Appendix B

The Configuration Management Database and Configuration Management System

There is no question that **application performance** and the performance of the business services they support, as well as initiatives such as **DevOps** and **agile**, all depend on managing change effectively across the broader application infrastructure. The requirements to do this have only accelerated, given the more dynamic and heterogeneous environments of public and **private cloud**, the move to virtualization, and even the need to provide more effective approaches for managing **containers** and **microservices**.

Ongoing research and dialogue with information technology (IT) organizations reaffirm that despite current industry controversy (including a fair amount of "bad press"), solutions related to **configuration management databases (CMDBs)**, **configuration management systems (CMS)**, and **application discovery and dependency mapping (ADDM)** are actually on the rise in terms of innovation and relevance. This is because the need to discover, capture, and optimize application-to-application and application-to-infrastructure interdependencies could not be more critical given dynamic, heterogeneous infrastructures. The use cases relevant to understanding service interdependencies range from **performance management**, to **change management**, to **asset management**, to data center consolidation, to security concerns.

This appendix, taken from *CMDB Systems: Making Change Work in the Age of Cloud and Agile*,[1] explores the processes and the technologies relevant to CMDB systems as they evolve to become more automated, more analytics-aware, and more supportive of transforming IT efficiencies and effectiveness.

Given the manifold debates about issues and values in CMDB systems, it comes as no surprise that few topics in IT today are as confusing and misunderstood as the CMDB. Some people define the CMDB and the CMS by their strict **IT Infrastructure Library (ITIL)** definitions, while others use one of the many descriptions given by the CMDB vendors in the market. Still others see the CMDB as a homegrown collection of data that never found its way to being useful. One common misconception is that a CMDB contains only configuration information on servers and devices in the infrastructure. Depending on the particular deployment, a CMDB may also contain information on applications, middleware, documentation, people, processes, providers, and other relevant information in addition to infrastructure components.

This appendix will examine the two foundations for the CMDB system: ITIL processes and associated CMDB-related technologies, as shown in Fig. B.1. These two foundations are not always understood from a common set of balanced perspectives. Processes are often favored by one set of

[1]Drogseth, D.N., Sturm, R., Twing, D., 2015. CMDB Systems: Making Change Work in the Age of Cloud and Agile. Morgan Kaufmann.

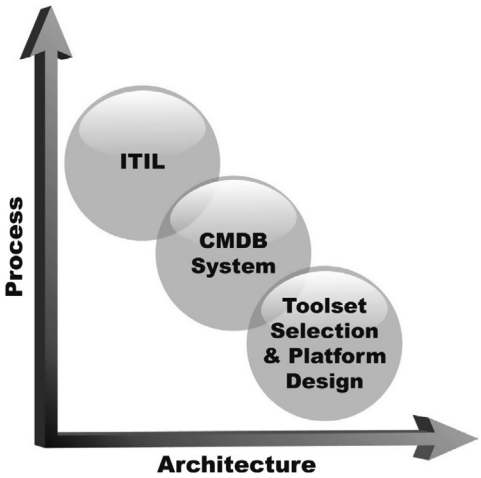

FIGURE B.1

The CMDB system's two foundations.

stakeholders, while other stakeholders may be fully consumed by technologies, integration issues, and architecture. If there is one central idea behind optimizing CMDB system deployments, it is the need to promote balanced attention and facilitate dialogue to both process and technology.

CMDB FOUNDATIONS PART 1: PROCESS

ITIL may not be perfect, and it is categorically best if viewed as a "departure point" rather than an ideology (as some might claim). However, ITIL offers well-thought-out guidelines that also pose questions with the potential to spark valuable analysis as IT managers and practitioners seek to adopt ITIL recommendations within their own unique environments.

ITIL is:

- A common lexicon for disparate disciplines
- Holistic—a place where the full IT process mosaic can come together
- A high-level set of recommendations for what processes should be in place
- A valuable catalyst

ITIL is not:

- A standard
- Completely prescriptive
- A perfect answer for a perfect world
- A panacea for all of IT's ailments

The ITIL framework is based on a service lifecycle and consists of five lifecycle stages: *Service Strategy, Service Design, Service Transition, Service Operation,* and *Continual Service Improvement*. There is a supporting publication for each of these stages as well as a set of complementary ITIL publications that provide guidance specific to industry sectors, organization types, operating models, and technology architectures.

ITIL'S CONFIGURATION MANAGEMENT DATABASE

CMDB is defined as a database used to store configuration items throughout their lifecycle. The CMS maintains one or more CMDBs, each of which stores attributes of configuration items and relationships with other configuration items.

ITIL defines **configuration item** (CI) as "any component or other service asset that needs to be managed in order to deliver an IT service." Additionally, "service transition" clarifies that CIs "typically include IT services, hardware, software, buildings, people, and formal documentation such as process documentation and service-level agreements." Defining, managing, and optimizing CIs and their interdependencies are at the heart of CMDB system deployments.

The dimensionality and range of configuration items open up a whole world of interdependencies and insights for which ITIL should be given a great deal of credit. We are no longer talking about just managing things—we are talking about managing IT services in their fully human and technical dimensions.

THE CONFIGURATION MANAGEMENT SYSTEM

In understanding the real impact of the CMS, it is best to stay focused on the term "system," as ITIL's CMS includes both a sense of federation and potentially more dynamism than a single, standalone CMDB.

According to ITIL, a CMS is:

> …a system that supports configuration management through effective data assimilation and toolsets that track, aggregate, analyze, and showcase critical information regarding configuration items and CI relationships. The CMS may also include information about incidents, problems, known errors, changes and releases. The CMS is…used by all ***IT service management*** processes.

This definition of CMS (in ITIL V2, largely used to define the CMDB itself) allows for more flexibility with "incidents, problems, known errors, changes and releases." As such, the CMS can become a foundation for empowering virtually any IT process or requirement, which increasingly includes operations and even **development**—well beyond ITIL's traditional service desk roots. Note the concluding sentence in the ITIL definition: "The CMS is…used by all IT service management processes." In the real world, this is a statement of potential that needs to be addressed in phases. Remember, despite its potential to bring about huge benefits, the CMDB system is not a single magic bullet. In its entirety, and in real deployments, the CMS represents a much more streamlined, systemic, and dynamic approach than the market stereotype for the traditional CMDB, in which service modeling and discovery are beginning to take center stage, supplanting data storage as the true "spine" of the system.

THE SERVICE KNOWLEDGE MANAGEMENT SYSTEM

ITIL's definition of CMS brings up yet another term: the **service knowledge management system (SKMS)**, which may be ITIL V3's single most visionary idea. If the CMS can house multiple CMDBs optimized for different levels of detail and relevance, the CMS is itself an enabler for this even larger vision—the SKMS.

ITIL defines the SKMS as:

> …a resource that encompasses the Configuration Management System along with other databases and information repositories. It is a true service management system that can acquire, analyze, update, store, and display all the information that an IT service provider needs access to in order to offer the most complete and precise IT services for their customers.

The SKMS is far from just being about collecting data and storing it in a single physical database; it is a system that facilitates the analysis and presentation of knowledge to support the "full lifecycle of IT services." If you want to achieve value through your CMDB system and you completely ignore the need for analyzing and presenting knowledge (not to mention automation), you will never optimize your efforts (Fig. B.2).

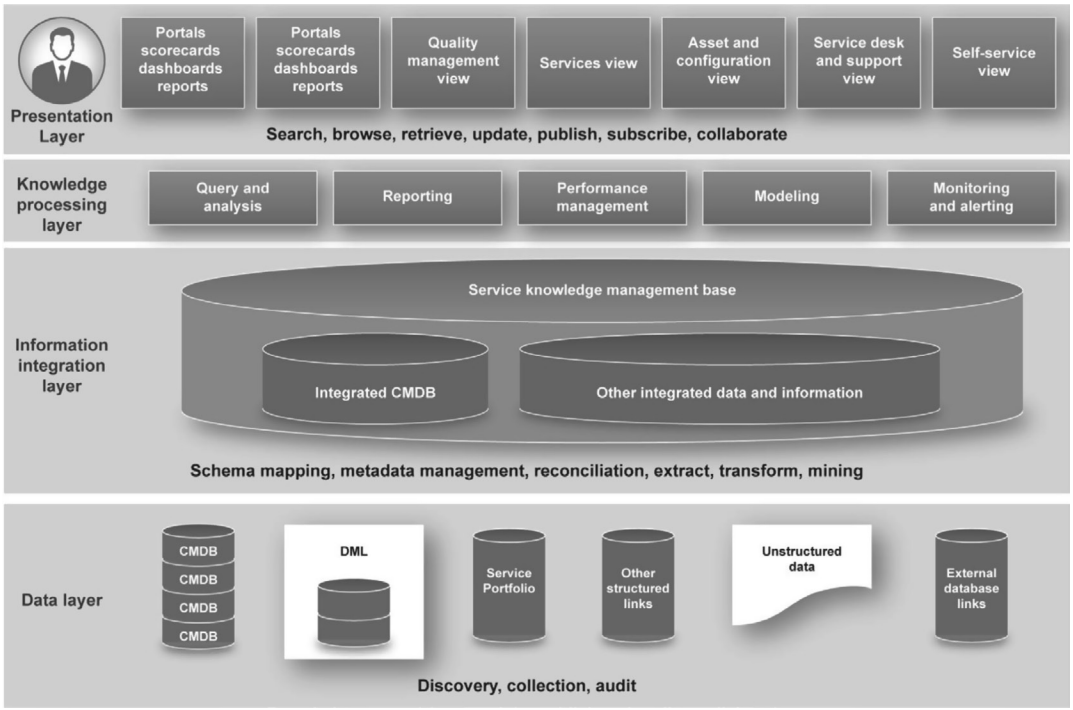

FIGURE B.2

The primary goal of ITIL V3's SKMS.

Figure B.2 shows an advanced logical architecture in which a core or "integrated" CMDB is federated with multiple CMDBs optimized in content and granularity for separate stakeholder groups (EMA's term for this is "citizen CMDB"). Core CMDBs enable the full IT organizations to enjoy a cohesive and reconciled view of services and their interdependencies.

CMDB DATA AND CONFIGURATION ITEMS: A SECOND LOOK

Understanding CMDB-related data and CIs is central to CMDB system success. ITIL V2 grouped CMDB-relevant data into the following logical categories:

Technology data—Includes asset data such as costs, locations, hardware and software configurations, related contracts, software licenses, and maintenance and support histories, physical and logical topology data, and operational data, such as availability, performance, and capacity.
Process data—Includes service models that map assets to services, business process models that map services to business processes, and IT service management process workflows.
People data—Includes asset-to-user mapping and user information, such as roles and responsibilities. It also includes IT staff member data, such as asset-to-support mapping useful for incident escalation (indicating which IT staff members support which assets).
Provider data—Includes which service providers engage, when, and under what conditions.

ITIL 2011 also refines how to better understand CIs and CI categories beyond broad groupings of hardware, software, buildings, people, documentation, etc. For example, ITIL 2011 also includes the following:

Service lifecycle CIs—Including business cases, release documentation, test plans, etc.
Internal CIs—Comprising those required for individual projects internal to the IT organization.
External CIs—Such as external customer requirements and agreements, releases from suppliers or subcontractors, and external services.
Interface CIs—Those required to deliver the end-to-end service across an ecosystem of multiple service providers, for example, an escalation document that specifies how incidents will be transferred between two service providers.

This last category is especially intriguing as it shows how ITIL is setting the stage for the very real and growing challenges of managing services across many separate political entities, which may nonetheless constitute a true business ecosystem of partners, suppliers, and service providers of all varieties (Fig. B.3).

Another key foundational concept from ITIL is the CI attribute, defined as, "A piece of information about a configuration item. Examples are name, location, version number, and cost. Attributes of CIs are recorded in a configuration management database (CMDB) and maintained as part of a configuration management system (CMS)." In real CMDB system deployments, CI attributes can extend to multiple sources through a consistent service modeling system (as described in the following section on technology) to support unique use case requirements, from capacity to performance, or state to financial data.

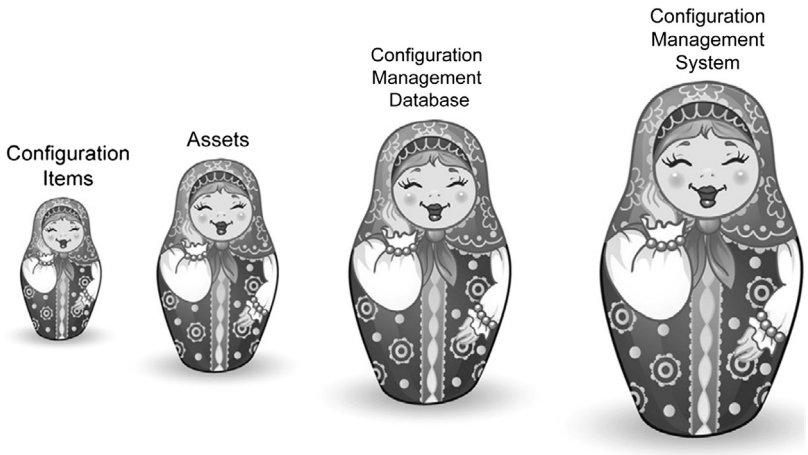

FIGURE B.3

Russian nesting dolls as an analogy for the relationship between CIs, assets, the CMDB, and the CMS.

CMDB FOUNDATIONS PART 2: CMDB SYSTEM TECHNOLOGIES AT A GLANCE

The fact is that the CMDB system is not a single technology, which in itself has become a major source of confusion. Sadly, the CMDB, and to a lesser degree the CMS, were largely defined by various constituencies according to what is most convenient for them. These include vendors that develop (or attack) CMDB/CMS-related solutions and technologies, consultants and systems integrators with processes and game plans associated with CMDB-related deployments, analysts seeking to cram the CMDB into a two-dimensional technology market, and even IT executives and professionals who have narrowly construed wishes for a magic bullet tailor-made for them.

EMA often refers to the challenge of CMDB-related deployments as an issue of the blind men and the elephant, in which a group of blind men touch an elephant to learn what it is. Each man feels only one part of the elephant, such as the tusk or the leg, and they compare notes. The men learn that they are in complete disagreement about the elephant. The same is true regarding the challenge of CMDB-related deployments. While the siloed perspectives usually include debates between those wedded to process (sometimes too firmly) and those who see the CMDB system primarily in terms of technology, the analogy can just as well be applied to the confusion and prejudice arising from the relevant technologies alone.

CMDB SYSTEM TECHNOLOGIES: A CLOSER LOOK

Critical CMDB system foundational technologies include the CMDB itself, application discovery and dependency mapping, other discovery and inventory tools, automation, analytics, dashboard and visualization, and other investments.

Configuration Management Database

The CMDB's core functions are to assimilate and reconcile critical data sources through manual population, bulk updates, or automation, informed by policies, dialogues, and increasingly by analytics. In turn, CMDB-supported modeling can help to articulate, access, and promote the critical service interdependencies that will ultimately make the CMDB investment worthwhile. How the CMDB evolves should correlate directly with careful IT planning that targets a phased approach based on use case priorities, such as *Change Management, Asset Management, Service Impact Management,* and even *Performance Management*.

The CMDB, whether a federated "citizen CMDB" or core, "integrated CMDB," is not a data warehouse optimized for high-volume data analytics. Rather, it is a resource aimed at capturing critical service interdependencies, relationships, and device and software configurations. Currently, few vendors offer the CMDB as a standalone solution, but typically embed it in other systems such as service desk with asset and change management, or service performance solutions vendors where CMDBs are optimized for more real-time usage.

Application Discovery and Dependency Mapping

The central value for all ADDM solutions is to discover, in as automated a fashion as possible, application-to-infrastructure and infrastructure-to-infrastructure interdependencies critical for the CMDB system and for effective service management overall. Cloud and virtualization have challenged, and continue to challenge, ADDM solutions to become more dynamic and adaptive to real-time, or near-real-time, changes.

ADDM solutions may be agentless, agent-based, or a combination of both. The varied approaches represent vendors' best efforts to get beyond thorny security issues, and even reach into **public cloud** environments (Fig. B.4).

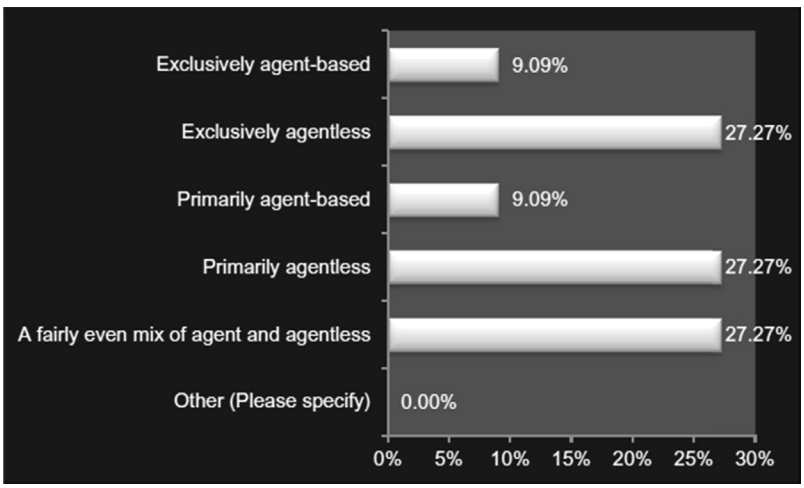

FIGURE B.4

ADDM solution approaches.

ADDM typically falls into either less real-time, configuration-centric capabilities more traditionally associated with CMDBs, or more transaction-aware, real-time capabilities with roots in performance management. Both are relevant for a CMS optimized to empower a true SKMS, and both are becoming more and more pervasive and effective in CMDB system deployments. Moreover, as the ADDM market evolves, we anticipate a gradual merging of ADDM values—in which real-time performance insights can directly become supportive of configuration and even asset interdependencies.

Other Discovery and Inventory Tools

Discovery and inventory tools can be correlated with CMDB deployments either dynamically or manually. In some cases, they are integrated with ADDM solutions to feed the CMDB system as a single package. They can range widely based on domain and use case, from network discovery to PC inventory, to security, to capacity planning, to monitoring tools that "discover" key pieces of infrastructure, and/or applications in support of event or performance-related analytics. We identified more than 50 discovery sources in some enterprises, while those with multiple geographical presences and localized buying preferences can escalate this number into the hundreds.

Automation

Little could be more critical to CMDB success than proactively phasing in automation, both in populating and maintaining the CMDB and in optimizing CMDB system insights in managing change, service performance, and even lifecycle asset values. Automation comes in multiple styles, ranging from what the market would call "configuration tools," to runbook and IT process automation, to triggering events based on shifting CI attributes for remediation and alerts. A partial list of options to consider for phased priorities over time might include:

- PC configuration, **patch management**, and audit capabilities
- System configuration, VM provisioning, and patch management
- Network configuration solutions
- Service desk workflow
- Development-centric simulation and design automation
- Automated application provisioning (DevOps) typically associated with cloud and virtualization
- Manifests for transitioning development models into production
- Security-related identity and access management solutions
- Runbook or IT process automation for stitching multiple automation capabilities together
- Load balancing and other performance-related capabilities
- Other cloud-specific or virtualized automation surrounding a whole new array of packaged solutions, including **software-defined data centers** and **software-defined networks**

Analytics

If the CMDB system is primarily about insight, then that insight needs to be fed and optimized through good analytic tools. These tools may have a range of capabilities, from advanced reporting to truly advanced heuristics, including self-learning algorithms, transaction analytics, and big data in various forms. "Big data for IT" and "operational analytics" have become buzzwords, but in some respects they

do not do justice to the breadth and variety of analytic choices that can contribute meaningfully to CMDB-related initiatives. The following are just a few categories to consider:

- Self-learning predictive analytics, some of which already use service modeling to correlate performance anomalies with CMDB-related interdependencies
- Big data in the form of integrated data warehousing and data mining, ranging from visualization and search to more advanced trending over time for capacity, performance, and other insights
- Big data in terms of integrated data warehousing and data mining for IT financial planning and optimization for CapEx investments and OpEx efficiencies
- Transaction-centric analytics that may look not only at IT service performance, but also at user behaviors and business outcomes, including **user experience** insights

All of these analytic options are already being actively mapped to service modeling and integrated with CMDB systems in some deployments. Most analytic **integrations** favor ADDM over core CMDB for a variety of reasons, chief among them dynamic currency. Some analytic tools already have ADDM-like modeling built into them.

Dashboards and Visualization

Dashboards and other forms of visualization may be directly or indirectly affiliated with any of the above technology choices. Good visualization is where IT harvests and shares its insights within silos and, even more importantly, across domains.

Other Relevant Investments

Enabling and empowering the full CMDB system across multiple phases and use cases may well involve other technology investments beyond the general categories listed above. Two examples are project management and social IT, both of which ideally support better stakeholder planning and dialogue. Another relevant investment, the service catalog, deserves special attention as an outwardly-facing expression of CMDB modeling to empower automation and governance in provisioning (or enabling self-provisioning for) new IT services.

A LOOK AT THE BROADER CMDB SYSTEM AND THE SERVICE MANAGEMENT ECOSYSTEM IT SUPPORTS

As shown in Fig. B.5, the potential reach of the CMDB system in supporting what ITIL would call an SKMS is rich and diverse. Indeed, even Fig. B.5 is not meant to be complete or exhaustive; it is intended merely as a sketch of how different technology investments can come together to harvest the power of CMDB-driven insights over time.

EMA once labeled the two spheres in Fig. B.5 "the two CMDBs." The idea behind this naming was to show how real-time or near-real-time insights could provide strong value in concert with more process-centric systems, so changes can immediately be understood and validated in terms of infrastructure, application service, and business outcomes. Conversely, when problems arise, insights into how and where changes were made and managed could immediately be linked with performance-related issues.

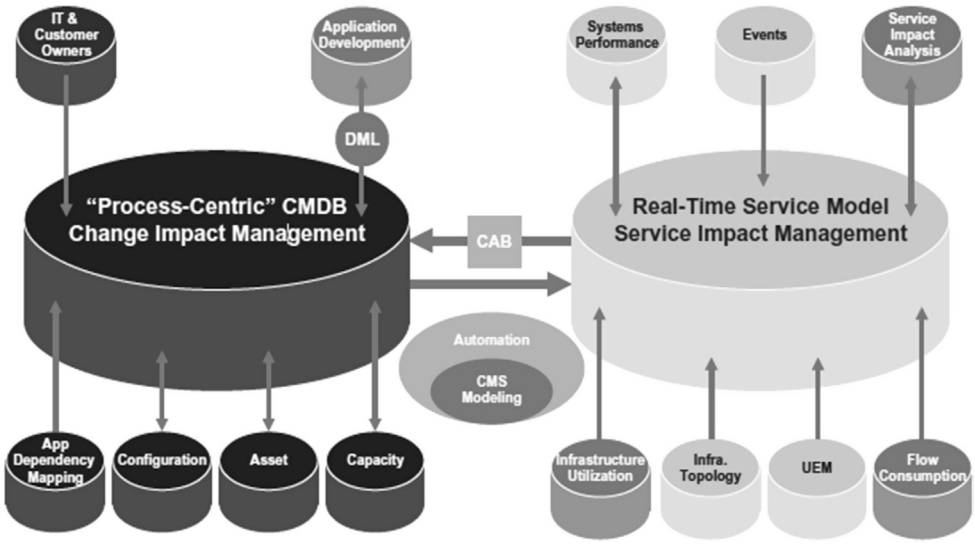

FIGURE B.5

ITIL support of SKMS.

However, the term "two CMDBs" no longer applies to the current market and industry taxonomy that buckets CMDB technology as a single, physical database. We are now using the term "real-time service model" to suggest how CMDBs, including federated CMDBs, may access critically relevant CI-related data without necessarily moving into a separate data store. This is a critical distinction because it centers the broader CMDB system less in database technology than in service modeling capabilities with reconciled data access.

A summary of what is included in Fig. B.6 is described in more detail with the following list:

- IT and customer owners—This phrase is shorthand for IT stakeholders, managers, and executives who have relevance to the broader CMDB initiative, as well as service and consumer owners outside of IT.
- Applications Development and the Definitive Media Library—These terms apply as the CMDB system evolves to support development and DevOps requirements for staging the introduction of new application services across the infrastructure. Analytic, automation, and service dependency insights are also key here.
- Application Dependency Mapping—This concept was discussed under "Application Discovery and Dependency Mapping" (ADDM). As CMDB systems evolve, the combination of real-time, transaction-centric ADDM tools with more traditional configuration-centric ADDM may serve as the spine for the broader system.
- Configuration—This term is shorthand for "configuration automation" solutions, such as those targeting network, systems, PCs, and other devices, as well as those that extend to provisioning new application systems. Note that here the arrow is drawn to show bidirectional interaction so

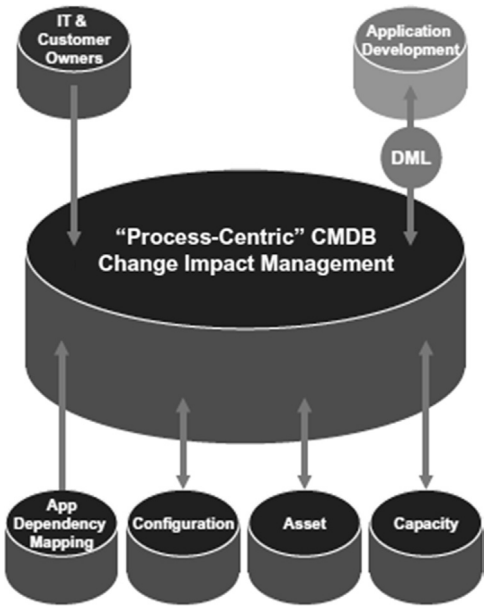

FIGURE B.6

A closer look at the process-centric CMDB, where most deployments begin.

that insights into larger service interdependencies can inform more point-optimized configuration tools while appropriate levels of detail and currency from these tools can update the CMDB based on policy.

- Asset—This term refers to asset-specific data, asset inventory tools, and/or an asset management database that houses financial information in greater detail than is appropriate for a core CMDB. Once again, the arrow is bidirectional so that appropriate asset-specific data can update the CMDB while service-relevant insights can help to deliver a baseline for more effective asset lifecycle optimization.
- Capacity—This term refers to one or multiple toolsets optimized for capacity planning analytics. Again, this is a critically bidirectional process ideally, in which capacity-related currency is maintained and service-related insights and interdependencies are shared.

Going clockwise from the top left in Figs. B.7 and B.8:

- Systems Performance—Refers to any number of systems management sources, from log files to time series data resident in monitoring and analytic sources.
- Events—Can come from any part of the application/infrastructure and once again, typically feed analytic engines.
- Service Impact Analytics—Link service interdependencies, change, and performance data together in a common analytic thread.
- Infrastructure Utilization—Refers to analytic tools that can harvest KPIs that impact shifting capacity requirements.

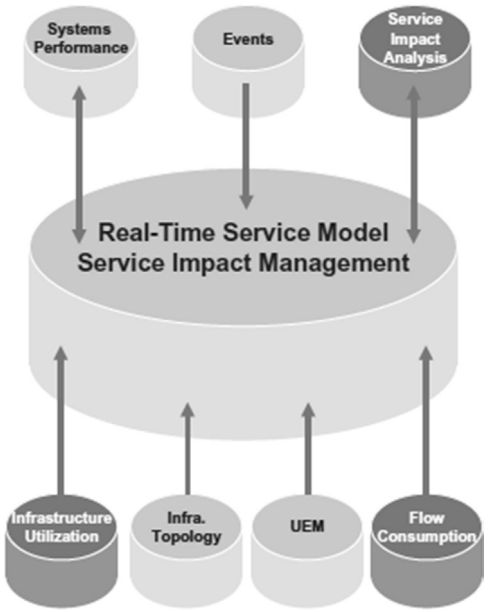

FIGURE B.7

Service impact management.

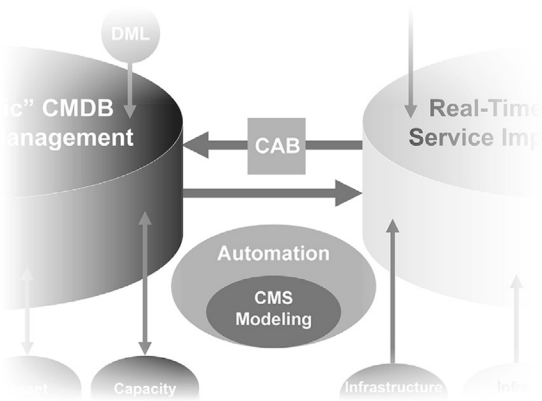

FIGURE B.8

Change advisory board (CAB) and automation.

- Infrastructure **Topology**—Keys on real-time awareness of the networked infrastructure (layers 2, 3, and above), as either integrated into an application dependency mapping tool or as a separate resource.
- UEM—Refers to **user experience management (UEM)** and what we call "the transactional stage," which can unify insights into application and business performance and business outcomes.
- Flow Consumption—Complements infrastructure utilization analytics with insights into how application traffic over the networked infrastructure may impact both performance and capacity-related issues. It may also provide insights into how and when applications are being used.

This is not meant to be a complete list, but it is a good departure point for planning.

Service Modeling: Where Logical and Physical Service Interdependencies Come Together

Fig. B.9 offers a logical depiction of the central requirement in the CMDB system: an adaptive service modeling system to bring together physical and logical interdependencies across the broader service

FIGURE B.9

Service modeling technology is at the very heart of CMDB system success.

landscape. This allows for data to reside in many different locations through a unified system of access. The beauty of associating logical (e.g., "customer" or "service provider") with physical infrastructure, middleware, and applications and their attributes is becoming increasingly more relevant as the extended IT organization becomes an ecosystem of partners, service providers, and, in some cases, suppliers.

SO WHERE DO I BUY ONE?

As inherently valuable as this modeling capability has become, no vendor actually markets just such a capability. To be clear, service modeling is often embedded in CMDB, ADDM, and service dashboard offerings, where it is presented as critical to functionality. However, most vendors approach service modeling with a tree-hugging reluctance once reserved for another innovative technology: **advanced operations analytics**. Service modeling also gained a bad name in many quarters due to the labors of customization, so it became necessary to make truly business-relevant service models appropriate for specific IT and business environments.

Nonetheless, some vendors actually worked to provide a cohesive and largely automated service modeling system that can support many elements, including imported insights on CIs and CI attributes from third-party sources. Moreover, there is a trend in some larger enterprises to adapt more fluid service modeling across multiple specific CMDB investments to create a broader unified system, a user-friendly "model of models" to help instantiate a more effective federated "manager of managers."

CMDB Systems: Making Change Work in the Age of Cloud and Agile is a unique combination of recommendations, industry insights, perspectives, and guidelines. It offers many diverse examples of how various approaches to making CMDBs and federated CMSs work. The overall goal is to provide both a landscape and a guide, drawing on past consulting experience, vendor dialogues and evaluations of vendor offerings, and deployment-related interviews. The book also leverages 10 years of research. In many places, the book is designed to be like a journalistic documentary, capturing many different first-person perspectives on issues, benefits, and recommendations for success. Such a rich and diverse set of sources place this book apart from all other books on the market today.