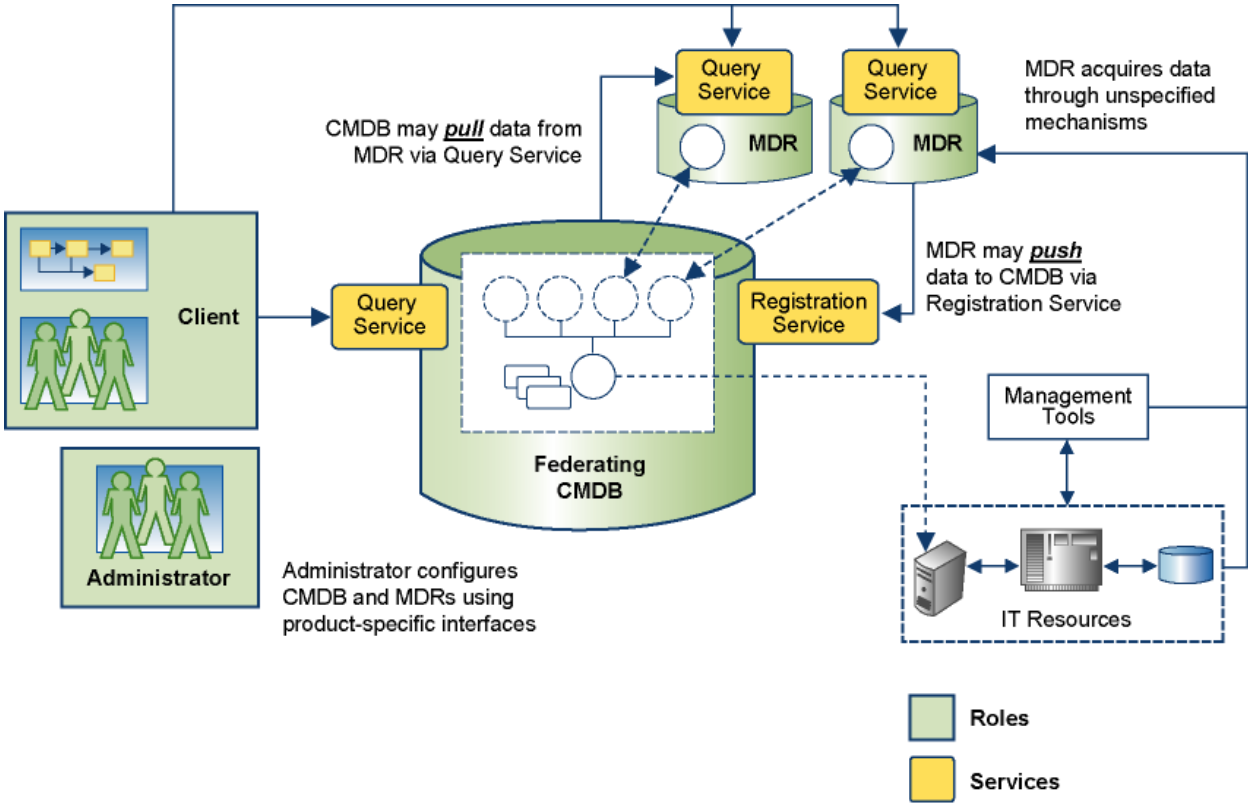

Figure 1 – Archictectural Components of CMDBf

Q: Is CMDBf an architecture or a piece of infrastructure?

A: CMDBf is a standard that includes an architecture. The software and hardware used in an implementation of the CMDBf standard is part of the enterprise infrastructure. (See DSP0252, "Introduction".)

Q: What is the difference between "Federating CMDB" and CMDBf?

A: In common use, the term CMDBf refers to the DMTF standard, *Configuration Management Database (CMDB) Federation Specification* (DSP0252). A prominent feature of DSP0252 is the Federating CMDB architectural component. The generic term "federating CMDB" without any qualification can refer to a CMDB with federating features that does not necessarily follow DSP0252. (See DSP0252, "Introduction".)

3 MDR Implementers

Q: What is an MDR?

A: MDR is an acronym that stands for Management Data Repository (see clause 4.6 of DSP0252). In general, an MDR is a data repository that is external to a CMDB and contains data that would be useful to integrate into a CMDB. The compelling value of CMDBf is that it provides the interfaces for incorporation of MDR data into a CMDB.

Q: Does an MDR need to be implemented using a relational DBMS?

A: No, any data source that is capable of implementing the CMDBf Query Service, and optionally a client that invokes the CMDBf Registration Service, may be used as an MDR. No assumption is made about the kind of technology that implements an MDR. It could be a relational DBMS, an XML file, or a data stream generated from a management application.

Q: What needs to be done to plug my MDR into a CMDBf-compliant CMDB?

A: To plug an MDR into a CMDBf-compliant CMDB, follow these steps:

1. Identify data to federate.
2. Decide how a CMDB will learn about the available data in the MDR.
3. Decide which items and relationships to make available through a Query Service and which to register.
4. Configure the Federating CMDB and/or MDRs to recognize each other.

Q: How does a Federating CMDB learn about the available data in an MDR?

A: A Federating CMDB learns about an MDR's available data in the following two ways:

- The MDR informs the Federating CMDB about resources of interest and when they change. This is called "push mode" because the MDR pushes information to the Federating CMDB. The Federating CMDB may also query the MDR for more detailed information about these resources.
- The Federating CMDB, either periodically or on demand, queries the MDR's data or data that has changed recently. This is called "pull mode" because the data in the MDR is pulled into the Federating CMDB. In a variant of pull mode, the Federating CMDB maintains no data within itself except metadata describing the types of data each MDR contains.

Q: What is the difference between push-mode and pull-mode federation?

A: Push mode registers instances with the Federating CMDB by the MDR invoking the Registration Service. Pull mode allows instances to be “pulled” from the MDR when a query operation is performed. Depending on the pull-mode implementation, the Federating CMDB may periodically poll the MDRs for data of interest or it may wait until a client requests data.

Q: Do I need to implement push-mode or pull-mode federation?

A: You must implement either push-mode or pull-mode federation, or you can implement both.

Q: What is required in push-mode federation?

A: Push-mode federation requires that MDRs invoke the Registration Service exposed by the CMDB. This means that the MDR has a means to actually “push” instances of data to the Federating CMDB.

Q: What is required by pull-mode federation?

A: Pull-mode federation requires that MDRs implement a Query Service. The Federating CMDB determines when to query the MDR data.

Q: How often should an MDR push data to the Federating CMDB?

A: The specification leaves the question to the discretion of the implementation. For example, some MDRs will push data in near real-time updates, like sending a discovery event; other MDRs will implement different policies, such as periodically pushing data at a frequency, such as once per minute, once per hour, etc.

Q: What does it mean for the quality of the federation when an MDR only supports push mode?

A: Using push mode will necessitate that the actual push operations be done at some frequency. If this frequency is occasional, such as once per night, the data in the CMDB may become “stale,” that is, the data might not correspond to what is actually in the corresponding MDR if the MDR changes between push operations. If the pushes occur in near real-time (that is, the CMDB is notified as soon as relevant instance data is added, modified, or removed), then the choice of push versus pull mode has little effect on the data quality. Depending on factors such as the rate of data change and the frequency with which a given MDR is accessed, the resources required to maintain the federation may vary.

Q: What are the advantages and disadvantages of push mode and pull mode?

A: Push-mode federation has the advantage of simplicity and improved query performance (because queries are often partially or completely processed within the CMDB's local data store, rather than being distributed to possibly several MDRs). Push-mode federation has the disadvantage that data may be pushed to the CMDB whether or not it is needed.

Pull-mode federation‘s advantage is that data is transported only when a client queries it or a CMDB wishes to refresh its cache. Pull-mode federation has the disadvantage that queries may take longer to execute, due to the performance cost of gathering data from remote data sources.

Q: What is the CMDBf Registration Service?

A: The Registration Service is an interface implemented by a Federating CMDB and invoked by an MDR that is used to notify the Federating CMDB about resources available for federation. The Registration Service allows instances of data to be “pushed” to a Federating CMDB. After the instances have been added to the CMDB, they can be used to resolve CMDBf queries made to the Federating CMDB. The federating CMDB may also perform some type of reconciliation during registration, but this is not required or defined by the standard.

Q: In the Registration Service what gets “pushed” to the Federating CMDB?

A: In the Registration Service, a set of items with their associated records is pushed. At a minimum, properties of the items that are being identified should be pushed. However, the standard intentionally neither limits registration to identifying properties nor requires that identifying properties be present. Other properties that are determined to be useful in the Federating CMDB may be part of the content that is registered.

Q: In the Registration Service used by push-mode federation, does the MDR initiate the operation or does the Federating CMDB?

A: In push-mode federation, the MDR initiates the operation by invoking the Federating CMDB's Registration Service.

Q: What is an MDR ID, and how can I determine my MDR ID (so that I can register my instances)?

A: An MDR ID is a unique identifier, perhaps globally unique, but at least unique among all the connected Federating CMDBs and MDRs. DSP0252 does not specify a means by which an MDR can determine its own ID. As such, this information could be communicated in a number of ways: through a configuration property, through some out-of-band mechanism, through a user interface, or through an API not defined in DSP0252.

Q: How will a Federating CMDB identify my MDR’s data model?

A: The data model of an MDR can be communicated to a Federating CMDB in a number of ways. Clause 8 of DSP0252 defines metadata. It is an optional feature of the standard (that is, metadata may be communicated in ways not specified in the standard or not at all in a standard-compliant implementation).

Q: What is the CMDBf Query Service?

A: The Query Service is an interface implemented both by MDRs and Federating CMDBs. It is used to query the data in the MDR and Federating CMDB. The Query Service includes operations to query instances of data and provide graph queries, allowing for the retrieval of instances that are linked together with relationships.

Q: Is the CMDBf Query Service the same as a SQL database query service?

A: The Query Service serves a similar role as a SQL database query service, but the CMDBf Query Service does not depend on the use of SQL or an RDBMS, it explicitly provides a way to aggregate data from multiple repositories, and it contains special support for graph queries. GraphQuery is the feature that most notably distinguishes CMDBf Query Service from SQL. A GraphQuery request describes the items and relationships of interest in the form of a graph. Constraints can be applied to the nodes (items) and edges (relationships). The GraphQuery response contains the items and relationships that, through their combination, compose a graph that satisfies the constraints of the query. (See DSP0252, clause 6.)

Q: What are the forms of CMDBf queries?

A: The two general types of CMDBf queries are property/value and XPath.

Q: Does a Query Service require the use of XPath?

A: XPath is an optional capability of the CMDBf Standard Query Service. If XPath is used, it should be used according to the guidelines provided in clause 6.5 of DSP0252.

Q: Does an MDR use the Registration Service to register itself with the federating CMDB?

A: No, the Registration Service is intended to register items and relationships, not MDRs.

Q: Isn't the Registration Service really a "synchronization" service rather than a "federation" service?

A: The CMDBf Registration Service is a synchronization service and, consequently, some might argue that the CMDBf is not a pure federation system. However, even a strictly push-mode CMDBf system has a strong element of federation. The primary intention of the Registration Service is to register the *identity* of items or relationships, so that extended information (which may change much more frequently than identity information) may be obtained directly from the MDRs using the Query Service when the information is required. Implementers have the option of building a strict pull-mode CMDBf system without implementing a Registration Service if they wish to avoid synchronization entirely.

4 Federating CMDB Implementers

Q: What is the difference between a CMDBf-compliant Federating CMDB and my current CMDB?

A: Most CMDBs provide a record of configuration items and relationships in an IT environment. Implementing CMDBf makes a CMDB interoperable with clients and MDRs that implement CMDBf, thereby enabling access to data from sources not managed within your current CMDB.

Q: What do I need to do to make my implementation CMDBf compliant?

A: Implementing all the required parts of the CMDBf standard (DSP0252) — usually indicated in the standard specification with the word "must" — qualify an implementation as CMDBf compliant.

Q: What is the basic data organization of the CMDBf?

A: CMDBf organizes data using three data wrappers: Item, Relationship, and Record. Item examples include a computer, application, service, building, and incident record. A Relationship is an Item that represents an association between two Items. Each Item or Relationship aggregates any number of Records of any mixture of types. Each Record is a wrapper around data; the wrapped data uses a format selected by the implementation. Clause 5.5 of DSP0252 describes these elements in more detail.

Q: How does a Federating CMDB know which MDRs to federate?

A: The CMDBf Standard does not specify how to determine which MDRs to federate. The two basic styles are to either configure the MDRs to initiate communication with the Federating CMDB's Registration Service (known as push mode), or to configure the Federating CMDB to initiate communication with each MDR's Query Service (known as pull mode). Such information could be specified in a number of ways, including by manually configuring each repository, reading the service addresses from a property in some configuration file, and looking up the information in a service registry. In DSP0252, clause 5.3.2 describes federation modes and clause 8 discusses the use of metadata to describe services.

Q: What is a federated query?

A: A federated query is a query invoked against one repository (the Federating CMDB) that may retrieve and combine data from one or more other repositories (the MDRs).

Q: Must a Federating CMDB implement a federated query?

A: A Federating CMDB must implement the CMDBf Query Service for data contained within its own local repository. It may optionally extend that query to data contained in other MDRs. The Federating CMDB may support a federated query for only a subset of the data available in MDRs.

Q: Can a Federating CMDB be federated by a different Federating CMDB?

A: A Federating CMDB may both federate other MDRs and be federated itself by another Federating CMDB, to which it appears to be an MDR. Two Federated CMDBs may federate each other in a peer-to-peer configuration, or one may federate the other (which in turn may federate others) in a hierarchical configuration.

Q: What is the relationship between roles and services?

A: The active roles are MDR, Federating CMDB, and Client. The services are registration and query. Each role may implement and/or use services. Specifically, an MDR may use the Registration Service and it may implement the Query Service. A Federating CMDB may implement both the Registration Service and the Query Service, and it may use the Query Service (to implement a federated query). A client may use the Query Service. In DSP0252, clause 5.2 describes roles and clause 5.3 describes services, including which services apply to which roles.

Q: How do I uniquely identify my items and relationships?

A: Each Federating CMDB and MDR should have a globally unique identifier or, at the least, an identifier different from any other repository that is part of the same system of federated repositories. Each Federating CMDB and MDR assigns an ID for each item and relationship that is unique within the repository. The combination of the repository ID and the item or relationship ID will be unique within the system of federated repositories. Clause 5.5.2 of DSP0252 describes the formation of IDs for items and relationships, and it also describes record IDs.

Q: How do I avoid collisions in item identification for items originating from different MDRs?

A: Collisions in item identification are avoided by combining the unique repository ID with the local ID that is unique within the repository. Clause 5.5.2 of DSP0252 describes the formation of IDs.

Q: What is the relationship between CMDBf and the DMTF Common Information Model (CIM)?

A: The DMTF CIM does not address the domain of a federated CMDB. Therefore, there is no relationship between the CIM and CMDBf. However, the content of CMDBf-conformant Federating CMDBs and MDRs may be based on the CIM.

Bibliography

DMTF DSP0252, *Configuration Management Database (CMDB) Federation Specification 1.0*, http://www.dmtf.org/standards/published_documents/DSP0252_1.0.pdf

ITIL® V3 Glossary, *Glossary of Terms, Definitions, and Acronyms*, v3.1.24, 11 May 2007, http://www.best-management-practice.com/gempdf/ITIL_Glossary_V3_1_24.pdf

ITIL® V3 Service Transition, 05 Jun 2007, by Sharon Taylor, Shirley Lacy and Ivor Macfarlane. The Stationery Office. ISBN 9780113310487.