An integration model for organizing IT service management

J. Black
C. Draper
T. Lococo
F. Matar
C. Ward

To develop an architecture for information technology service management (ITSM) and design integrated solutions, it is necessary to establish a common understanding of the key conceptual domains involved in delivering IT services—organization, process, tools, and technology—and how they interrelate. This paper presents an integration model for ITSM practitioners. The model is a framework for organizing the assets that constitute an ITSM design. Using this framework, an organization can document the available set of IT services offered and understand how they are composed from finer-grained services delivered by internal or external providers. Different service designs are supported, depending on the service requirements and organizational context. This integration model is applicable to both in-house IT organizations and IT service providers, regardless of industry or size of enterprise. It may be used by management software vendors to describe the capabilities of their ITSM offerings and to align those with the needs of different customers, by consultants and integrators to develop engagement materials and solution offerings, and by IT service delivery organizations to document their service designs.

INTRODUCTION: NEED FOR AN ITSM INTEGRATION MODEL

In information technology service management (ITSM),[1] there are many models and frameworks created for various purposes.[2–9] This profusion of models, standards, and frameworks presents challenges to anyone leading an enterprise service-provisioning organization. Integrating the models, standards, and frameworks into seamless solutions requires common terminology and approaches for specifying service-oriented solutions. Without this, there can be little understanding of how separate initiatives should relate to one another or whether there are any gaps or overlaps.

This paper proposes an integration model for designing, developing, and deploying ITSM products and solutions. Its aim is to facilitate collaboration

among the many individuals and roles involved in creating and supporting these complex systems. The unique value of this model is that it offers leaders and managers of an ITSM organization a holistic model that covers the entire ITSM space. A key attribute of the model is that it supports the end-to-end specification of ITSM solutions, encompassing both product and service engineering.

The ITSM integration model that we propose starts with the premise that ITSM should be understood in terms of the services that are delivered. (Within ITSM, and throughout this paper, "service" should be understood as an overall IT service, such as software distribution or server support. The term does not refer to Web Services in a service-oriented architecture [SOA] context.) To allow for the highest degree of adaptability and parallelism across the development, sales, and delivery units, a clear distinction should be made between what a service must do functionally and how it is delivered. The complete description of how a service is delivered should encompass organization, process, tools, and technology. We show here how all of these concepts can be combined into a model that represents the entire ITSM space.

The integration model is grounded in the realities of running an ITSM organization and was developed to meet real-world needs. It is not, therefore, a formal model, but a pragmatic framework for use by practitioners in the documentation and communication of ITSM solutions. The model includes constructs that are specifically of value to commercial service providers, such as the sales and marketing aspects of services. The people who can benefit the most from using the model are those who lead enterprise IT service provision, either as executives or as IT architects.

The ITSM integration model is not intended to replace existing models; rather, it provides an overall framework that allows models from many domains to be cross-referenced and their relationships understood. The model shows how to develop and describe IT solutions, especially ITSM solutions that use structured design methodology and deliverables. It does not prescribe a specific solution or technology. The process and service domains are defined to a higher degree of detail in the model due to how critical they are to ensuring the success of ITSM. To define abstract concepts, such as *component* and *operational models*, we have relied heavily on the IBM Global Services Method and IBM IT architecture professional standards.[10]

MODELING APPROACH AND PRINCIPLES

Our goal in developing this model was clear. The practitioners within IBM Global Services, who are involved with the day-to-day delivery of infrastructure services, typically as part of a commercial outsourcing contract, are faced with a wide variety of frameworks and architectures, both within IBM[6,11–13] and throughout the industry,[2–4] each of which attempts to address part of the ITSM space. Although practitioners have little interest in the finer details of taxonomies or glossaries, they do need to know which asset will provide value to them as they work, and how. We created the integration model to help with this. We needed a structure that we could use to position all these initiatives, and our ultimate goal was to turn this into a repository that could provide a coherent source of information for practitioners.

Out of this basic need grew the major fundamental principles behind our approach to the model:

- *Consistency with existing practice is a primary concern*—Practitioners must be able to relate to the model and recognize individual parts of it. Thus, for example, we describe the process elements using the terminology "process-activity-task." This is because practitioners recognize this construct from structures such as the IBM Process Reference Model for IT (PRM-IT).[8,11] When describing the technology aspects, we used the names of deliverables from the IBM Global Services Method, primarily because they also would be familiar to our practitioners. We have followed this principle even where it leads to some inconsistency in the model or an unusual construct.
- *Clarity of expression has taken precedence over model formality*—We chose not to depict the model in formal modeling notation precisely because this would be inaccessible to many people in our target audience. We do keep a formal version of the model, but it is not widely published to practitioners.
- *Our major need is practical rather than academic*—It can be argued that what we have created is a metamodel (a model about models), an ontology, or merely a taxonomy. We make no attempt to

classify the model here; instead, it is presented as a practical tool to integrate the initiatives and assets around ITSM.

In this paper, we present the model as a series of informal entity-relationship diagrams (using untyped relationships as distinct from formal entity-relationship diagrams[14]). These entities can be thought of as work products or assets, and their connections are dependencies or inputs and outputs between those work products. In the simplest terms, an *entity* in this model is an asset needed to deliver an IT infrastructure service. The asset could be of any nature: In the application and product realm, these artifacts are code, software modules, and even hardware; in the services realm they consist of documents and specifications, such as requirements statements, designs, and service flow definitions. Later, when the model is translated into a repository, the entities are all represented by documents, which are either the assets themselves (in the case of designs or specifications) or a description of the asset and where to find it (in the case of hardware and software).

We first present the entire model in the same manner that we present it to our practitioners—as a simple diagram of boxes (entities) and lines (relationships). In the detailed descriptions of each domain, we provide more detail on the cardinality of the relationships, and in the supporting text we describe the nature of key relationships in more detail. The cardinalities of the relationships then indicate how asset types relate to each other. For example, the *many-to-many* relationship between technical component and deployment unit indicates simply that one technical component may be implemented by one or more deployment units, and one deployment unit may implement one or more technical components.

In developing the model we adopted several other key principles: simplicity, elaboration, and depth of model. The principle of *simplicity* means that we tried to make the model as simple as possible, with a focus on removing entities and combining concepts where possible. Our starting point was a set of integration models and ideas from a diverse set of contexts that ranged from component business modeling through process design specializations to technical solutions based on SOA. This gave us many different, overlapping ideas. Wherever possible we worked to bring concepts together in one entity, rather than justify why separate entities should exist.

One example of this is the *service element* (SE), a central entity that is the smallest unit of service that can be sold or charged to a customer by an ITSM provider. At the outset we had two separate concepts, an SE and an *atomic service* (the smallest unit of work that can be assigned to a provider organization for delivery). We unified these into the single entity SE by allowing involution (SEs can call other SEs) and by identifying SEs as either internal or external. Decisions such as these have business significance, and they make the model simpler. In this instance, delivery groups that are working to produce atomic services now have to work directly with the groups preparing the customer-facing services catalog to ensure that a consistent set of services is specified.

The second key principle is *elaboration.* Most of the key entities in the model can exist at different levels of elaboration. For example, a high-level operational model could consist of a collection of more detailed operational models, and the operational models could, in turn, be decomposed into deployment units and then physical nodes and connections. As a general rule, we have not identified separate entities for different levels of elaboration. *Operational model* as an entity should be assumed to include all levels of an operational model, from the conceptual level through the specified level to the physical level. *Process flow* similarly contains both the logical and physical elaborations of a process.

In some cases, however, different levels of elaboration have been identified. *Process* and *activity* are clearly different levels of elaboration within the process model; however, we have retained the separate terms because the concept of process–activity–task is firmly embedded in process reference models such as PRM-IT. *Information model* and *logical data model* could also be seen as elaborations in the data-and-information domain. However, the information model contains unstructured data that does not appear in the logical data model.

Decisions on whether to show levels of elaboration explicitly within the model were not guided by any rigid modeling rule or logical criteria; they were

made on the basis of judgments of which entities were useful to describe the ITSM space, taking into account existing assets, practices, and terminology, and ease of understanding by our practitioners.

The last principle, *depth of the model*, means that we have not attempted to take the model to the lowest level of detail. Its primary role is to help us understand and rationalize other related initiatives that provide the detail in ITSM service delivery.

At times, though, our adherence to the principles of familiarity and ease of understanding led us to include some entities that (at face value) were quite disparate in their levels. For example, *task* exists at a low level in the process hierarchy, and it may be surprising to see it on the same diagram as higher-level constructs, such as component model or business component. However, for the process specialist, it is both powerful and useful to see the familiar notation of process–activity–task clearly related to the other parts of the model. In choosing the level to go to in each domain in the model, practitioner accessibility remains the overriding criterion.

In concluding this section, we recognize and connect this work to other peripherally related models, such as ontologies,[15–18] Unified Modeling Language** (UML**),[19,20] Model Driven Architecture** (MDA**),[21] and SOA.[22–24] These models offer tremendous expressive depth and the potential for inference or tool-assisted realization of facets of our integration model. However, it is not likely that our audience would appreciate these structures, and indeed, their formal syntax may actually be an obstacle to their being understood. In contrast, the principal value proposition of the integration model is to provide a single structured framework (indicating key concepts, terminology, and relationships in a managed environment) that has wide appreciation by a variety of practitioners.

OVERVIEW OF THE MAJOR DOMAINS

The integration model is centered on the concept of services. ITSM is about the definition and delivery of IT services and the management of the organization that provides the services. The model provides a structure that allows us to describe *what* the services are (the service definition aspect), and it links the service definition information to *how* the service is delivered (the service delivery aspect). These two aspects are further broken down into a total of six domains, shown by the legend of ***Figure 1***.

The service definition aspect contains two domains that describe what the services are: the *service offering* domain is for the purpose of marketing and selling the services; the *service provision* domain identifies the base set of services and groups them into a hierarchical structure. The service offering domain contains entities that facilitate combining services in different ways in order to support customer demand and exploit marketplace opportunities. This domain is clearly of great importance for a service provider company, but may be less relevant for an IT service-provision division that is part of a larger enterprise.

The service delivery aspect contains four domains that describe how the services are delivered. The *process* domain describes the processes to be followed when responding to service requests, events, and triggers. The *organization* domain identifies the locations, people, roles, and skills required. The *tools-and-technology* domain describes the applications and systems used to deliver a service, and the *data-and-information* domain describes the data and information that need to be managed in the delivery of the services.

The model is represented as an informal entity-relationship diagram. Figure 1 identifies all the entities proposed for the model and shows how they relate to each other. At this overview level, cardinalities are not shown, but these are included in the detailed descriptions of the domains that follow.

Within this model there are two central entities: An SE, which defines what the individual services are but not how they are delivered, and a *service design* (SD), which defines how each service is delivered by using a combination of people, processes, tools, and technology. The complete set of SEs represents the complete set of capabilities within the ITSM space. Each SE must have one or more SDs, which represent the alternate ways in which the service is delivered. For example, the SE might be patch management, supported by different SDs for different platforms (such as patch management—IBM zSeries* and patch management—Linux**).

Finally, the model allows different perspectives of the entire ITSM domain. The process domain

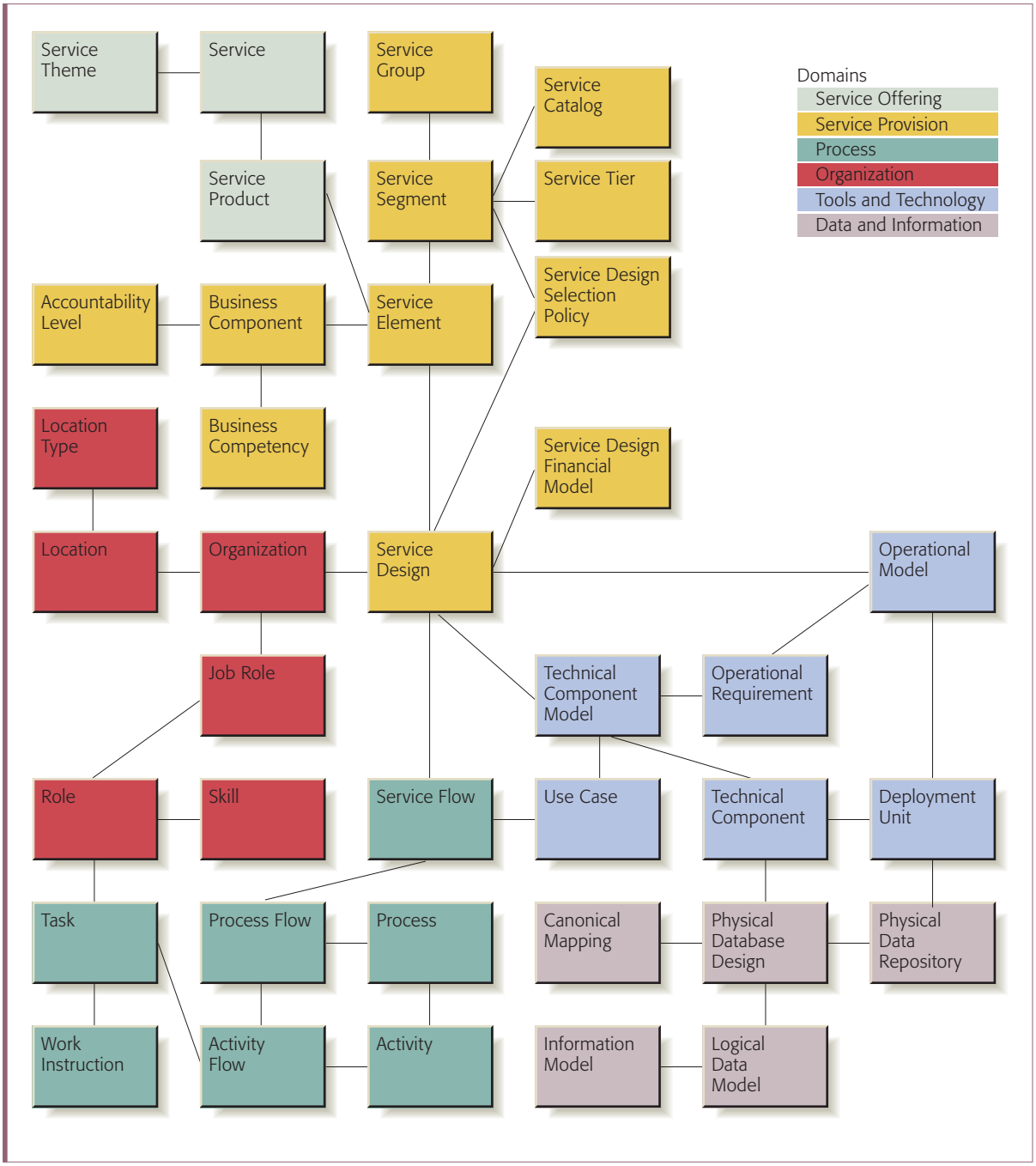

Figure 1
The ITSM integration model with its six domains expanded to illustrate additional detail

accommodates an overall process reference model (such as PRM-IT), but also has links that show how individual activities and tasks support specific SEs. In the tools-and-technology domain, an overall component model shows how all of the tools and technology for service management fit together, and the links for individual SDs show how individual components support the delivery of specific SEs.

SERVICE PROVISION DOMAIN

The service provision domain is the central domain that describes the services provided by the ITSM

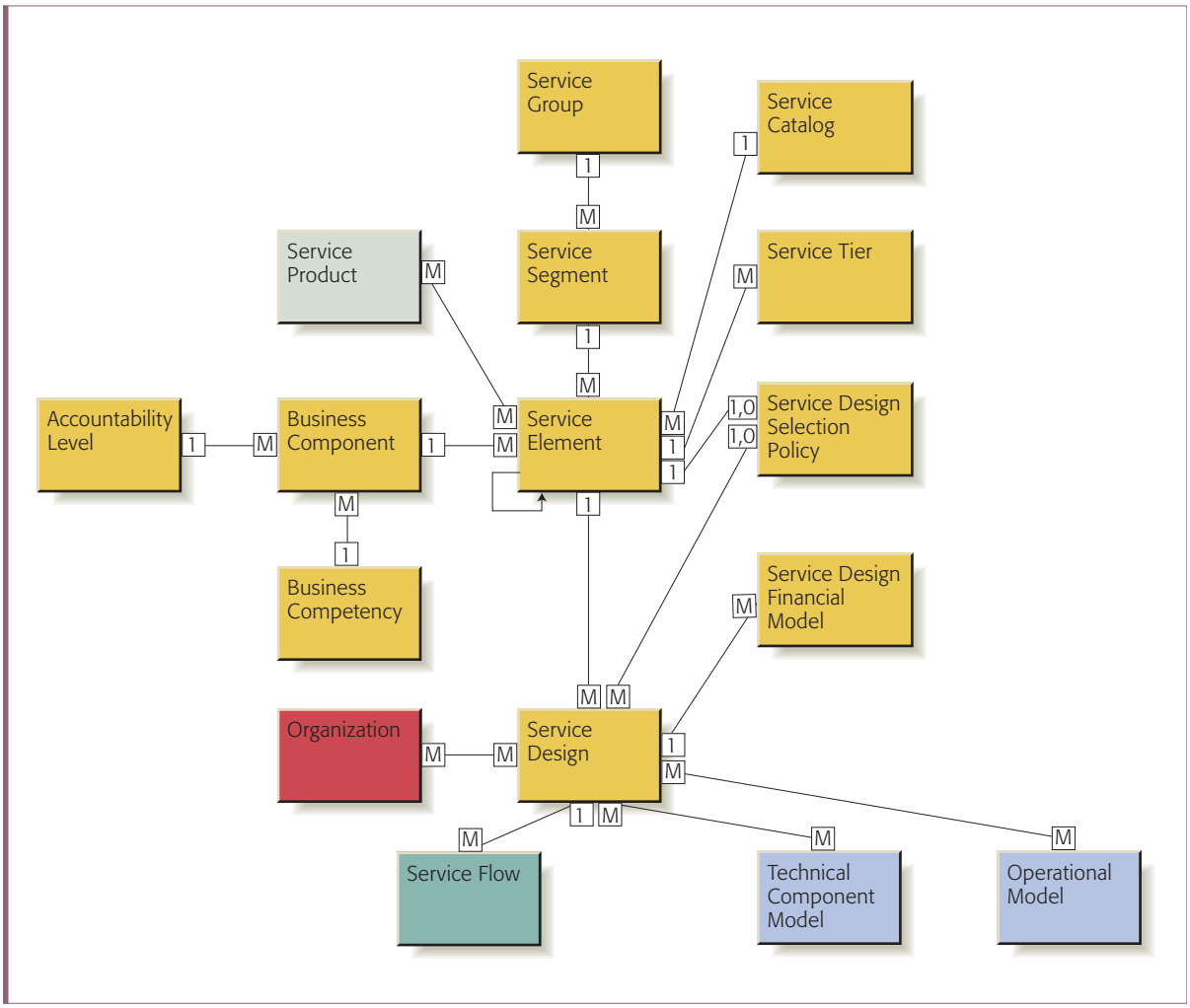

Figure 2
The service provision domain of the ITSM integration model

organization. The entities in this domain and their relationships are shown in ***Figure 2***. In these more detailed modeling diagrams, we have indicated cardinalities as 0, 1, or M (many) in the conventional way. Thus, in Figure 2 for example, the relationship between SE and SD should be read as: "An SE relates to many SDs; an SD relates to just one SE." As a further example, consider the SD selection policy. Each SE may relate to one or zero SD selection policies, and each SD selection policy relates to just one SE, but many SDs.

The starting point for any ITSM organization is to understand and classify the services it provides. The key entity in this respect is the SE, the basic building block used to construct the service portfolio. It represents a logically bound unit of IT service defined to meet a customer requirement. The SE represents the lowest level of service for the purpose of determining cost and price and for assignment to a service provider.

To manage the set of SEs, it is grouped into a three-layer hierarchy. A collection of related SEs is labeled a *service segment,* and collections of service segments are called *service groups.* This hierarchical structure contains no gaps or duplicates: Every SE belongs to one and only one service segment, and each service segment in turn belongs to one and only one service group. The choice of three layers is a pragmatic one, based solely on the number of SEs to be managed: We have identified approximately

220 SEs that make up the ITSM space. To manage these, we have created 14 service groups (for example, asset services and data network services), and 43 service segments (for example data network planning and asset management) that contain between two and 15 SEs.

This structure might be a close match with the ITSM organization or process structure; however, there is no defined link, and the hierarchy exists only for the purpose of classifying and structuring the SEs. Thus it will be of use primarily to those managing the ITSM organization who need to understand the complete set of services being offered.

SEs should be classified as external or internal. External SEs are those that are visible and used by customers of the ITSM organization, for example, a password reset service. Internal elements are visible only internally: An example might be the communications services provided by an enterprise services bus. SEs may use each other to fulfill their responsibilities; that is dependencies may exist between SEs: One SE may use another one to fulfill its responsibilities. However, an internal SE should always be used by an external service in some way; otherwise, it is redundant.

This hierarchy down to the SE level describes only what the services are. Behind each SE sit one or more SDs; that is, an internal view of the service that describes how it is delivered. The SD is an overview and summary of the detailed delivery description contained in the organization and service flow and in the technical component and operational models.

One or more SDs must exist for each SE, and each one must implement the full scope of the SE it supports. Where more than one SD exists, they represent alternative solutions for one of the following reasons:

- Different solutions are needed for different platforms (e.g., software distribution for IBM zSeries servers and software distribution for Microsoft Windows**).
- Different solutions are required to meet different levels of service (see service tiers below).
- Different solutions are required to meet customer needs (e.g., software distribution based on Microsoft Systems Management Server and software distribution based on IBM Tivoli*).

An SD actually contains no unique information: It is a summary of detailed technical, process, organization, and data solutions held in the design of other domains, and its purpose is solely to give an overview of how a specific SE is delivered in practice. Nevertheless, the SD is a key entity in the overall model. Taken together, the full set of SDs represents a summary of the entire capability of the ITSM organization. Although we have placed the SD in the service provision domain, it could be argued that, because it forms the bridge between what the service is and how it is delivered, it belongs outside all domains.

Other entities in the integration model support the SE and SD structure. The *service catalog* is a summary list of all the services provided by the ITSM organization. The *service tier* captures the common requirement to have the same basic functional service but with different service levels. The levels might be higher or lower availability targets, response times, fix times, or others. Where there is more than one SD for a single SE, an *SD selection policy* contains the criteria for choosing between them.

Costs associated with the service are held at the SD level using the entity *service design financial model*. There is a one-to-many relationship here, as different countries, geographies, and organizations may have different cost models for the same basic solution.

The final group of entities in this domain provide a link to the IBM Component Business Modeling (CBM)[25] framework. CBM is unlike component modeling for software. A *business component* is defined by its potential to operate independently as an aspect of the enterprise that offers products or services. It can be contrasted with a *software component,* which is defined in terms of its technical characteristics, such as encapsulation and whether it is context-specific. Conceptually, therefore, a business component is very similar to an SE; they both describe some capability or service that can be delivered by a combination of people, processes, tools, and technology. Also, they both incorporate the concept of nonoverlapping, provider-assignable functions (which is why the model proposes that an SE can only be part of one business component). However, the level of granularity is very different. SEs are the lowest possible level of service that can

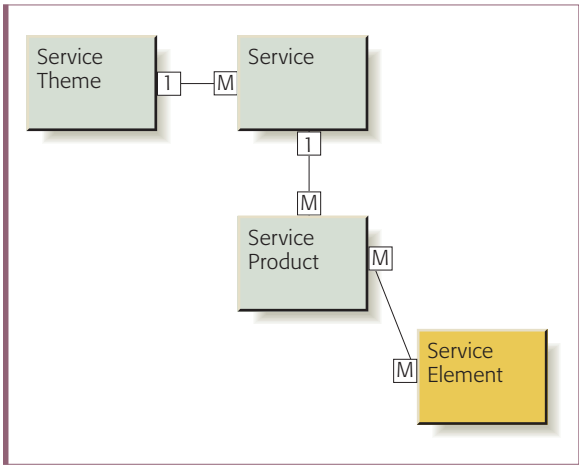

Figure 3
ITSM integration model: service offering domain

be priced, whereas business components are higher-level constructs used primarily in business analysis and planning. The integration model proposes a link between these two concepts: A business component should be the same as a collection of SEs with their associated SDs.

Within CBM, components are grouped into business competencies: A *business competency* is a large business area with characteristic skills and capabilities. Business components are further classified by their *accountability levels,* which characterize the scope and intent of activity and decision making. The three levels used in CBM are directing, controlling, and executing. Details of these terms and concepts can be found in CBM documentation.[8]

SERVICE OFFERING DOMAIN

The service offering domain provides a set of entities that are valuable to those who market and sell ITSM services and offerings (***Figure 3***). As such, they are likely to be more relevant to commercial ITSM providers. There are three entities in this domain—service theme, service, and service product—linked to the modeling entity SE, which provides the bridge to the service provision domain. Unlike the service hierarchy in the service provision domain, these entities do not attempt to structure or organize the set of SEs. Although they build on the same basic SEs, they may be created, changed, and discarded dynamically to respond to changing marketplace and customer needs. They are defined as follows:

- *Service theme*—A broad, strategic, go-to-market messaging construct designed to capture market interest in the service provider's ability to assist customers in top business and IT challenge areas.
- *Service*—Comprised of service products or SEs, or both, that address a broad range of customer requirements within a designated market segment.
- *Service product*—The configuration and packaged integration of one or more SEs or other service products consistently needed to fulfill a common and repeatable set of customer IT service requirements.

There is a close relationship between the service-offering and service-provision domains: An organization can offer only services that it knows how to provide. In turn, that relationship translates into a multiple threaded linkage with the domains associated with the "how to deliver" through SDs and associated entities: There are multiple ways in which a service may be delivered, based on factors such as location and required quality of service.

Cost (and consequently market price) is an important parameter of interest for this domain. However, a key feature of the model is that costs and prices should be calculated based on the cost of service provided, and this can only be done correctly at the SD level. This is why the only financial entity (service design financial model) appears in the service provision domain, not here. Of course, sales and marketing professionals will obtain the cost information they need by referring back to the basic building blocks (SDs and SEs) that make up their proposals.

ORGANIZATION DOMAIN

The organization domain consists of six interrelated entities that describe the structure and roles in a service-oriented ITSM provider (***Figure 4***). Three entities describe organization and location information:

- *Organization*—Part of the overall enterprise that delivers one or more SDs. Involution on this entity allows multiple organizational levels to be described. At the highest level, *organization* is the reporting structure and the relationships between units accountable for delivering the service. IT organizations should align with the needs of the enterprise and balance the efficiency of sharing IT services across lines of business with the business

growth potential of line-of-business-focused service differentiation.

- *Location*—A specific place where people and machines are sited for the purposes of delivering service. Multiple organizations can be sited in one location, and one organization may span multiple locations.
- *Location type*—Used to classify locations. A location type is a generic type or class of location; for example, data centers, call centers, and administrative offices.

The remaining entities describe individual jobs. A *role* is the smallest collection of responsibilities given to an individual so that that person will perform a number of defined tasks. Roles rely on one or more *skills* and may be combined for a specific person to create his or her job role. As an example, the job of delivery center support representative may combine two roles: that of customer service representative (taking support calls) and that of server administrator (carrying out server administration tasks). The server administrator role may require certain skills, for example, UNIX** administrator level 1.

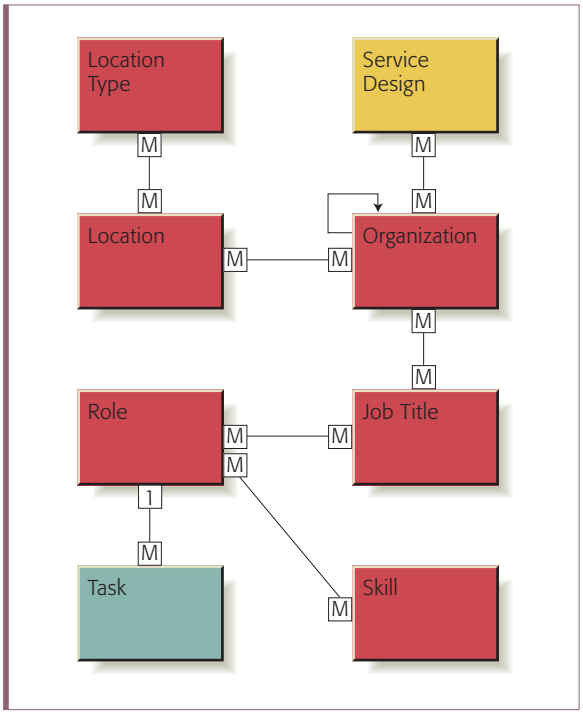

Figure 4
ITSM integration model: organization domain

The organization domain interfaces directly with the process domain and the service domain. Both process and service development and deployment throughout the IT organization require clearly defined roles with commensurate decision-making authority and performance accountability. The organization domain defines and coordinates these roles within the IT organization structure.

PROCESS DOMAIN

The process domain is key within the ITSM integration model (***Figure 5***). Too often new service management tools and automation technologies receive insufficient focus and management attention. Yet typically, 50 to 70 percent of total budget spent on ITSM is on people, and the consistency and reliability of processes is critical in making IT delivery as efficient as possible.

Three basic concepts underpin the process domain. A *process reference model* defines the complete set of processes that are needed for ITSM by providing a hierarchical, nonoverlapping breakdown of the activities required. Within the process reference model, the processes are organized into disciplines, such as problem management and systems management. The principal use for the process reference model is as a reference point or checklist, enabling IT managers to confirm that they understand all the necessary process disciplines and have implemented them appropriately. The IT Infrastructure Library** (ITIL**) framework is perhaps the most common industry-standard process reference model, and ITSM conforms to ITIL structures and terminology (see below).

A *process flow model* follows the same hierarchical structure as the reference model, but develops the processes further, breaking activities down into *tasks* linked together by *activity flows,* and specifying the *process flows* that link together activities. A task is defined as the smallest unit of work that can be assigned to a role; that is, the entire task would be assigned to a single person for execution. Each task may be supported by *work instructions* that guide the user on how to perform the task. The process and activity flows can be used as the basis of actual process design.

A series of *service flows* define the processes to be implemented in practice. Each service flow is created in response to a specific event or trigger, for

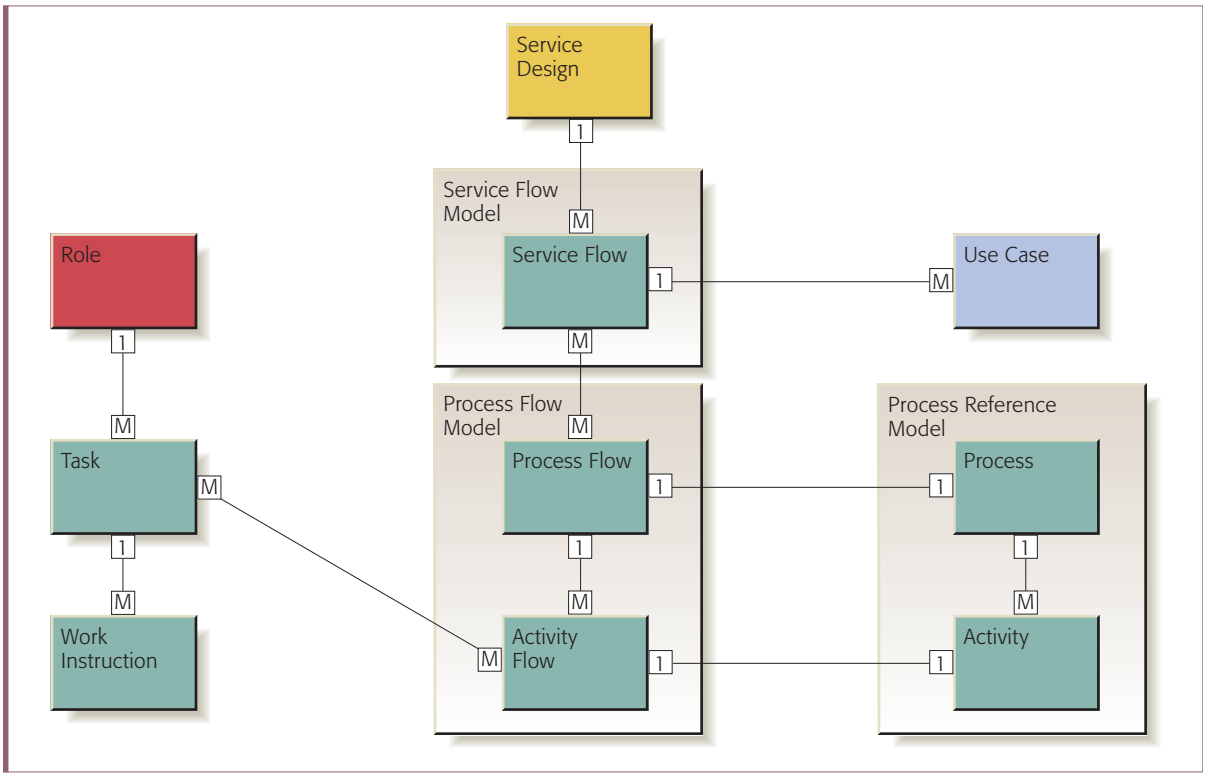

Figure 5
ITSM integration model: process domain

example, a system condition such as "disk full" or a user event such as a help desk call. A service flow must cover all of the activities required to handle an event and may span several disciplines in the process reference model. Service flows can only be designed by considering the events or triggers that need to be handled.

For example, servicing an event might require tasks from the release-management process and activities from the change-management process linked together in an overall flow. Models of a process flow help ensure that individual tasks are linked together and executed in a consistent fashion, but only the service flow shows the complete set of steps required to handle the event. Unlike the disciplines in the process model, the complete set of service flows contain overlap and cannot be organized into a single hierarchical structure.

The key to using these various process artifacts effectively is to focus on the service events or triggers that need to be managed. Process design starts from understanding the complete set of events and using the scenarios (service flows) to identify the actual work that must be done to handle each of them. As process definitions mature, the process reference model and the service flows need to be maintained and updated. IBM and industry assets (such as PRM-IT, the IBM Tivoli Unified Process (ITUP), and ITIL) form useful starting points and provide much useful reference material for this task, but all organizations need to tailor the service and process flows to their own specific circumstances.

TOOLS AND TECHNOLOGY DOMAIN

Once an organization has decided on the IT services it will deliver and has defined them in the service domain and then decomposed those into flows, processes, activities, and tasks in the process domain, technological solutions to achieve the appropriate level of automation must be documented. That is the purpose of the tools-and-technology domain (*Figure 6*).

The use case, technical component, and operational models are the key entities used to describe architecture in the tools-and-technology domain.

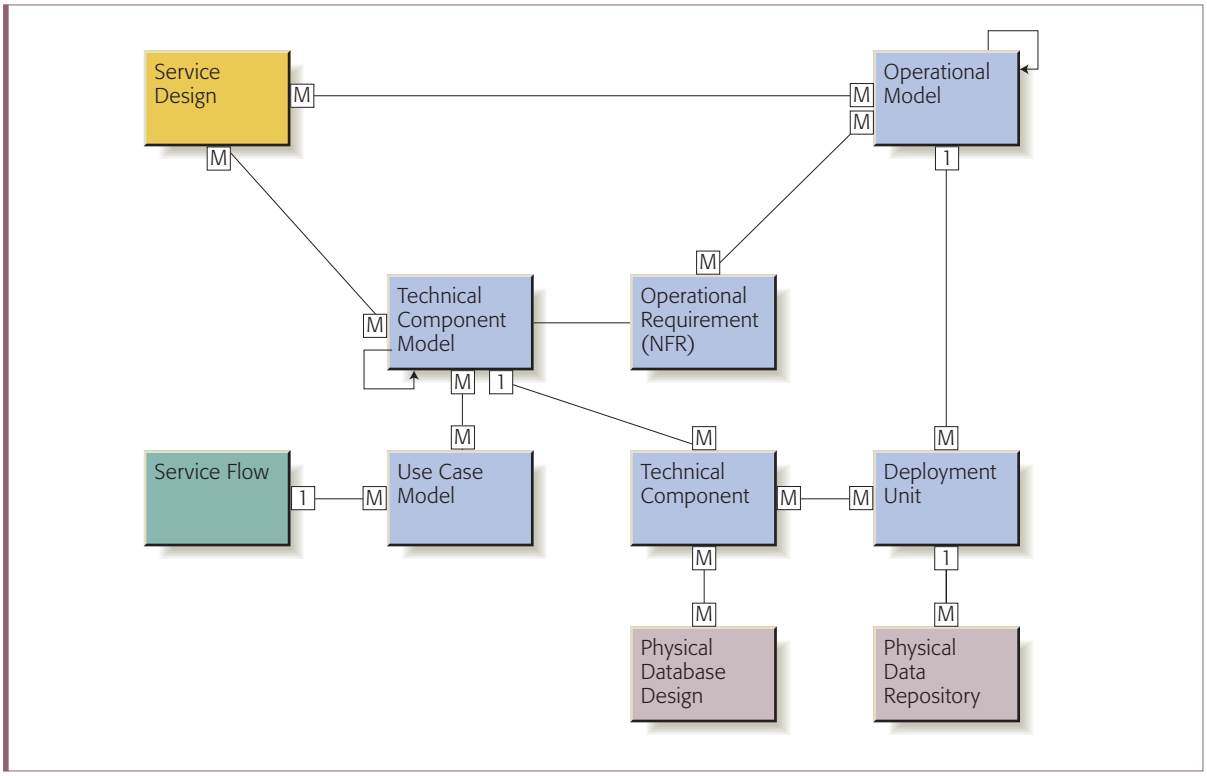

Figure 6
ITSM integration model: tools-and-technology domain

The *use case model* gives the context or process flow linkage and the roles of the users who will interact with the service as designed. Use cases are therefore closely related to (and may be derived from) the service flow for a specific SD. Capabilities and test cases are designed to match or cover these use cases.

The *technical component model* is where the designer or IT architect captures the functional characteristics of what the solution must do, that is, the functional specification of the system. It breaks down the SD into manageable, modular pieces, called *technical components,* to facilitate development, testing, and deployment of the service.

The *operational model* captures how the service will be delivered through specific technologies (platforms and networks). The operational model must address *operational requirements* (sometimes referred to as nonfunctional requirements or NFRs), including how the service must perform, how secure it must be, and what availability criteria it must satisfy. Technical components are then grouped into zones for deployment and placed onto deployment units within the operational model.

No company or organization can afford to be an island, so the assets in this domain, as in others, need to make use of available standards to promote optimum levels of interoperability. To do this in a cost-effective way, the service management architect should use standardized assets. This can only be done if assets are documented and categorized in a way that allows them to be searched, reviewed for "fitness to purpose," and retrieved for tailoring, once selected. The integration model provides an ideal scheme to categorize and link content.

ITSM solutions may be created at varying degrees of elaboration for multiple audiences. As ITSM designs proceed from the concept phase to implementation, the designs are developed from the conceptual level to the logical and specified level to the physical product-specific level of elaboration.

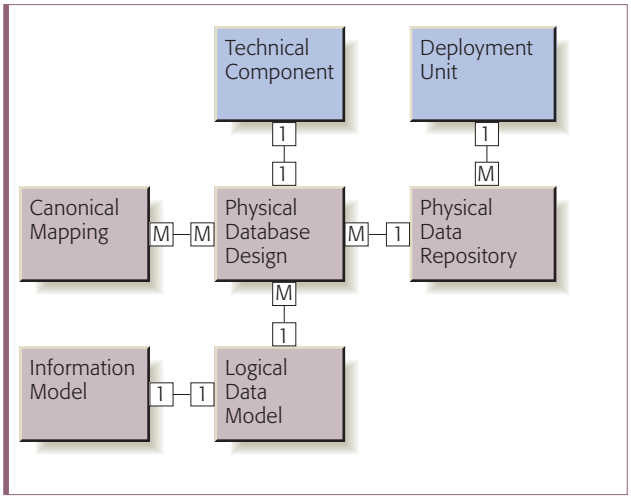

Figure 7
ITSM integration model: data-and-information domain

Product, service, and customer perspectives are created by architects who work in each of those environments. The integration model, once built, supports this work breakdown by formalizing the discrete work products that must be created and determining how they are linked for successful integration. For example, application architects focus on the component model and functional design, and infrastructure architects focus on the operational model and aspects associated with it, which often include the service management disciplines. An SD document links all these pieces together. By focusing on these aspects in parallel, large projects can proceed more quickly.

This level of architectural rigor is familiar to anyone who has worked in business systems application development, but may seem to be overkill to specify an ITSM environment. It is not. It must be remembered that most products in this space today are multitier applications made up of Web servers, application servers, and databases, just like the business applications they manage and support. Additionally, most ITIL-aligned service management processes, such as problem management, capacity management, and service-level management, involve all these "technology towers," in addition to others, such as networking, storage, and security. By applying this level of rigor, we can ensure these processes are implemented in a seamless, "cross-tower" fashion.

DATA AND INFORMATION DOMAIN

The data-and-information domain describes the key data model elements that represent managed entities in a comprehensive ITSM solution. It also describes how this information is used by the key service management entities and how the data-and-information domain contributes to the other domains.

Data and information represented in the integration model are described by five entities as shown in *Figure 7*:

- *Information model*—The single information model for ITSM summarizes the information managed in the ITSM space. This information model reflects both unstructured and structured data and represents the totality of information available. It covers all types of data: structured data that will be implemented in a database and unstructured data, for example, knowledge bases. Ideally there is just one information model covering the entire space.
- *Logical data model*—A single logical data model, which provides a detailed decomposition of each of the major areas described in the ITSM information model, is associated with the information model. Whereas the ITSM information model describes all information, including information that is unstructured (such as knowledge bases), the logical data model provides entity and relationship information for all the actionable data elements; that is, the information that can be accessed through programmatic means. Ideally there is just one logical data model for the entire ITSM space.
- *Physical database design*—This implements a fragment of the logical data model and can be modified as required for performance considerations or to support canonical models. From an implementation perspective, the logical data model is mapped into one or more physical database designs.
- *Physical data repository*—A physical data repository implements one or more physical database designs. Each physical database design is mapped to a physical data repository that provides concrete realization of the deployed design.
- *Canonical mapping*—A canonical mapping is a standard data design or schema that constrains or dictates the data design for a physical database design. The designs, which reflect some facet of

the logical data model, may be constrained in accordance with one or more canonical mappings, such as those articulated in standards groups or the commercial product being used for that particular function.

It is obviously highly desirable to have a single information model and an associated logical data model from an overall service management perspective. However, it requires a governance structure and design authority for its maintenance, and any overall data model is likely to be at odds with the physical database designs for individual solutions. This is because the systems used in ITSM for such functions as configuration, change, problem, and systems management are predominantly commercial packages rather than custom applications. It is therefore unrealistic to expect that the information model and logical data model will do more than map the various physical database designs and support identification of gaps and overlaps. Nevertheless, such mappings are important in order to retain the best possible control of the multiple data repositories in the ITSM organization.

Here are typical use patterns based on the entities within the data-and-information domain:

- A valuable entity in the data-and-information domain is the logical data model for the managed entities themselves. An example of this would be the Tivoli Common Data Model (CDM).[26–28] The logical data model is used by many ITSM components to understand resource and relationship information in the deployed environment. It can also be used by other business components to provide insight into the ITSM space; for example, the Tivoli CDM[12] embraces business applications and business processes and shows how they relate to ITSM entities. This model and programmatic access to it may be used in SOA-enabled environments[29] and for autonomic control enablement.[30,31]
- The information model provides a high-level overview of all the information in the ITSM space and is therefore a valuable resource for new IT outsourcing accounts, enabling service offerings and operational management of existing services.
- The canonical mapping entity provides an industry-wide (and hence interoperable) perspective on the representation of managed-entity details. It allows us to capture and compare the data models that exist in different commercial products. This entity also provides the opportunity for the concise specification of concepts in the data-and-information domain that are common across managed-entity subdomains.
- The physical database design provides service developers with the necessary insights to design and implement entities and databases to support new and existing services. The physical database design may also be used to identify similarities and gaps when comparing and contrasting products that provide equivalent service.
- The physical database repositories are used by service applications and service management applications to accomplish their stated functions. Metadata regarding the physical database repositories is used by service support teams to manage the applications and services that use the repositories.
- In some organizations, more typically organizations that build tools and products as opposed to ITSM service providers, the logical data model may be used as the basis of their data modeling and interactive design in order to integrate products.

USING THE INTEGRATION MODEL IN PRACTICE

In this section we discuss the practical applications of the ITSM integration model. Although we draw primarily on experiences within IBM, the principles apply to any enterprise ITSM provider.

A repository of ITSM assets

Our prime motivation for creating the model was to force an integration of related ITSM assets and initiatives to support our IBM practitioners in the delivery of IT services. Few practitioners can or should understand the full detail of the model: Our main route to help practitioners was through a repository of ITSM assets.

The major features of the repository are the following: It is a content management system based on IBM Lotus Notes* technology that implements the integration model by specifying a set of documents and prescribed linkages. A user portal, available through the company intranet, enables users to access those documents through a graphical user interface, and it offers a governance system that

controls how content is loaded into the repository and how it is used.

The practitioners who define our standard service offerings are considered *creators* of content in the repository. For each service they are asked to provide a linked set of documents that defines what the service is (the SE) and how it is delivered (the SD and supporting documents, such as service flow, organization, component model, and operational model). The governance system ensures that these documents meet required quality standards and are subject to peer review before publication. In general, we do not insist on complete population of all document types, but prescribe a minimum set of linked documents that must be provided, with optional additional material. Links to other Web sites and data sources are also supported within the documents.

Content *users* are those ITSM delivery practitioners who have to understand and implement our standard services. This includes architects, technical and process specialists who work with specific customer accounts, and management staff who need to understand our service organization. It also covers sales staff preparing new outsourcing proposals for customers.

The user portal screen broadly follows the domains in the integration model (with the exception that we combined the tools-and-technology and data-and-information domains into the technology framework view to simplify the user interface). From the top-level view, a user can navigate to the service framework view and the technology framework view. The service framework view presents our service catalog in a format that users can understand. From this screen, users can drill down into individual SEs. The SE documents contain links to the supporting SD, and so on through the model. Using this approach, users can explore any aspect of a service—the tools, the organization, or the technology—through a series of links from the service catalog.

The technology framework view provides an alternative view that shows how all of the technical components fit together. This overview is particularly interesting to an architect or technical specialist who needs to understand how the technical aspects of all SDs fit together. From this view, they can drill down to individual technical components and use the links in the model to explore such areas as related SDs and process details. Similar framework views are provided for the process and organization domains.

There are many benefits that have flowed from the implementation of the ITSM repository:

- It has implemented a clear book of standards, which sets out preferred services, processes, and technologies in a structured way. This is a prerequisite for standardization of our services and technologies.
- It enforces a complete description of any particular service by covering the service definition (through the Service domain) and all aspects of delivery of the service (organization, process, tools, and technology).
- It provides a single point of reference for all those involved in the delivery of a service.
- It provides an integration point for those who need to define all aspects of a service. In particular, any group wishing to start a new initiative in one particular domain (for example, a revision of the process standards) must show how they will implement that within the repository model and how it will be integrated with other domains, such as the service catalog.
- The higher-level framework views also prove valuable for discussions with customers of our services. It is common for an account or customer to specify different tools and processes for a particular service; the framework views are very useful in assessing and managing the impacts of these changes on the overall architecture for the account.

In summary, implementation of the repository based on the integration model has allowed us to manage the true integration of our services, which was a key goal for our modeling work.

Case study: Mapping initiatives to the integration model

A key purpose of the integration model is to enable different initiatives within an organization to be related to each other and rationalized. Shortly after completing the integration model, we used it in practice to examine a number of proposed initiatives

from different parts of our ITSM service delivery organization. As an example, we consider just three of those initiatives:

1. A proposal to implement an automated service-request and service-delivery management solution
2. A proposal to build and publish a standard catalog of services to support our sales and marketing force
3. A proposal to converge the delivery processes with those described in the latest release of IBM Tivoli Unified Process

From the start, it was evident that there was some overlap among these initiatives. Because it was difficult to assess what the overlap was, the discussion had to start with each initiative by asking basic questions concerning taxonomy and intent: What is meant by *service*? or *process*? Without the integration model, a complex series of discussions would have been required just to understand and agree on the areas of overlap, let alone rationalize them.

Using the integration model, we were able to address this issue much more rapidly. We gave a briefing on each model initiative, and in the space of a short discussion (lasting no more than one to two hours in each case), we were able to map the initiative to the model. ***Figure 8*** is the result.

It was immediately very clear where some of the major areas of overlap lay. It is obvious that the service-delivery management initiative, which was focused mainly on automation technology, must base its services and service requests on the set of SEs being documented by the service catalog project. It is also clear that the service-delivery management initiative can and should link the back-end delivery processes to the ITUP if the two are to be consistent.

Both of these examples may seem obvious; however, none of the individual initiatives had anticipated them. The use of the model was essential as a catalyst to bring these initiatives together. In practice, we extend the study to a total of seven initiatives from different groups. The integration model proved extremely effective in identifying gaps and overlaps rapidly, in a noncontentious way, and without long hours of confusion as terminology is explained and debated. As a result, we were able to set up a number of focused working groups to improve the integration among our projects and rationalize them.

This is an example of how the integration model could be used in practice by any ITSM provider organization. Any such organization is always running a number of major initiatives to transform and update their delivery capability. They also use several frameworks or architectures to help control and manage the technical, process, and organizational aspects of their work. The ITSM integration model provides a management tool that assists in controlling the initiatives and architectures.

Any ITSM organization can use the integration model in three ways:

1. For every major initiative, architecture, or framework proposed for adoption in the organization, map it to the integration model and ask: Do we understand which entities it covers? What does this tell us about how this initiative or architecture fits with others? Is there overlap or inconsistency to be eliminated? Are there interfaces we need to build to ensure consistency?
2. When the complete set of architectures and initiatives are mapped, consider the overlap coverage. Are there any gaps that would represent areas of the delivery function that we do not currently understand or manage? Are there overlaps that represent duplicate work and generate unnecessary expense and inconsistency?
3. Above all, look to the heart of the model. Does our organization understand and describe the full range of services we offer (the SEs) and our standard solutions for delivering them (the SDs)? Having a clear understanding of these fundamental capabilities is a vital first step to creating an overall set of coherent architectures and programs.

CONCLUSION

We believe that the ITSM integration model provides a coherent, holistic analysis of the ITSM space. It is firmly grounded in the reality of ITSM organizations. Therefore, in places it does not rigorously follow ideal modeling practice because our guiding principle was to produce a model that our users could relate to and use.

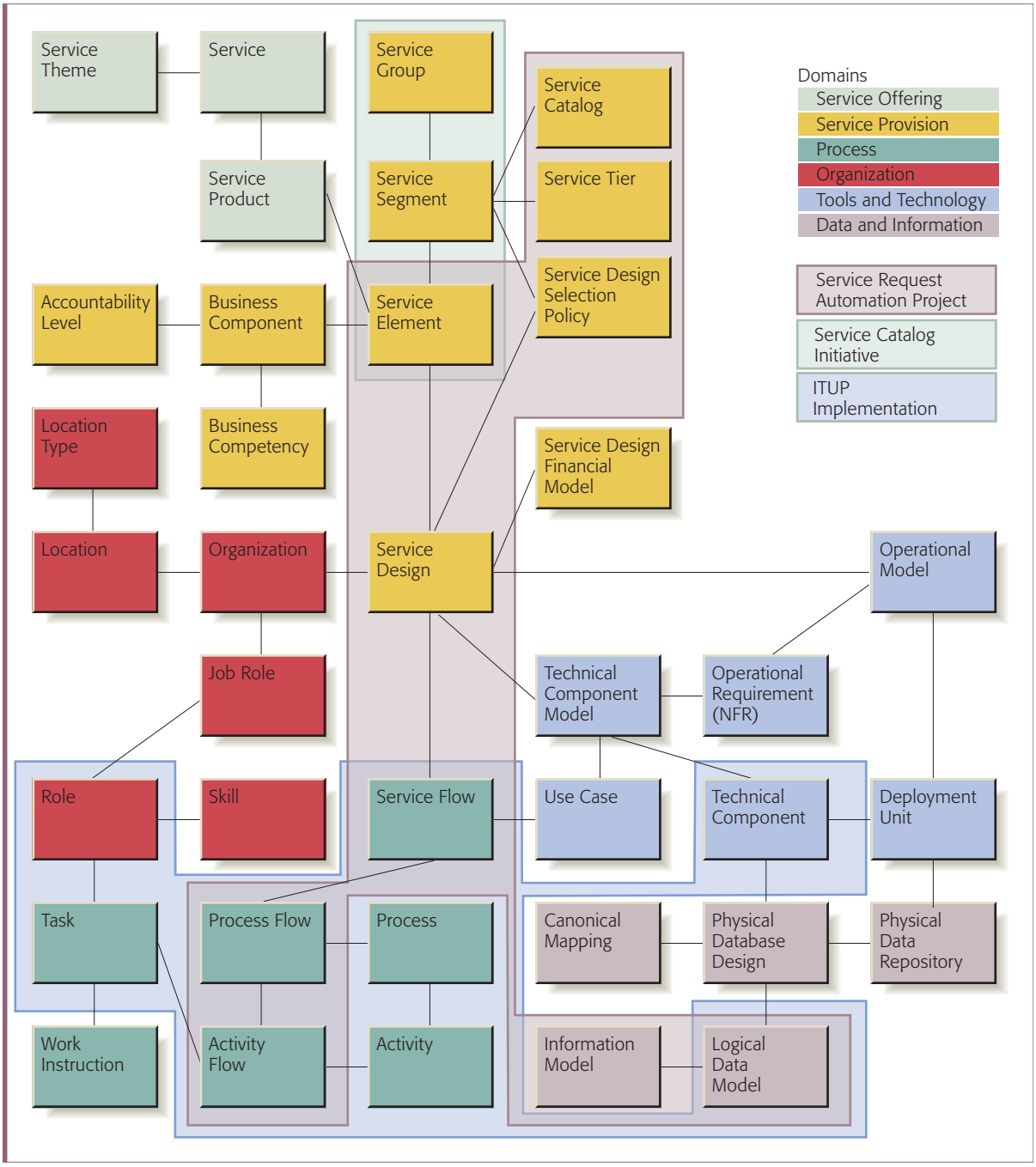

Figure 8
Example of mapping three initiatives to the integration model

We have shown two clear examples of the value of the integration model. We have based our repository of standards on the model, and this has driven integration among several different disciplines within the organization and forced us to describe our services in a consistent and comprehensive way. It is a key tool for the integration of our work. We have also shown how the model can be used to position and rationalize overlapping change initiatives within an ITSM organization. We believe that any ITSM service-provider organization could use the model in a similar way, as a valuable management tool to organize a program of change and improvement.

*Trademark, service mark, or registered trademark of International Business Machines Corporation in the United States, other countries, or both.

**Trademark, service mark, or registered trademark of Object Management Group, Inc., Linus Torvalds, Microsoft Corporation, The Open Group, or the United Kingdom Office of Government Commerce in the United States, other countries, or both.

CITED REFERENCES

1. J. van Bon, *IT Service Management: An Introduction*, Van Haren Publishing, Zaltbommel, Netherlands (2005).
2. U. K. Office of Government Commerce, *Service Support, ITIL Managing Services,* Stationery Office, London, United Kingdom (2005), http://www.tsoshop.co.uk/bookstore.asp?FO=1159966&Action=Book&ProductID=0113300158.
3. *ISO/IEC 20000–1:2005, Information Technology—Service Management Specification*, International Organization for Standardization and the International Electrotechnical Commission (2005).
4. eServicesCapability Model (eSCM), IT Services Qualification Center, Carnegie Mellon University, http://itsqc.cmu.edu/.
5. J. E. King, Jr. and J. R. Nilsen, *System for Ordering Items Using an Electronic Catalogue*, U.S. Patent No. 5,319,542, (June 7, 1994).
6. IBM Tivoli IT Service Management, IBM Corporation, http://www-306.ibm.com/software/info/middleware/management/index.jsp.
7. H. Ludwig, J. Hogan, R. Jaluka, D. Lowenstern, S. Kumaran, A. Gilbert, A. Roy, T. R. Nellutla, and M. Surendra, "Catalog-Based Service Request Management," *IBM Systems Journal* **46**, No. 3, 531–548 (2007, this issue).
8. M. Ernest and J. M. Nisavic, "Adding Value to the IT Organization with the Component Business Model," *IBM Systems Journal* **46**, No. 3, 387–403 (2007, this issue).
9. HP IT Service Management (ITSM)—Enhancing IT Service Management for Business Success, Hewlett-Packard Development Company, L.P., http://h20219.www2.hp.com/services/cache/78734-0-0–225–121.html.
10. M. Galic, J. Adams, J. A. Bell, R. Disney, V.-M. Kanerva, S. Matulevich, K. Rebman, and P. Spaas, *Patterns: Applying Pattern Approaches, Patterns for e-Business Series,* IBM Redbook, IBM Corporation (July 28, 2003), http://www.redbooks.ibm.com/abstracts/SG246805.html?Open.
11. M. W. Johnson, A. Hately, B. A. Miller, and R. Orr, "Evolving Standards for IT Service Management," *IBM Systems Journal* **46**, No. 3, 583–597 (2007, this issue).
12. IBM IT Operational Management Products, IBM Corporation, http://www-306.ibm.com/software/tivoli/solutions/it-operational-management/.
13. IBM Tivoli Unified Process, IBM Service Management, http://www.ibm.com/software/tivoli/governance/servicemanagement/itup/tool.html.
14. P. P.-S. Chen, "The Entity-Relationship Model—Toward a Unified View of Data," *ACM Transactions on Database Systems* **1**, No. 1, 9–36 (1976).
15. S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuiness, P. F. Patel-Schneider, and L. A. Stein, *OWL Web Ontology Language Reference*, M. Dean and G. Schreiber, Editors, W3C Recommendation (February 10, 2004), http://www.w3.org/TR/owl-ref/.
16. D. Fensel, I. Horrocks, F. van Harmelen, S. Decker, M. Erdmann, and M. C. A. Klein, "OIL in a Nutshell," *Proceedings of the 12th European Workshop on Knowledge Acquisition, Modeling and Management*, Juan-les-Pins, France (2000), pp. 1–16.
17. The DARPA Agent Markup Language (DAML), Defense Advanced Research Projects Agency, http://www.daml.org/.
18. D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. McDermott, S. McIlraith, S. Narayanan, et al., *OWL-S: Semantic Markup for Web Services*, W3C Member Submission (November 22, 2004), http://www.w3.org/Submission/OWL-S/.
19. OMG Unified Modeling Language Specification, Version 1.4, Object Management Group (2001), http://www.omg.org/docs/formal/01-09-67.pdf.
20. M. Fowler, "What is the Point of the UML?" in "*UML*" *2003—The Unified Modeling Language—Modeling Languages and Applications*, P. Stevens, J. Whittle, and G. Booch, Editors, Springer, Heidelberg, Germany (2003), p. 325.
21. *Model Driven Architecture (MDA)*, J. Miller and J. Mukerji, Editors, Technical Report, Object Management Group (OMG), Architecture Board ORMSC (2001), http://www.omg.org/docs/ormsc/01-04-01.pdf.
22. R. High, Jr., S. Kinder, and S. Graham, *IBM's SOA Foundation: An Architectural Introduction and Overview, V 1.0*, White Paper, developerWorks, IBM Corporation (December 8, 2005), http://www-128.ibm.com/developerworks/webservices/library/ws-soa-whitepaper/.
23. OASIS Reference Model for Service Oriented Architecture 1.0, C. M. MacKenzie, K. Laskey, F. McCabe, P. F. Brown, and R. Metz, Editors, Organization for the Advancement of Structured Information Standards (OASIS), Committee Specification 1 (August 2, 2006), http://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf.
24. R. Heckel, M. Lohmann, and S. Thöne, "Towards a UMP Profile for Service-Oriented Architectures," *Proceedings of the Workshop on Model Driven Architecture: Foundations and Applications*, Enschede, The Netherlands (2003), pp. 115–120.
25. CBM—Component Business Modeling, IBM Haifa Research Laboratory, http://www.haifa.ibm.com/projects/software/cbm/index.html.
26. IBM Tivoli Application Dependency Discovery Manager, IBM Corporation, http://www-306.ibm.com/software/tivoli/products/taddm.
27. IBM Tivoli Change and Configuration Management Database (CCMDB), IBM Corporation, http://www-306.ibm.com/software/tivoli/products/ccmdb/.
28. Common Information Model (CIM) Standards, Distributed Management Task Force, Inc., http://www.dmtf.org/standards/cim/.
29. C. Draper, "Combine Autonomic Computing and SOA to Improve IT Management," developerWorks, IBM Corporation, http://www-128.ibm.com/developerworks/webservices/library/ac-mgmtsoa/index.html.
30. C. Mayerl, T. Vogel, and S. Abeck, "SOA-Based Integration of IT Service Management Applications," *Proceed-*

ings of the IEEE International Conference on Web Services, Orlando, FL (2005), pp. 785–786.

31. A. G. Ganek and T. A. Corbi, "The Dawning of the Autonomic Computing Era," *IBM Systems Journal* **42**, No. 1, pp. 5–18 (2003).

Accepted for publication February 12, 2007.
Published online July 13, 2007.

John Black
IBM Global Services Integrated Operations, IT Delivery, No 1 The Square, Bristol, BS1 6DG, United Kingdom (john.black@uk.ibm.com). Dr. Black is an architect in the Integrated Operations, IT Delivery organization. He was principal of the IBM United Kingdom Object Technology Practice and subsequently an e-business services principal consultant in the IBM Banking Sector. He moved into IT Service Delivery in 2003 as Design Authority Leader and Chief Architect for IBM Service Delivery in Europe. He is currently Chief Architect for the Global Solutions Architecture Repository within the IBM Global IT Delivery organization. Dr. Black is an Open Group Master Certified Architect and a Fellow of the Institution of Engineering and Technology.

Christine Draper
IBM Software Group, 11501 Burnet Road, Austin, Texas 78727 (cdraper@us.ibm.com). Mrs. Draper is a Senior Technical Staff Member on the IBM autonomic-computing architecture team and works with development teams within IBM to apply autonomic computing concepts to ITSM. An active member of the Organization for the Advancement of Structured Information Standards (OASIS), she is working to standardize a solution deployment descriptor. She has a B.A. degree with honors from the University of Cambridge, England.

Tom Lococo
IBM Global Services, 11 Kalina Drive, Rhinebeck, New York (tlococo@us.ibm.com). Mr. Lococo is a Senior Technical Staff Member in the Integrated Operations, IT Delivery organization. He works on IT outsourcing offerings and solutions, primarily in the Financial Services sector. His current focus is on developing ITSM reference architectures for IBM global delivery centers.

Fouad Matar
IBM Global Services Integrated Operations, IT Delivery, 800 No. Magnolia Avenue, Orlando, Florida 32803 (matar@us.ibm.com). Mr. Matar is a Senior Technical Staff Member in the Integrated Operations, IT Delivery organization. His focus is on establishing and advancing strong linkages with the Global Technology Services service-product line organizations. He has an M.S. degree in electrical engineering from the University of Central Florida and is currently working on a Ph.D. degree in mathematics. Mr. Matar creates service solutions in the server and storage areas.

Christopher Ward
IBM Research Division, Thomas J. Watson Research Center, 19 Skyline Drive, Hawthorne, New York 10532 (cw1@us.ibm.com). Dr. Ward is a research staff member and manager in the Service Delivery department. He has a Ph.D. degree in computer science from the University of Florida. He joined IBM in 2000 and is most recently responsible for innovative functionality in the configuration management process for the IBM Tivoli ITSM-based CCMDB product. Dr. Ward has published over 50 papers addressing a variety of computer science problems, is author or coauthor of numerous patents, and is a Senior Member of the IEEE. ■