WHITE PAPER: SERVICE MODELING AND CMDB DESIGN

IT Service Modeling for the CA CMDB

JANUARY 2014

Brian Johnson, Malcolm Ryder and John Sorensen

CA TECHNOLOGIES

Table of Contents

Executive Summary

Challenge

Configuration Management Databases are designed and built to centralize availability of an authoritative description of the configuration of a company's IT services being used internally or by its external customers. But traditional data identifying IT infrastructure are often unable to offer a CMDB the type of description needed to explain what makes the structure of a service generate or host the operational functionalities appropriate for users. To assure that the service is supportable, technical descriptions need to translate into language and definitions that make sense to service stakeholders, including affected parties whose own familiarities are mainly outside of IT. Their various perspectives make finding a common language for the CMDB quite difficult to achieve, if not speculative.

Opportunity

A standardized technique for modeling a service provides a reliable guide to logically select the information that is sufficient to explain service viability. This model serves as a blueprint for navigating the service internals, and for recognizing where the service has health and performance dependencies on its particular parts and connections between parts. By using a consistent technique for preparing the model, information about the service becomes more selective, more regular and more accountable as a shared resource across different types of management affecting the service availability and impact for users. The guidance of the model reduces trial and error in determining and storing the information in a CMDB, and it increases the quality of the CMDB.

Benefits

The CMDB itself supports a higher quality of management communication because of the strategic selection of data it provides and the consistency with which that data can be verified and maintained. Visibility of service structures allows faster design and delivery of additions, enhancements or modifications that improve the infrastructure underpinnings of service performance. Confidence in the service construction gives service providers the ability to warranty its utility and functionality. User confidence in services encourages them to use services as the building blocks of their own operations, effectively basing their performance as users on a more predictable level and quality of support. Predictability decreases decisions that lead to unjustifiable risk, or to waste, lowering the unwelcome costs associated with both.

Section 1

Modeling Definitions of Services in the CA CMDB

Introduction to Service Modeling

This document outlines numerous critical ideas on how to approach service modeling, CMDB design and population of a CMDB. The approach is adjusted in order to ensure that it can be effective in a pragmatic way with the CA CMDB design, and it is influenced by the capabilities, but also the current limitations, of CA CMDB.

A key reference for this approach is the 2006 Forrester paper, “The "Just Enough" CMDB”, in which the executive summary reads: “The convergence of more structured management processes, such as ITIL, and technical innovations like application dependency discovery has brought configuration management and the concept of the configuration management database (CMDB) to the attention of many enterprises. This holds the promise of a more rational and efficient management of IT operations, which is, in many cases, a major element of IT budgets. While most organizations have bought into the benefits of the CMDB and business service management (BSM) approaches by now, implementation is, as always, a sore spot. Corporations are reluctant to embark on a broad, sweeping strategy that may take years to implement and wonder if a more tactical and process-driven approach would be more effective. As many IT operations still function in firefighting mode, with a fragmented product set, using a top-down approach that focuses on the most critical issues first may be the best and quickest way to reap the benefits of a service management solution.”

In this paper, the top-down tactical approach is interpreted as a pragmatic approach to information management. The key objective is that the necessary set of information is designed to be lean, addressing how consumers of the information require it to be represented according to explicit use cases. This approach will need to accommodate two big challenges: continuing addition of new information over time; and continuing expansion of the range of information found necessary to include for new consumers. These challenges are usually driven by ongoing change in the business itself, and by technology evolution in the IT infrastructure for operations.

To tackle these challenges, this discussion will run along a two-pronged attack.

First, best practices and concepts as outlined in ITIL v3 inform this approach. In ITIL v3 guidance on Service Asset and Configuration Management, an overall service knowledge management system (SKMS) includes a configuration management system (CMS). The CMS then includes the configuration management database or CMDB, which is maintained by a **configuration management process**.

Second, the **primary use cases** for the CMDB are typically to determine the root cause of a service interruption or degradation, and to forecast probable impact of a change proposed to an item that may be a structural element of a service. The two use cases give operational context to the modeling and design approach involving service definitions and CMDB structure.

ITIL v3 also includes the concept of federating CMDBs. As a design concept, federation of data from multiple management data repositories (MDRs) reinforces attention to minimizing

unnecessary duplication in the CMDB of information that is already contained in MDRs, while further reinforcing selectivity in identifying any MDR as a contributor of CMDB information.

Finally, this discussion leans on field work reported in educational and conference venues to customers and staff of CA Technologies by John Sorensen, a co-author of this paper.

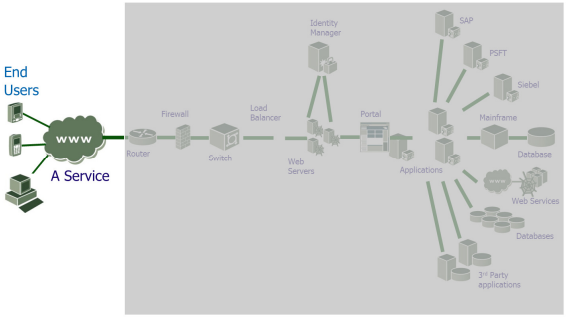
The Business View of Service Structure

What is a Service?
According to ITIL v3 a service is defined as "A means of delivering value to Customers by facilitating Outcomes Customers want to achieve without the ownership of specific Costs and Risks." The key point is a clear separation between the responsibilities of the consumer and provider and their mutual independence. For the consumer, one service can be replaced by another from a different provider, and a Service Level Agreement (SLA) provides a foundation for measuring the performance and quality of the service. In light of the two primary use cases, performance is primarily a matter of the user experience of the service if it is impacted (positively or negatively) by a change or if demonstrably it has sub-par behavior or has been interrupted.

If you're a service provider, the term Customers (or consumers) can here refer to a company's internal business users, but it can also refer to that company's external customers.

When we say 'services' we often mean IT operations such as hosting, telecom, and the like. However, it is important to recognize the difference between the business customer's view of that, and the provider's view. External providers who offer these services to another company consider them to be **business services** provided to the receiving company, whereas within that customer's organization those deliverables are seen as **IT services** underpinning the customer's own business operations by which the company offers as services to its own clients.

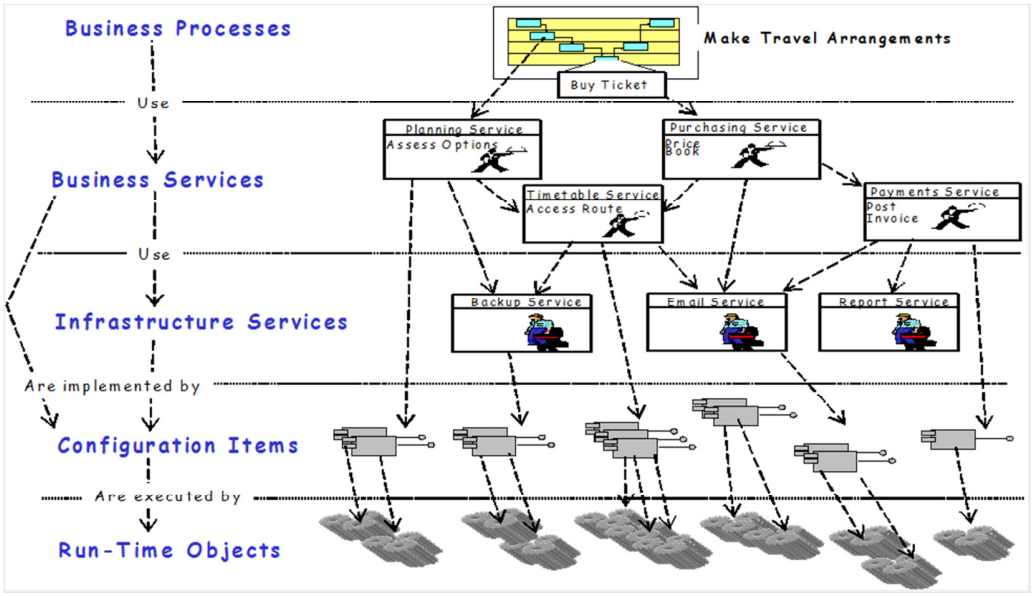
In either case, there is a scenario in which services may be subscribed to, from a catalog of offerings; the external provider has a services catalog for its customer company, and that company has a services catalog for its own clients. In this scenario, actual service users don't really care about how the service is being delivered as long as it "lives up to their expectations", and that attitude will always prevail. For the users, the question is whether there is any enforcement of their expectations, such as through a formal agreement (often referred to as a contract or a service level agreement). The delivery system that provides the service (from right to left in the illustration below) is largely obscure and, unless it stops working, irrelevant.



For our purposes, ITIL v3 defines the following: Service, IT Service, Business Service and Infrastructure Service. Business Services and Infrastructure Services are subsets of IT Services. IT Service is a Service provided by IT.

An example of the ITIL view is seen below in the diagram describing the overall supporting mechanism for an “everyday” capability – a business system that allows someone to buy a ticket for travel. We can refer to that capability as a business process that then uses some business services, for instance a planning service and a purchasing service, those relying in turn on other business services like timetable publishing services of the different airline companies and payment services.

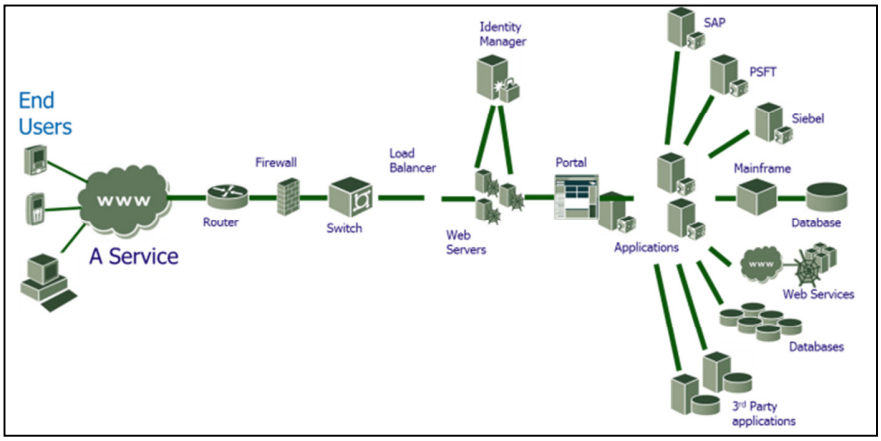
Underlying those business services, we also find infrastructure services, some of these being visible to end-users like the email and report services, where as other infrastructure services may not be visible to end-users, for instance data backup services.



In order to provide an overall model of the design of the ticket-buying mechanism, we represent different parts logically making up the service, along with the relationships that allow those parts to work together such that the service becomes operational. Seen as objects, those parts with their relationships are diagrammed in what we call a service model. The intended successful experience of buying the ticket is the business expectation of the operation and use of this complex system.

Any internal IT organization is expected to “operate the IT infrastructure” in such a way that the business utilization of the infrastructure will meet business expectations.

Given that, the IT organization in the role of “service provider” must have comprehensive knowledge about how the underlying IT infrastructure causes a service to be delivered — in particular, about all the different infrastructure components required for this purpose (as suggested in the picture below):



As a provider, the IT organization needs to have a model of the delivery system that accounts for how it causes the service to appear for the customer in a way that the *customer* needs to recognize it. The business model of the service consists largely of requirements about what functions the service must serve, under what conditions, over what durations. But the service provider must have a model of the service that allows those outcomes of service utilization to be achieved and sustained through continual management of the operations by the provider.

Some examples of how the IT organization needs to think about services include the following:

- A low-level IT service such as DNS or DHCP
- A set of resources that you want to manage collectively, such as the servers or printers contained in a specific location
- A network service such as VPN
- A systems service such as Active Directory
- A database service such as a Microsoft SQL Server cluster
- A high-level business service such as Payroll, Email, and so on
- External services such as an Internet Service Provider or Cloud

Most services are a combination of all of the above, with high-level business services comprised of various low-level services. For example, a Blackberry business service may contain subservices that represent key IT services that support Blackberry communication, such as Active Directory, Exchange, DHCP, and so on. Summarizing, any set of resources that depend on each other to provide a useful business function can comprise a service. Identifying services requires an understanding of the resources that make up your enterprise and the typical contents of a service.

The service contents, as indicated above, include various elements that interact with each other in a prescribed way. The overall arrangement of items and relationships makes up the configuration of the service. As a result, we call these elements and relationships Configuration Items, but only because we recognize them as such *in the context of a presumed service*.

The configuration management process

Configuration management is focused on IT services and concentrates on the CIs that need to be managed in order to deliver a service.

That concentration is sustained in a specific manner. According to ITIL v3 configuration management is: “The Process responsible for maintaining information about Configuration Items required to deliver an IT Service, including their Relationships. This information is managed throughout the Lifecycle of the CI. Configuration Management is part of an overall Service Asset and Configuration Management Process.” Continuing, it states, “The term configuration item or CI refers to the fundamental structural unit of a configuration management system. Examples of CIs include individual requirements documents, software, models, plans, and people.”

The configuration management database (CMDB) stores “CIs” as defined above, making a critical distinction between managing objects and managing information about the objects.

“The Configuration Management System oversees the life of the CIs through a combination of process and tools by implementing and enabling the fundamental elements of identification, change management, status accounting, and audits. The objective of this system is to avoid the introduction [into the infrastructure] of errors related to lack of testing as well as incompatibilities with other CIs”

ITIL v3 also talks about modeling the business service (a type of IT service) in the configuration management system (CMS), most importantly including the relationships between the underlying CIs representing the IT infrastructure components. This is in order to describe how they contribute in delivering the business or infrastructure service. Thus, the configuration management process provides the ability to consistently identify, manage and verify which IT infrastructure components and component relationships are elements within the business service – with the CMDB being the authorized repository of the service information.

An important contrast to highlight here is the one with asset management, where you typically will have the broader attention on all assets owned by the company. Although CMDBs have been touted as “the single source of truth”, configuration management is focused on describing services, and is not about sticking every bit of information about the infrastructure into a monolithic single database or data warehouse. Qualifying and validating infrastructure as CIs will reduce the number of items that will fall under configuration management, even though configuration management itself might require retention of a large amount of information about each item. This information volume issue should be brought immediately under a defined scope through considering the use cases and the actual capacity for managing the quality of the data.

SECTION 2

Service Modeling with Operational and Business Data

Key use cases: analyzing root cause and change impact

The decisions about what information belongs in a CMDB are strongly dependent on the needs of the intended CMDB users.

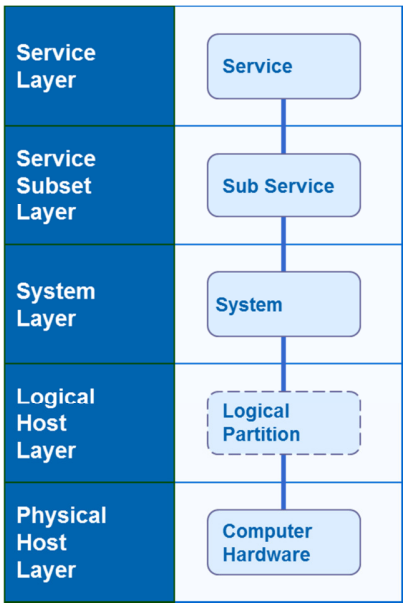
Probable users of the CMDB are varied by their responsibilities, and in a given organization the immediacy of their needs may not be all alike.

As noted earlier, the business view of a service is likely to be of something with a complex structure of dependency relationships between elements. From an engineering standpoint, some elements have dependencies on multiple other elements. As the total arrangement of interactions continues to be operationally managed, the service continues to be delivered to the user.

One key use case for constraining the modeling of a service is the need for support teams to zero in on a point of failure within the structure of the service. The after-effect of the failure, which is an interruption or degradation of normal use called an Incident, highlights the need for a model of the way the failed item is involved in the service functionality, not just in its construction. The failed item is the Root Cause. In accordance with ITIL v3, the item may be a component or a relationship, but we can assume that a failed relationship will be further attributable to a component. The question is, given the probability that infrastructure items are integrated into other items, how is the component itself defined?

The objective of the service model is to provide an understanding of the probable dependency of a service on technology.

Such a model is exemplified in this diagram of a (top level) Service dependency on (bottom level) Computer Hardware:



Note that the Service dependency can be based on several things (defined components) situated in between the underlying Computer Hardware and the Service, such as logical partitions of hardware, and operating systems running in the logical partitions. As we see how these underlying components combine (upwards) to make larger combinable elements, we see how the structure of the service is composed of dependencies. Included in that is the formation of sub services to contribute as key components of the ultimate service. This mode of representation groups items having similar responsibilities into a layer. The connection between the different responsibilities is the decisive issue in modeling the service such that an included item is understood in terms of its potential impact on the service. Under normal circumstances, the item can be relied upon by other items that are dependent on it. In abnormal circumstances, or if changes to the item itself are made, then there is risk that the item's expected impact on other items and on the service will be different. Parties responsible for solving service problems need the ability to trace the relationships between prescribed responsibilities, so that they can both deduce (diagnostically) and predict (synthetically or based on prior experience) the possible impact of changes when considering a given infrastructure item.

The minimum information needed in the model, which would then be stored in the CMDB, is the definition of the components in terms of their service responsibility, and the definition of the relationships that allow those responsibilities to interoperate.

Overall, a record of an infrastructure item would be meaningful in the CMDB if the thought process behind modeling has identified:

- the service to provide,
- the structural responsibilities of the service elements and relationships,
- and the criteria by which an infrastructure item is validated as being appropriate and active in the responsibility required within the service.

What is THE CMDB SCOPE?

The CMDB is intended to be the master repository of the prescribed and authorized configurations of services. If we cannot trust the information in a CMDB, there's no value in creating one in the first place, and the fastest way to lose confidence in the information is to lose control of it. CMDB quality will be determined by the IT organization's maturity level in sustaining the configuration management process. The design and implementation of a service model and a CMDB to store service models should correspond to the maturity level of the organization assigned to execute the configuration management process to maintain it.

The service model designed should call for only a scope of information that is initially right-sized but is also extendable as the maturity level of the organization grows.

Regarding maturity, an important distinction here is between the ability to acquire data versus the ability to use it effectively. For example, high volumes of data about infrastructure items are generated by monitoring and auto-discovery management tools, but these may not reveal the key evidence of which services are involved or how.

Broad and powerful item detection through such tools is significant because we need it to account for the real-world availability of components necessary to the integrity of the service.

This accountability is presumptively established by the records of asset management and systems engineering. For good reasons, those records are fairly exhaustive in their coverage, being concerned with all available resources, and auditing takes place to see if they are still current. In contrast to those inventories, configuration management needs to account only for the items constituting services... Therefore, the labor-convenience of auto-detection must still be just a contributor to the more important matter of selectivity.

The priority of relationship definitions in CMDB design

Use cases point out another important characteristic of the CMDB that distinguishes it further from an asset management repository. Namely, in the CMDB it is actually more important to populate the relationships between CIs than their attributes, as the typical use cases are mainly sensitive to dependencies and are relying heavily on this type of information. It's therefore even more important to consider which relationships per CI type are indicated by each MDR.

But when differing CMDB consumers and service stakeholders decide how to represent a service for their own respective needs, each party may initially rely on identifying types of CI associations that make sense in their own perspective but do not have exact or clear counterparts in other perspectives. This is a semantic problem of translation -- that is, interpreting whether different descriptions of things are actually describing the same thing or not.

To illustrate these different perspectives, the table below lists examples of different ways of referring to items and their associations, bringing up the question of whether apparent likenesses can be used meaningfully by multiple responsible parties,

Type of item	**How name of type is significant**	**Typical party of interest**	**Examples of when identity is specified**
Vehicle	Function	Shipper	Logistics
Automobile	Construction	Manufacturer	Development
Coupe	Supply	Buyer	Products
Racer	Utility	Driver	Events

In the table above, it is clear that each type of item, as named, easily can be synonymous with the other types; however, using the type name in communications is going to be more or less appropriate and helpful depending on who is the target audience.

In the table below listing item associations, a similar need for being appropriate and precise is evident particularly because of the possibility of superficial similarities and coincidences.

Type of association among two items	How association is established	Typical party of interest	Examples of how association is audited
Found with	Correlation	Process manager	Business rule
Belongs to	Allocation	Asset manager	License
Is member of	Assignment	Resource manager	Task status
Is part of	Integration	Systems manager	Blueprint

Whether dealing with item types or association types, the inherent risk is that a record will not be meaningful enough to correctly support decisions or analysis being performed by someone other than the party that is the data source.

This means that if a service is described by Party X with named items and named relationships, this particular service description may be correct on its own terms but it still may not be useful to some other Party Y.

In modeling a service, the sources of descriptive data can include Operations, Finance, Engineering, Support, and other distinctive domains that each contribute at a management level to the presence and utility of a service. Consequently, they are all busy tracking some aspect of a service, with the possibility of being a decision-maker (e.g. planner or approver) about the required provision or performance of the service.

The critical understanding needed here is that a good service model is not a lowest common denominator amongst multiple stakeholders. Nor is it a collection or union of all the different ways of describing things. *The most important problem to solve, therefore, is how to identify the service and its elements in a way that the context of the description is also evident to users of the description.*

In our discussion, the primary contexts are two use cases: Root Cause Analysis and Change Impact Analysis. By identifying the responsible parties in those two use cases, we identify the audience for the information provided by the service modeling, and in turn we determine the needs for supporting the interpretation of the information across these parties. The focus in both cases is on being able to trace the relationships between CIs with confidence in what particular service is the operational context of the CIs.

Management Data Repositories (MDRs)

Best practice in service modeling and CMDB design is therefore to assure that the relationships that position CIs within real operational services are not lost in translation when service models are referenced across different domains.

Logically, relationships make sense depending on the nature of the CIs that they associate, so the design of the information records in the CMDB must provide for assuring that CIs themselves are represented in an unambiguous way.

This includes allowing different parties, for example a systems manager and an asset manager, to correctly indicate the same CI when they need to share responsibility for affecting the item represented by the CI. But we should expect that different parties will bring, from their MDRs, their own respective information about CIs into their discussions and activities. So, *in order to preserve relationship information across different views, and thereby make the same service reference-able from different viewpoints, the CIs themselves must first be defined with attributes that can always be reconciled between different MDRs.*

During the initial service modeling the priority is to think about which trusted data sources, aka. Management Data Repositories (MDRs) exist in the environment in order to provide data to populate the attributes and relationships required per CI type into the CMDB. Here, again, the emphasis on the major use cases should inform the decision about what MDRs are necessary.

A related challenge in the CMDB design is to decide whether the selected MDRs are leveraged merely to initially supply the CMDB with "baseline" CI attributes and relationships, or whether these same MDRs will continue keeping that CMDB information up-to-date going forward. This has to be evaluated and agreed on per CI type.

At that point in planning, it is also necessary to agree on which CI attributes and relationships will have to be entered and kept up-to-date manually in the CMDB instead of automatically.

MDRs can also introduce technical restrictions or procedural issues that will have to be taken into account. For instance, if the required standard view of a given type of CI means there is a need for federating and composing data from multiple MDRs, how do we make sure that we can correctly coordinate (consolidate) the data from the different sources?

In modeling, the identification of CIs must drive specifications that create both a definition of the CI type, and an ability to distinguish every deployed instance of that type in a proposed or live service. This specification becomes the set of record attributes and object classifications in the CMDB.

In the CA CMDB implementation, the **Common Object Registration API (CORA)** plays an important role in ensuring a singular reference to each unique CI. For CORA to do this, it's important to plan and agree on the minimum set of attributes that together indicate to all parties the specific CI. Reconciliation, then, means the ability to recognize that some CI information from one source is indicating a unique particular CI that is also represented in another source's information.

The defining attributes distinguishing the CI are the keys for reconciliation. These so-called CORA attributes makes reconciliation easier to automate across MDRs. Included in this process are CIs of a type for which composite descriptions must always be assembled from

multiple MDRs. It's therefore important to agree on naming conventions, especially for CIs representing instances of applications and databases. It is also necessary to include sufficient amount of information in at least one CORA attribute to ensure the uniqueness of the CIs.

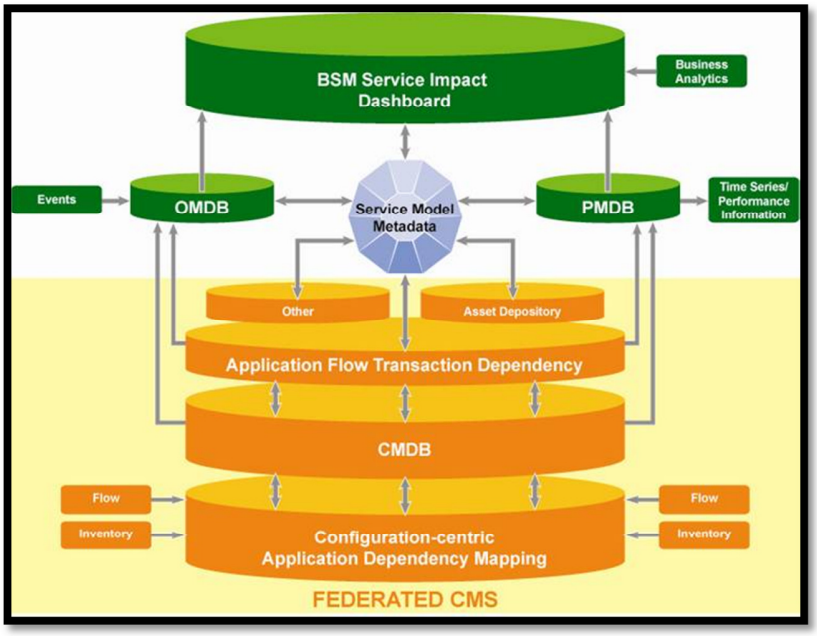
However, because IT shops may use many MDRs from many different sources or vendors, we cannot just assume that every MDR will include the needed attributes.

- A given MDR can usually import and export information specific to a given CI, but that information may not include the key **attributes** required for proper reconciliation with other MDRs.
- The CMDB, meanwhile, assumes that MDR information will be imported into the CMDB if it can be **reconciled** along the way.
- Additionally, information accuracy is critical to the CMDB's purpose, so the incoming information from multiple sources must be synchronized for providing only accurate information on time. Therefore proper planning and design of the **import/synchronization** mechanisms is required.

To ensure that the CI information is correctly reconciled and synchronized before the CMDB accepts it, CA CMDB includes functionality to handle ambiguous CIs. A powerful tool for adjusting the incoming information is the **Transaction Work Area (TWA)**, used to pre-load information in a staging area for inspection and revision before final data recording in the CMDB.

But the IT organization's resourcing (time, knowledge and labor) will impose practical limits on the TWA approach. Consequently, further automation has been developed, based on some basic assumptions about how infrastructure descriptions can be standardized at an industry-wide level and brought into the IT organization to be leveraged. *The availability of this automation and these standards is a critical design factor, since it is the recommended path for arriving at a mature modeling and CMDB solution.*

Speaking generally, in today's complex IT environments and fast-changing business operations, effectively addressing the many types of elements that make up services means that MDRs need to interoperate through a singular reference model of service and CI attributes.



This naturally leads us to anticipate the overall configuration management system, or CMS, as consisting of multiple MDRs and potentially multiple CMDBs. The diagram above by the analyst group EMA shows integration of data from many sources, in a federated service model (CMS). Note that there are at least three different management perspectives indicated: performance, operations and applications. In this EMA view, applications are an IT service, and the CMDB should record the model of that service, allowing the visibility to the CIs and relationships needing to be traced for analyzing root causes and impact of changes.

In a CA CMDB context, the CMDB is also considered to be deployable as a federated CMDB (or CMS) where there may and probably will be multiple MDRs to consider as trusted sources of attribute and relationship data. As mentioned earlier, for each of the CI types within scope it needs to be outlined which MDR is the trusted source of information about each given attribute of a type of CI and each service-enabling relationship to other CI types.

It's also worth noting that there are different rules and mechanisms that may be used to federate the information into the CA CMDB. Relevant to the above picture, they include:

- the **standards** agreed on by the CMDBf working group, where CA Technologies was a founding member,
- the traditional **federation adapters** that have been included with the CA CMDB product since its first release.
- and finally, **Catalyst**, which is the unified service model information bus that CA Technologies internally developed, which all solutions can sign into in order to contribute and consume the information about the CIs that they need or have. Catalyst is implemented to ensure that information can be communicated to and from this reference in the same way for all MDRs. The reference itself is called the **unified service model (USM)**, discussed in the next section.

The Unified Service Model

As noted earlier, federation presupposes that an MDR can send and receive information in comparison to the same reference used by other MDRs. The reference provides a uniform modeling language for CI and relationship identification.

While CORA provides a minimum amount of coverage in this regard, it is usually necessary to reach beyond CORA where more attributes are required to adequately describe the CI or service, especially from the business view or for items that are not hardware-based.

In CA CMDB, the default attributes defining a CI "type" are based on its classification in a **family**. Each family has a set of attributes used for distinguishing it from other families, but also for tracking metrics about the CI from a delivery and support point of view. Recording CIs requires using a family classification with every CI.

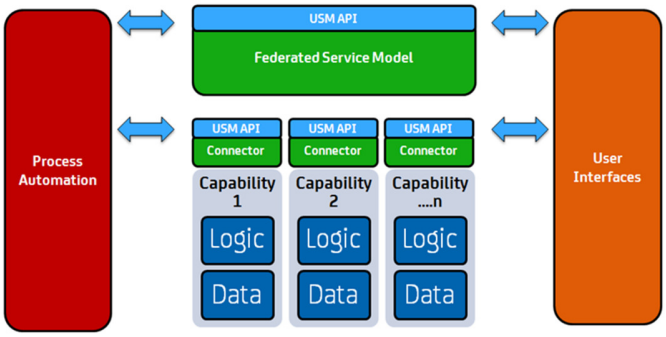
In our discussion of "MDRs" we are commonly referring to item managers or monitoring solutions, known as element managers, which identify many of the same items that may become known as CIs. These element managers have a whole host of attributes that they track - as appropriate for their own respective responsibilities.

Element managers are a critical source of information about what items exist in the infrastructure. But as pointed out already, there is a need to track a common set of attributes that makes each individual item (CI) unique in the CMDB as well as identifiable in the element manager. Additionally, in order to define a service with the CIs, relationships of CIs need to be represented.

In order to ensure reliability of service references across management domains, CA Technologies has defined and produced the unified service model (USM). The USM enables multiple management systems to share a common model of any service that they concurrently affect, which in turn allows different management systems to communicate states of that service into a unified **view** and into orchestrated **processes** across the systems.

The USM has been designed so that the data and capabilities of MDRs can be integrated systematically:

(i) at the data level (bottom) to collect information for a federated service model;
(ii) at the process level (left) through process automation; and
(iii) at the presentation level (right) through cross-product user interfaces:



Through the USM mechanism, an MDR can be selected and used as a trustworthy repository for the CMDB to reference, while the CMDB itself is an authoritative repository of information for sharing across multiple systems and management processes, especially including Incident Management and Change Management.

Given the data communications capability:

- If managed for CMDB federation, the MDRs can populate the CMDB and verify that the CI configuration and relationships as stored in the CMDB are being kept up-to-date.
- Some MDRs can be both contributors and consumers of information in the CMDB.

For communicating information about CIs to the CMDB, key attributes must be included and mapped to designated families.

For referencing CIs from the CMDB, unique identifiers of the instances of any given CI type must be provided for matching against item identity data in the MDRs.

However, in the CA CMDB implementation, searching for CIs in the CMDB has the additional requirement that a further CI categorization, **Class**, be provided.

A CI in any given Family has a further categorization of Class in order to distinguish how the CI type (family) is filling a role-based position in the service structure. For example, hardware in the form of a computer (Family) may perform as a database server (Class). This is consistent with the idea that the model of a service is primarily needed to account for the prescribed behaviors of the service, therefore the functional dynamics within the service are more important than what the form of the CI instance may happen to be.

In terms of the major use cases, it is certainly possible that the form of a CI will surface as the key factor in an actual or potential mal-function of the service. For that reason, *Classes are always logically associated with Families even though they are actually a different term and criterion of categorization.*

Supporting the Use Cases

Thinking in terms of the use cases, the important distinctions are always ones that allow us to identify things that can be individually modified for the purpose of restoring the service or decreasing any structural risks.

In both Incident and Change Management processes, in order to understand how a service can be delivered by all the infrastructure component instances, we are required to describe how all these different infrastructure components are inter-connected or inter-related specifically in order to deliver the service to the end-user.

Any service that we have chosen to initially work on will turn out to have a certain associated underlying IT infrastructure and interrelationships. One issue is to decide how many layers of functionality (what precision of granularity) we'll need in order to distinguish and describe the important distinctions and connections of the components.

Meanwhile, different services, having different components and dependencies, may turn out to require different types and numbers of layers to be identified and included in the model, *to satisfy the needs of the intended use of the model*.

It's therefore important to create a model of a service in an iterative manner that is sufficiently open and flexible so that, when additional distinctive information about the service is discovered to be critical to the use cases, more layers can be added into the hierarchy if and when they are required.

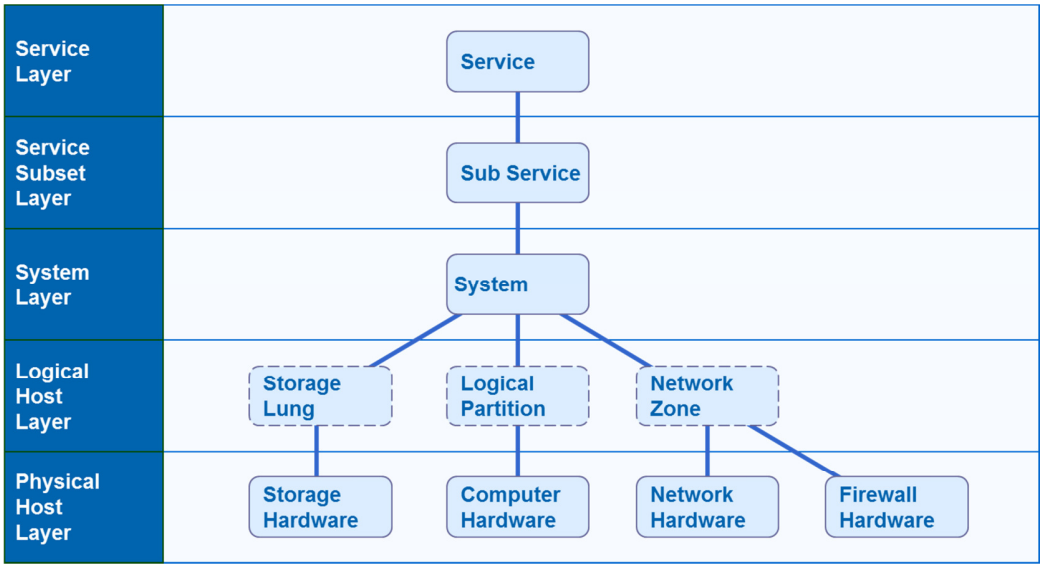
Over time, the discipline of creating service models adds more knowledge to the overall vocabulary for describing service types, particularly new ones differing from ones that have previously been modeled. For a CMDB, this large and growing vocabulary of reference-grade descriptive information needs to evolve in a way that is standardized, and used in a way that is highly consistent.

- As a reflection of this, the USM researches and incorporates industry standards for service component descriptions;
- Meanwhile, within one company's modeling efforts, the initial modeling needs to rely on consistent logic reflecting how things are going to be organized to be manageable. This is due to existing management disciplines being the most likely suppliers of CI data. The first available data supply may be driven more towards standards by decisions to use newer mechanisms or by data transformation tools.

In an iterative approach to the design of the model, there are no pre-requisites about ensuring that we have the right number of layers in a descriptive hierarchy or have labeled the layers with finality before proceeding. Pragmatism is the key feature, aligning with management capabilities and management maturity levels. As capabilities and maturity increase, the models can be revised to include more precision through updated definitions of components, relationships and layers.

By following this approach the model will reasonably accommodate future discoveries and decisions.

As soon as one gets started on this iterative approach (where subsequent drafts refine or extend earlier ones) and try to map more services, more CI types and/or layers may be selected for inclusion in the hierarchy, as demonstrated in the lower layers of the hierarchy below:



However, the design of the service model is expected to drive the definition of the information that is stored in the CMDB. Since continual modifications of the model would trigger the need for corresponding modifications of the CMDB, it is important to coordinate model and CMDB modifications with a reasonable business justification for the level of effort required.

The simplest way to accommodate future modifications is to assume from the beginning that they will occur, and to use standardized definitions of things, obtained from a strong reference, as the vocabulary to identify additional findings and inclusions. This will minimize the need for re-engineering data that has already been recorded from prior models.

SECTION 3

Service Model Information in the CA CMDB Context

Defining Service Model Information in CA CMDB Context

The importance of modeling the service description information is to ensure that the CMDB can function as the central point of reference across the group of management stakeholders responsible for success in service delivery and continuity. This calls for explicit agreement on who owns the configuration management process in the organization.

It is important to conduct an initial workshop where the different stakeholders are involved. They are supposed to come to an agreement about their involvement in the major use cases, which will inform the awareness of what CMDB objectives there are and how they should be met by the service information provided. These stakeholders include:

- line of business owners/representatives,
- service managers, application managers
- change manager, configuration manager, infrastructure managers,

- and CMDB data consumers in general

As usual, having selected the most important services to address, then the most important use cases should be the focus of the discussion amongst these stakeholders.

It is also important to discuss the granularity of the CMDB, i.e. which item types must be represented in the CMDB, and which item types don't have to be included although many excluded items may be important from other perspectives. For example:

- IT operations needs to monitor infrastructure component items defined at a much finer level of granularity than what potentially is represented in the CMDB as CI types, or through CI attributes/relationships.
- However, it is also important to note that the service itself, as per its degree of dependency, is what dictates the need to include an item as a CI; thus it is even possible that only one instance of something in the infrastructure may be recorded, but its CI record will conform to a family and class that can also be used by any number of additional needed item records going forward.

Presuming the hierarchical model structure, and following the iterative approach, we recommend leveraging a large set of standardized predefined categorizations; with CA CMDB this means bringing the supplied product content early into the modeling analysis.

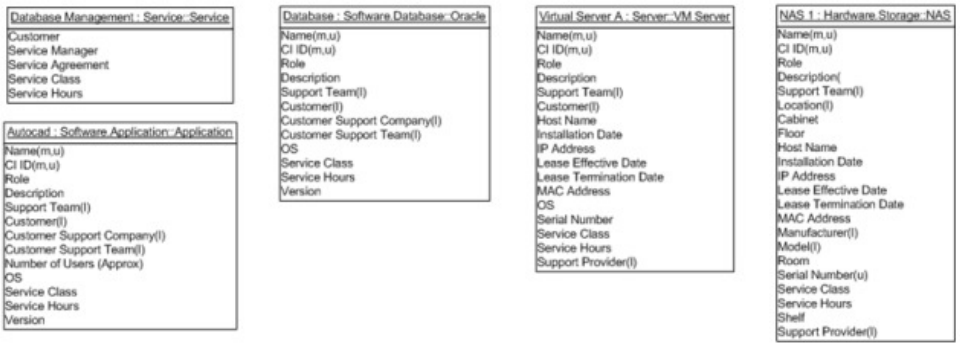
CA CMDB r12.9 CI families

- families can be renamed
- more can be created
- each family has a set of attributes

Family	Family	Family
Cluster	Hardware.Server	Service
Cluster.Resource	Hardware.Storage	Service Level Agreement
Cluster.Resource Group	Hardware.StoragePool	Software
Computer	Hardware.StorageVolume	Software.Application
Contact	Hardware.Virtual Machine	Software.Application Component
Contract	Hardware.VMDataStore	Software.Application Server
Document	Hardware.Workstation	Software.Application System
Enterprise Service	Investment.Idea	Software.Bespoke
Enterprise Transaction	Investment.Other	Software.COTS
Enterprise TransactionContext	Investment.Project	Software.Database
Facilities.Air Conditioning	Location	Software.DirectoryServer
Facilities.Fire Control	Network.Bridge	Software.ESXHypervisor
Facilities.Furnishings	Network.Controller	Software.HyperVHypervisor
Facilities.Other	Network.Frontend	Software.In-House
Facilities.Uninterruptible Power Supply	Network.Hub	Software.MessageServer
Hardware	Network.Network Interface Card	Software.NetworkServer
Hardware.DiskPartition	Network.Other	Software.Operating System
Hardware.EnvironmentalSensor	Network.Peripheral	Software.ResourceServer
Hardware.File	Network.Port	Software.Schema
Hardware.Logical Partition	Network.Router	Software.Tablespace
Hardware.Mainframe	Network.Switch	Software.VirtualManager
Hardware.Memory	Organization	Software.Website
Hardware.Monitor	Other	Telecom.Circuit
Hardware.Other	Projects	Telecom.Other
Hardware.PowerSupply	SAN.Interface	Telecom.Radio
Hardware.Printer	SAN.Switch	Telecom.Voice
Hardware.Processor	Security	Telecom.Wireless

identify CI's attributes for each family

— select for each family the set of attributes needed (and available), examples:



— for each attribute document which MDR will be used to populate it

CA CMDB r12.9 CI classes

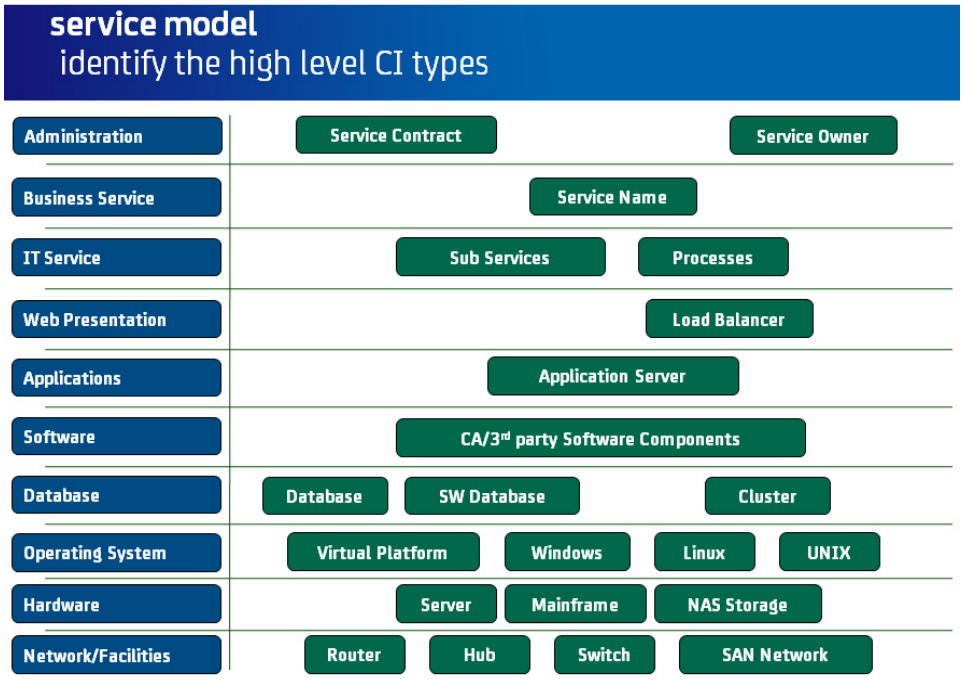
Class	Family
3270 Terminal	Network.Frontend
680x0	Hardware.Processor
80x87	Hardware.Processor
ACD	Telecom.Other
Admin Guide	Document
Air Conditioning	Facilities.Air Conditioning
AirFlow	Hardware.EnvironmentalSensor
AIX	Hardware.Server
AIX OS	Software.Operating System
Alpha	Hardware.Processor
Application	Software.Application
Application Instance	Software.Application
Application Security	Security
Application Server	Software.Application.Server
Application Server Instance	Software.Application.Server
Application System	Software.Application.System
Application Test Plan	Document
ARM	Hardware.Processor
BackgroundProcess	Software.COTS
Barcode Reader	Hardware.Other
Batch	Software.COTS
Bespoke	Software.Bespoke
BFS	Hardware.DiskPartition
Bootsoftware	Software.COTS
Bridge	Network.Bridge
Bubble Jet	Hardware.Printer
Building	Location
Building Security	Security
Business Continuity Plan	Document
Business Service	Enterprise Service
Business Transaction	Enterprise Transaction
BusinessProcessServer	Software.COTS
CA-Datacom	Software.Database
CA-IDMS	Software.Database
Cache	Hardware.Memory
Campus	Location
CD-Rom Drive	Hardware.Storage
Centrex	Telecom.Voice
CICS	Software.COTS
City	Location
Cluster	Cluster
Communication Circuit	Telecom.Circuit
CommunicationServer	Software.MessageServer
Component	Service
Conference Bridge Line	Telecom.Voice
Controller	Network.Controller
Copier	Hardware.Other
COTS	Software.COTS
Country	Location
Cray	Hardware.Mainframe
CRT	Hardware.Monitor
Data Security	Security
DatabaseSchema	Software.Schema
Datacenter	Location
DB2	Software.Database
Desk Phone	Telecom.Voice
Digital Camera	Hardware.Other
Directory	Hardware.File
Discovered Hardware	Hardware
Disk Array	Hardware.Storage
Docking Station	Hardware
Document	Service
DVD	Hardware.Storage
EFI	Hardware.DiskPartition
Electronic Whiteboard	Hardware.Other
Equipment Rack	Facilities.Furnishings
ESX Server	Hardware.Virtual Machine
ESXHypervisor	Software.ESXHypervisor
Executive	Contact
External	Organization
External Contact	Contact
FAT	Hardware.DiskPartition
FAT12	Hardware.DiskPartition

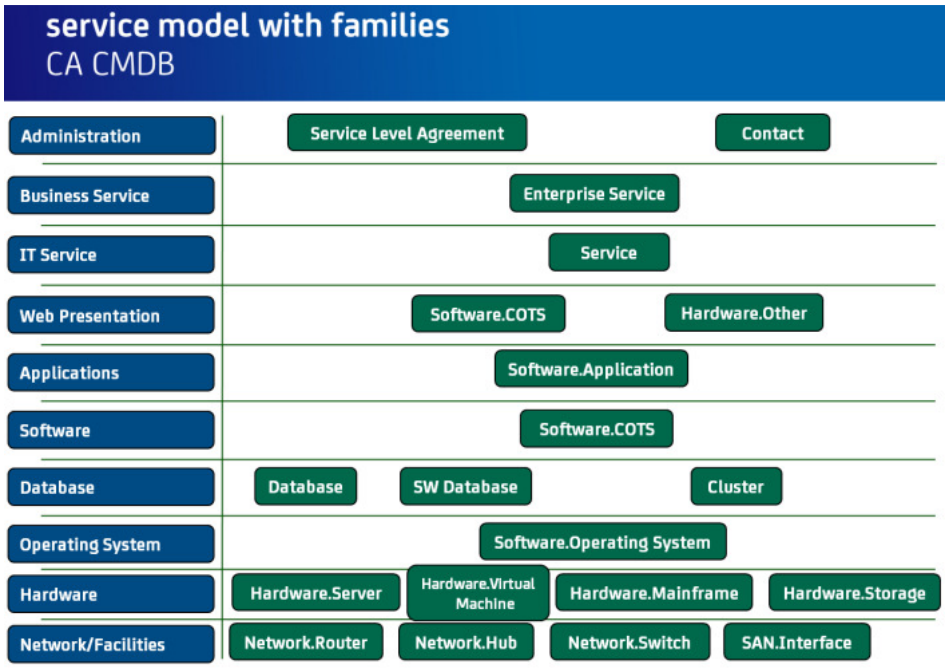
Class	Family
FAT16	Hardware.DiskPartition
FAT32	Hardware.DiskPartition
FAT64	Hardware.DiskPartition
Fax Machine	Network.Peripheral
File	Hardware.File
File Cabinet	Facilities.Furnishings
File System	Hardware.Storage
File-Cabinet	Hardware.File
File-Data	Hardware.File
File-Exe	Hardware.File
File-Log	Hardware.File
Fire Control	Facilities.Fire Control
Flat Screen	Hardware.Monitor
Floor	Location
GasDetection	Hardware.EnvironmentalSensor
Group 80	Hardware.Mainframe
GSX Server	Hardware.Virtual Machine
Hard Drive	Hardware.Storage
HFS	Hardware.DiskPartition
HP UX	Hardware.Server
HP UX OS	Software.Operating System
HPFS	Hardware.DiskPartition
Hub	SAN.Switch
Humidity	Hardware.EnvironmentalSensor
HyperVHypervisor	Software.HyperVHypervisor
IBM-Extended	Hardware.DiskPartition
IMS	Software.Database
In-House	Software.In-House
Infastructure Service	Enterprise Service
Ingres	Software.Database
Ink Jet	Hardware.Printer
Interface	SAN.Interface
Internal	Organization
Intrusion	Hardware.EnvironmentalSensor
IVR	Telecom.Other
JFS	Hardware.DiskPartition
Laptop	Hardware.Workstation
Laser	Hardware.Printer
LDAP	Software.DirectoryServer
License	Software
License Agreement	Contract
Linux	Hardware.Server
Linux OS	Software.Operating System
LocalStore	Hardware.VMDataStore
LocalStore-VMwareFileSystem	Hardware.VMDataStore
Logical Partition	Hardware.Logical Partition
MailServer	Software.MessageServer
ManagementAgent	Software.COTS
Managerial	Contact
MessageServer	Software.ResourceServer
Microfiche	Hardware.Printer
Microsoft Virtual Server	Hardware.Virtual Machine
MIPS	Hardware.Processor
MIPS-BigEndian	Hardware.Processor
MIPS-LittleEndian	Hardware.Processor
Mobile Phone	Telecom.Wireless
Monitor	Hardware
MVS	Hardware.Mainframe
MVS OS	Software.Operating System
Network Attached Storage	Hardware.Storage
Network Hub	Network.Hub
Network Interface Card	Network.Network Interface Card
Network Software	Software.COTS
Network Switch	Network.Switch
Network Terminal	Network.Frontend
NetworkFileSystem	Hardware.VMDataStore
NetworkServer	Software.NetworkServer
NTFS	Hardware.DiskPartition
NTFS1-3	Hardware.DiskPartition
NTFS4	Hardware.DiskPartition
NTFS5	Hardware.DiskPartition
Operational Level Agreement	Service Level Agreement
Optical	Hardware.Storage

Class	Family
OPUS	Hardware.DiskPartition
Oracle	Software.Database
OS/390	Software.Mainframe
OS/390 OS	Software.Operating System
Other Contact	Contact
Other Contract	Contract
Other Document	Document
Other Facilities	Facilities.Other
Other Hardware	Hardware.Other
Other Hardware Mainframe	Hardware.Mainframe
Other Hardware Storage	Hardware.Storage
Other Hardware Virtual Machine	Hardware.Virtual Machine
Other Monitor	Hardware.Monitor
Other Network	Network.Other
Other Operating System	Hardware.Server
Other Printer	Hardware.Printer
Other Security	Security
Other Service	Enterprise Service
Other Service Level Agreement	Service Level Agreement
Other Software Database	Software.Database
Other Software OS	Software.Operating System
Other Telecom	Telecom.Other
Other Telecom Circuit	Telecom.Circuit
Other Telecom Radio	Telecom.Radio
Other Telecom Voice	Telecom.Voice
Other Telecom Wireless	Telecom.Wireless
PA-RISC	Hardware.Processor
Pager	Telecom.Wireless
Paging	Hardware.Memory
PBX	Telecom.Voice
PDA	Telecom.Other
Person	Service
Phone Card	Telecom.Voice
Physical	Hardware.Memory
Physical-Fast	Hardware.Memory
Physical-IO	Hardware.Memory
Physical-Multibus	Hardware.Memory
Physical-PCI	Hardware.Memory
Physical-Processor	Hardware.Memory
Physical-Video	Hardware.Memory
Plotter	Hardware.Printer
Policies and Standards	Document
Port	Network.Port
Portfolio Application	Investment.Other
Portfolio Asset	Investment.Other
Portfolio Idea	Investment.Idea
Portfolio Product	Investment.Other
Portfolio Program	Investment.Project
Portfolio Project	Investment.Project
Portfolio Service	Investment.Other
Portfolio Work	Investment.Other
PowerPC	Hardware.Processor
PowerSupply	Hardware.PowerSupply
Practice	Service
Pressure	Hardware.EnvironmentalSensor
Probe	Hardware
Process	Service
Projector	Hardware.Other
Radio Data Modem	Telecom.Radio
Radio Handsets	Telecom.Radio
Resource	Cluster.Resource
Resource Group	Cluster.Resource Group
ResourceServer	Software.ResourceServer
Role	Service
Router	Network.Router
s390x	Hardware.Processor
Satellite Link	Telecom.Circuit
Security Software	Software.COTS
SecurityServer	Software.ResourceServer
Server	Hardware.Server
Service	Service
Service Level Agreement	Service Level Agreement
Shredder	Hardware.Other

Class	Family
Silo	Hardware.Storage
SoftwareComponent	Software.Application.Component
SPARC	Hardware.Processor
SQL	Software.Database
STC	Software.COTS
Storage Area Network	Hardware.Storage
StoragePool	Hardware.StoragePool
StorageVolume	Hardware.StorageVolume
Sun	Hardware.Server
Sun OS	Software.Operating System
Switch	SAN.Switch
Sybase	Software.Database
System 390	Hardware.Mainframe
System z	Hardware.Mainframe
Tablespace	Software.Tablespace
Tandem	Hardware.Server
Tandem - Mainframe	Hardware.Mainframe
Tandem OS	Software.Operating System
Tape Array	Hardware.Storage
Tape Library	Hardware.Storage
Technical	Contact
Television	Hardware.Other
Temperature	Hardware.EnvironmentalSensor
Terminal	Hardware.Monitor
Training Class Collateral	Document
TransactionalNTFS	Hardware.DiskPartition
TransactionContext	Enterprise.TransactionContext
TransactionServer	Software.ResourceServer
TSO	Software.COTS
Undisgoining Contract	Service Level Agreement
Unformatted	Hardware.DiskPartition
Uninterruptible Power Supply	Facilities.Uninterruptible Power Supply
Unisys	Hardware.Server
Unisys - Mainframe	Hardware.Mainframe
Unisys OS	Software.Operating System
Unix	Hardware.Server
Unix OS	Software.Operating System
Unknown	Hardware.EnvironmentalSensor
User Guide	Document
Vax	Hardware.Server
Vax - Mainframe	Hardware.Mainframe
Vax OS	Software.Operating System
VCR/DVD	Hardware.Other
Vibration	Hardware.EnvironmentalSensor
Video Camera	Hardware.Other
Video Conferencing Unit	Telecom.Other
Virtual Storage Array	Hardware.Mainframe
Virtual Tape System	Hardware.Storage
VirtualImage	Hardware.File
VirtualImage-CDDVD	Hardware.File
VirtualImage-Floppy	Hardware.File
VirtualizationManager	Software.VirtualManager
VM	Hardware.Server
VM OS	Software.Operating System
VMware	Hardware.DiskPartition
Volume	Hardware.File
Warranty/Maintenance Contract	Contract
Website	Software.Website
WebSphere MQ	Software.COTS
Windows	Hardware.Server
Windows OS	Software.Operating System
Workstation	Hardware.Workstation
X Terminal	Network.Frontend
x86	Hardware.Processor
x86-32	Hardware.Processor
x86-64	Hardware.Processor
x86-64-AMD64	Hardware.Processor
x86-64-IA64	Hardware.Processor
Xenix	Hardware.DiskPartition
z/OS	Hardware.Mainframe
z/OS OS	Software.Operating System
Zip Drive	Hardware.Storage

The next step is mapping the CI types (CA CMDB families and classes) into appropriate layers of the hierarchical service model.





In the hierarchical modeling, any item on a given layer will presumably be a provider to an item on a higher layer, while it would be a dependent of an item on a lower layer.

The CA CMDB also comes with pre-built content (pre-defined entities) when it comes to relationship types, which can be leveraged as is or modified, and new relationship types can also be created. The CA CMDB in its current releases also doesn't contain any restrictions on which relationship types can be used between the CI types. Maintaining the logical correctness of selected CI associations must be done by the parties who are modeling the service.

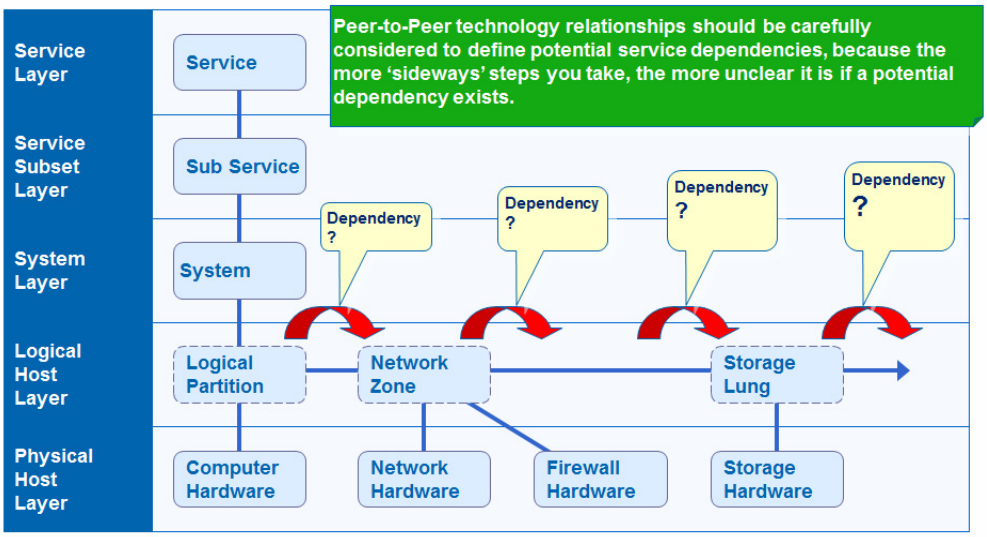
CA CMDB r12.9 relationship types

Provider To Dependent	Dependent To Provider	Is Peer-to-peer?
administers	is administered by	No
approves	is approved by	No
authors	is authored by	No
authorizes	is authorized by	No
backs up	is backed up by	No
connects to	connects to	Yes
contains	is contained by	No
defines	is defined by	No
deploys	is deployed by	No
documents	is documented by	No
governs	is governed by	No
hosts	is hosted by	No
has as assignee	is assigned to	No
is high availability server for	has for high availability server	No
is gateway for	has for gateway	No
is the child of	is the parent of	No
is recovery server of	has for recovery server	No
fails over	fails over	Yes
manages	is managed by	No
monitors	is monitored by	No
notifies	is notified by	No
is required by	requires	No
runs	runs on	No
secures	is secured by	No
services	is serviced by	No
is subscribed to by	subscribes to	No
supports	is supported by	No

Provider To Dependent	Dependent To Provider	Is Peer-to-peer?
updates	is updated by	No
is used by	uses	No
is business owner of	is owned by	No
is source code for	source code is from	No
serves	is served by	No
is server of	is client of	No
regulates	is regulated by	No
controls	is controlled by	No
complies to	is complied to by	No
communicates with	communicates with	Yes
is location for	located at	No
fronts	is fronted by	No
provides to	is provided by	No
is proxy for	is proxied by	No
is primary contact for	has primary contact of	No
has access to	is accessed by	No
has detail	is detail of	No
has Member	is member of	No
is affected by	affects	No
is bound to	is bound to	Yes
is clone of	is original of	No
is composed of	is component of	No
is discovered by	discovers	No
is evolution of	is evolved to	No
is successor of	is predecessor of	No
is triggered by	triggers	No
is result of	results in	No

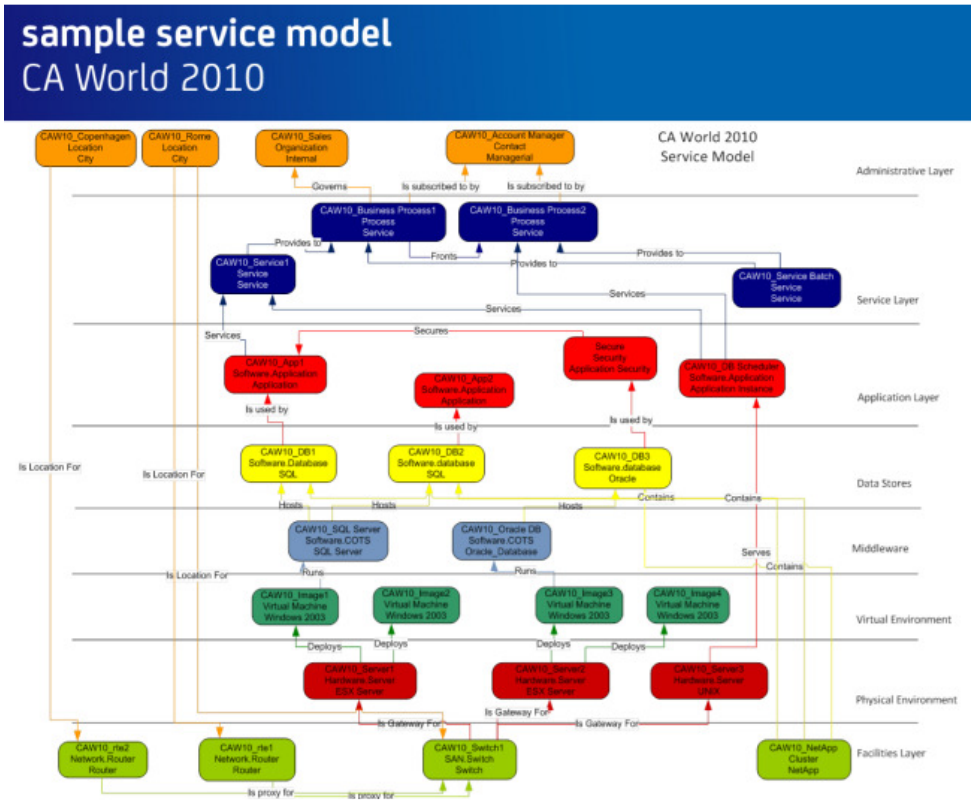
- they can be renamed
- or you can create new ones

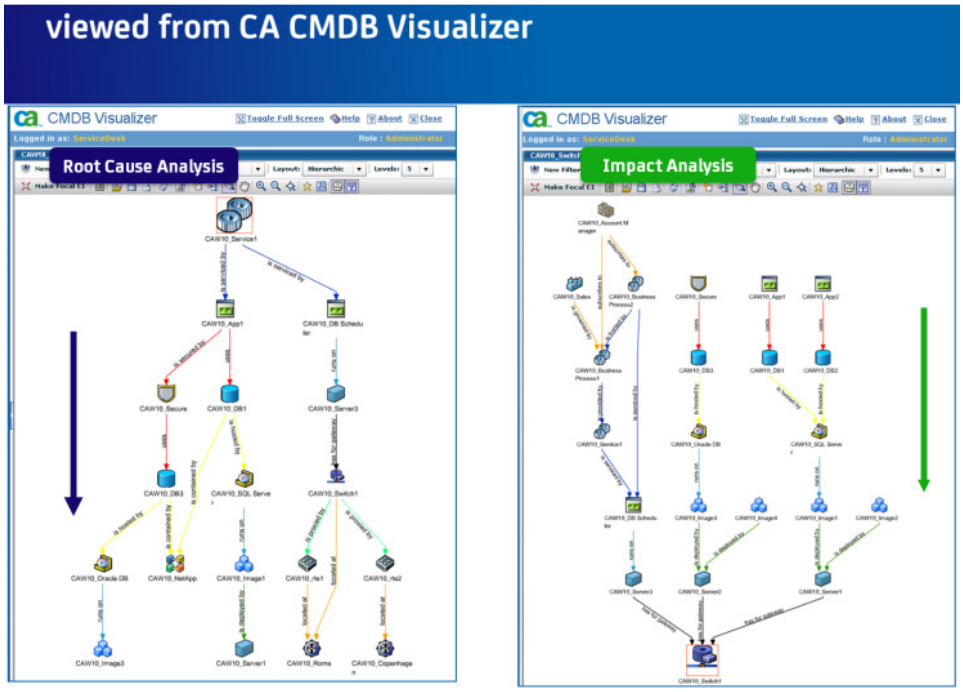
Peer-to-peer relationships can be used in modeling, but because they do not communicate dependencies, they are far less important within the model per our two major use cases: critical dependency-based analyses of root cause and change impact.



The current CA CMDB technology at hand simply avoids that ambiguity. One of the key features of the tool is to search for CIs and relationships. However, in its current release, several of the technology's key filters do not take peer-to-per relationships into account. Notably, this applies to CMDB Visualizer filters for impact analysis and root cause analysis.

The result of combining the layers, defined CIs and defined relationships is diagramming such as the below:





Service Management Integrations

A “service” is an idea that has meaning primarily because it represents the business perspective on how to leverage IT. By using services, business users consume IT as a resource. In that way, we take “service consumption” as the central point of view for deciding how to identify, track and evaluate infrastructure components.

In this viewpoint, the recommended ideal interaction between the business users and the services they consume would be through a service catalogue, in which the terms and conditions (agreements like SLAs) of provision and support are linked to the services.

Services represented in a service catalogue are therefore good examples of IT services that should be represented in the CMDB.

CA Service Catalog
catalogue the standardized service offerings

After services are requested, we gain visibility of whether the deliverables actually fulfill those agreement obligations through reporting tracked results in a business service dashboard. But the results are obtained through active investigation of the service behaviors.

Monitoring of systems and applications provides the bulk of the evidence on why a service may have some "in process" degradation or interruption. This evidence indicates current actual states within the service structure.

In contrast, configuration management builds the CMDB to provide description of an authorized prescribed state of services. Operations need to support that prescription with dynamic monitoring of the current actual operational status, often through attention to more granular levels of attribute detail than the service model of the CMDB provides. **CA Service Operations Insight** or SOI, which is CA's consolidated event correlation engine, is a contributor of information; SOI's underlying monitoring tools also act as discovery tools of both attributes and relationships of components; that capability stages the ability to cross-reference discovery of components and states with the prescribed conditions and defined CIs as found in the CMDB.

To accomplish this cross-reference, the idea for the design of the CMDB is to import the service model defined in the CMDB into CA Service Operations Insight so that operations consumes the service model into its own operations management database (OMDB, which by the way is not an ITIL term).

The benefit is to be able to manage infrastructure according to business priority. To drive realization of this benefit, the best practice is to assure the cross-referencing of SOI discovery with CMDB records.

- If we populate the CMDB, while discovery, management and monitoring tools populate the operations management database (OMDB) the key is to reconcile the two views of components.
- If the very same operational tools feed their important events into CA Service Operations Insight, SOI's Service Dashboards will give operations the insights required to improve the monitored states and components, and the CMDB linkage will expose where this improves service quality and eliminates or at least minimizes the risks to service delivery.

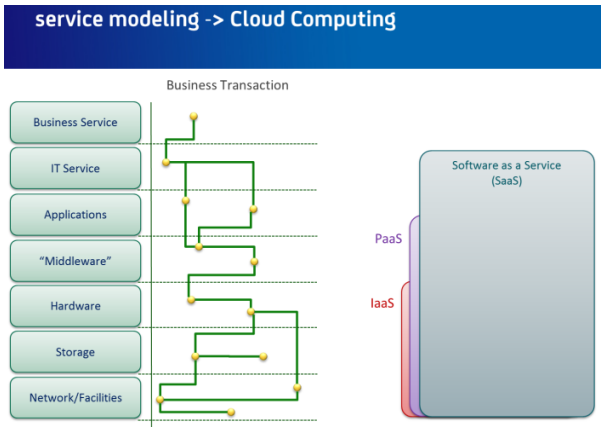
The service view then lets operations prioritize support of infrastructure according to the business importance, quality and risk of services.

Modeling Services in a Cloud Computing Scenario

As Cloud Computing matures, we'll probably approach a model similar to what we know today from power supplies, telco supplies, etc. where we as technicians have quite limited insight into how these service providers actually deliver the power and communication lines that we all depend on. We'll instead have to rely on the services being delivered according to the SLAs or underpinning contracts that we've signed with them as a company.

That is a scenario paralleling the circumstances initially described in this paper:

The business conditions are that the service recipient neither knows nor cares how the service gets provided (from right to left), as long as the actual provision meets user expectations.

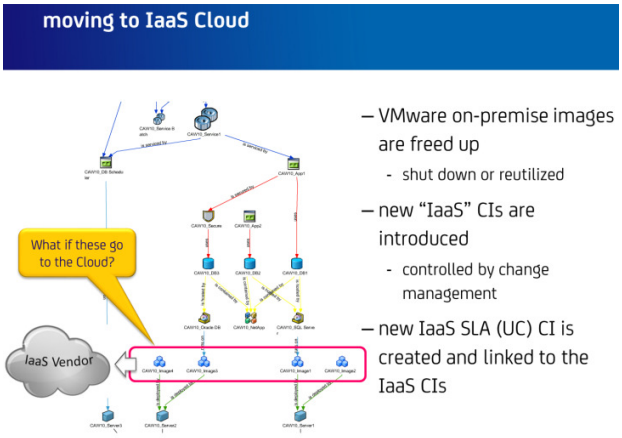


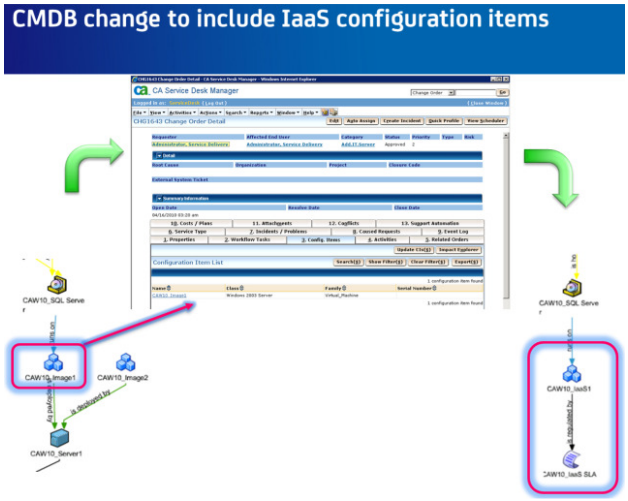
The scenario here is about introducing IaaS (amongst other options) into our service provision practice. What happens if we sign up with an IaaS vendor and subscribe to their IaaS services? Well, as a CIO we would expect that one, some or maybe even all of our similar virtual images in our infrastructure are going to be hosted by the IaaS vendor.

So the virtual images that are currently running on-premise will be shut down, or reutilized for other purposes, potentially for test or pre-production. The CIs representing the virtual images

highlighted in our usual service model are going to disappear or at least become irrelevant, and instead we'll have to introduce new CIs representing the virtual images running in the IaaS provider's environment, and then link this Provider environment as a CI into our service model.

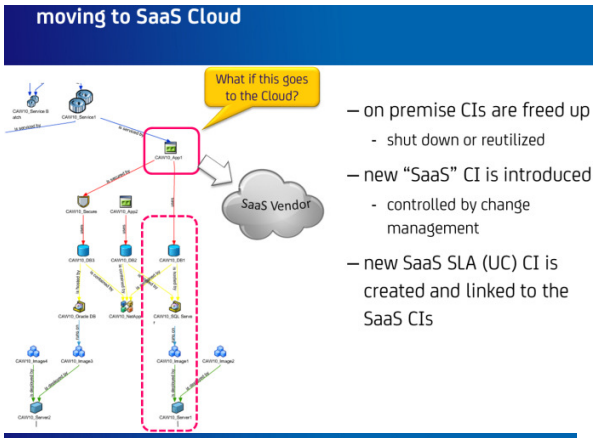
As a CIO we would also have agreed with our IaaS service provider or subcontractor on some terms and conditions about the IaaS services delivered, which from our point of view could be considered an underpinning contract and from the IaaS organization would be considered an SLA with our organization as the customer.

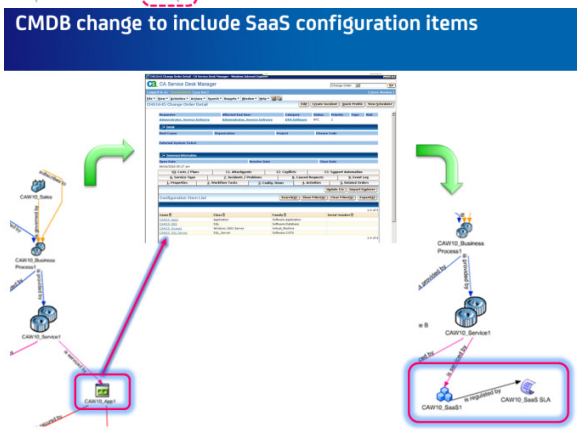




Now whether we (as an IaaS customer) are going to or even allowed to discover and as such be able to represent the physical and virtual components of the IaaS vendor's environment in our CMDB is a complete other question that would all come down to an agreement with the IaaS provider, and also whether it's of any interest to our CIO or not.
We also must not forget to go back and talk to our consumers of the information that we're managing in the CMDB whether this level of detail would be needed or not.

The same situations would apply if we were subscribing to Software as a Service (SaaS):





SECTION 4

Conclusions

Infrastructure information is readily available from many existing management disciplines that maintain data repositories. However, conventionally these have not been collections of information designed to describe services. Applying service models to the existing management information requires agreement amongst stakeholders on the target use of data, mainly being analyses aimed at assuring the manageability of infrastructure based on business priorities for service provision.

Basic initial service modeling is aimed at selectively identifying the key characteristics that account for those interdependencies within the service's infrastructure that are typically manageable in restoring or predicting the behavior of the service. The service model drives the use of standards in descriptions, allowing multiple management stakeholders to better understand how to engage the infrastructure on behalf of assuring service quality. The essential challenge is to adopt an information design and management technique that pragmatically produces the necessary reference data in the form of a practical and

maintainable CMDB. This tactical approach is the most likely to produce useful results from the level of effort required.

SECTION 5

References

This discussion is based on a number of sources including the ITSMF; industry analysis by EMA and Forrester; published ITL v3 books; field work by CA Services; prior original reporting and consultative field work by the authors; and product documentation from CA Technologies.

SECTION 6

About the Authors

Brian Johnson moved ten years ago to the USA to work for CA Technologies where he practices business and solutions architecture for CA Services. Brian was part of the team that created ITIL and has been a published author in all three versions. In total he has either authored or co-authored over twenty five books related to good practices in IT management, including a series of cartoon books to illustrate that he is not completely boring.

Malcolm Ryder joined CA Technologies eight years ago where he has practiced solutions development for CA and architecture for CA clients, drawing on over 20 years in ITSM and management consulting. Malcolm was part of the team that gained the first ITIL certification for a commercial ITSM solution. He has co-created or managed several market-leader ITSM products from various companies.

John Sorensen joined CA Technologies seventeen years ago where he is a Sr. Principal Consultant in the EMEA presales organization practices. In the context of this white paper John volunteered to consolidate ideas developed by him and exchanged with colleagues in order to come up with a pragmatic and minimalistic approach of designing and implementing a CMDB in order to provide immediate return on the investment (ROI) for CA Technologies' customers.

The authors would like to thank many reviewers throughout CA Technologies for their probing and validations of key points covered in this paper.