Part III — Technical Architecture

Chapter 3 — SOA Concept

Introduction

This chapter provides an overview of the Medicaid IT Architecture (MITA) service-oriented architecture (SOA). It first examines the current Medicaid Management Information System (MMIS) architecture and challenges. It then defines SOA, introduces the various components of an SOA, and describes how a State builds a service in a MITA environment. The final sections describe the benefits of using SOA and how a State can begin to prepare to transition to this new architecture.

This chapter answers the following questions:

- What are the challenges facing MMIS today?
- What is service-oriented architecture?
- What are services?
- How are services built?
- Why should future MMISs move to a service-oriented architecture?
- What are the challenges facing MMIS?
- How can States prepare for a service-oriented architecture within Medicaid?

Purpose

This chapter provides the reader with an introduction to the basic concepts related to SOA. These concepts provide the foundation necessary to understand the MITA Technical Architecture (TA) presented in later chapters. This chapter is not designed to be a detailed reference to SOA.

Scope

SOA is the core technical concept of the MITA TA. The concept does not constrain how services are implemented, only that business process and technical functions are exposed using standard defined interfaces.

What Are the Challenges Facing MMIS Today?

Every State's MMIS, as its core, has automation for processing claims and other business functions. This core functionality includes handling baseline recipient, provider, reference, and other data. Additional systems might reside in the environment, including decision support systems (DSSs), financial systems, third-party recovery (TPR), and other applications. Their interfaces are often referred to as *tightly coupled,* which means that all system elements (e.g.,

program, database, subsystem, etc.) are highly dependent on each other and are generally interconnected through individual, point-to-point interfaces, as represented in **Figure 3-1**.

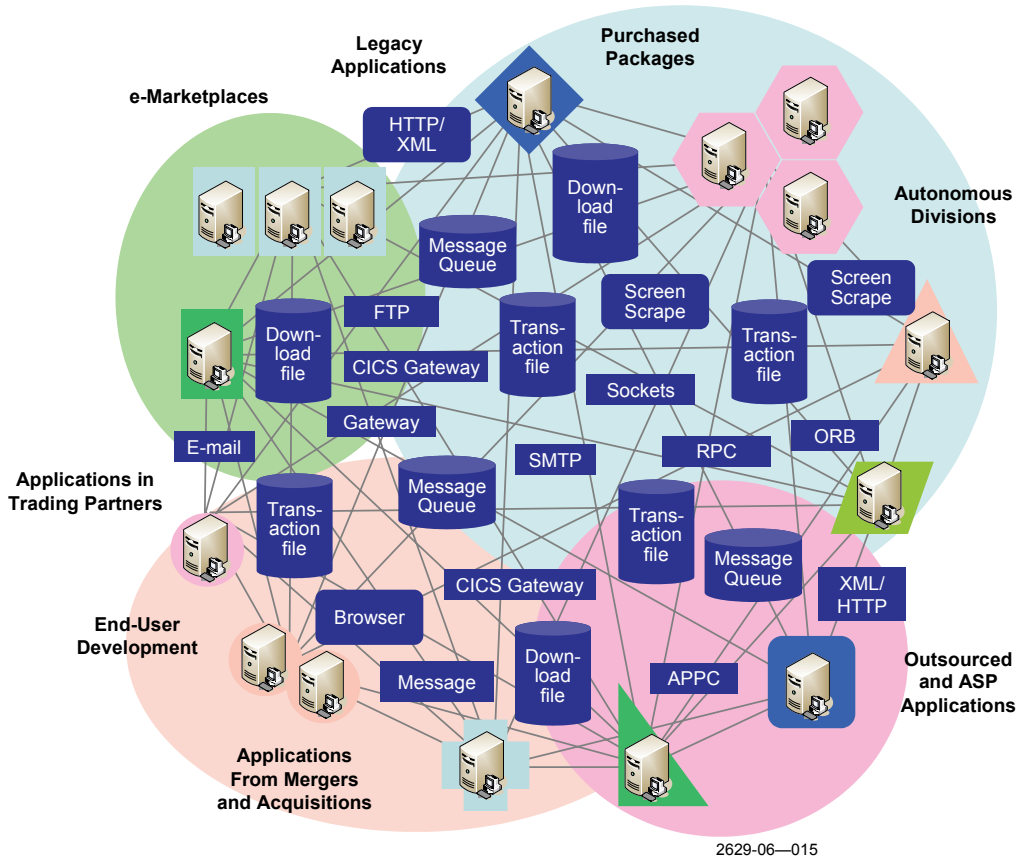

Figure 3-1. The Typical State of Current IT Systems

When business rules, policies, or legislation change, any and all parts of the system may be affected, including applications, databases, and interfaces. Thus, it becomes increasingly difficult to manage this complex environment. It is also becoming exceedingly costly to maintain these systems, both from the financial perspective (i.e., State and Federal) and from the intellectual property perspective (i.e., as the systems get older, fewer and fewer resources are familiar with the systems and know how to best maintain or change them). Additionally, different systems run in their own platform-specific environment. Each has its own components that do not communicate across functional or technical boundaries, so each new system upgrade creates a unique implementation environment. Therefore, the leverage for transporting certified systems from one State to another has been lost. There are other challenges as well. For example, end users may have difficulty identifying which subsystem performs what function, or they might be required to sign on to multiple systems in order to perform a single task (e.g., verifying eligibility and enrollment in several programs).

In addition to these challenges, difficulties are encountered whenever new changes are requested or required. A nonmandated change may take so long to implement that the end user forgets why the change was initially needed. In the final analysis, the States, Federal government, vendor community, and entire industry are held hostage by obsolete and expensive software applications and out-of-date technology. Although these solutions worked earlier, the prior methods of software integration are not sufficient for future use; these methods do not provide an adequate foundation on which to build the rapidly changing Medicaid enterprise of tomorrow.

To summarize, the three key challenges facing MMIS are:

- Highly interconnected systems using point-to-point interfaces require pervasive modifications to accommodate changes to business requirements, making them difficult to change.
- Users must navigate through multiple functional systems to perform a single task.
- MMIS are, to a large extent, platform dependent and do not communicate easily across functional or technical boundaries. This causes difficulty in sharing information or reusing functionality.

What Is Service-Oriented Architecture?

SOA is an application architecture within which business functions and selected technical functions can be employed, using documented interfaces. The World Wide Web Consortium (W3C) refers to SOA as a "set of components which can be invoked, and whose interface descriptions can be published (made available) and discovered (found)." SOA is an architectural framework that can incorporate and integrate many different technologies. The focus is on defining cleanly cut service contracts with clearly defined functionality in a manner that is transparent to the underlying technology platforms that provide the functionality.

Description of Service-Oriented Architecture

Figure 3-2 illustrates a simplified view of an SOA with examples of MITA services. Services can perform either business or technical functions. Services such as Process Claim, Enroll Provider, and Verify Provider Credentials perform *business* processes. Services such as Portal Service, Forms Management, and External Data Interchange Hub Services (i.e., Electronic Data Interchange [EDI] Gateway) perform high-level *Technical Service*s that are shared by many Business Services.

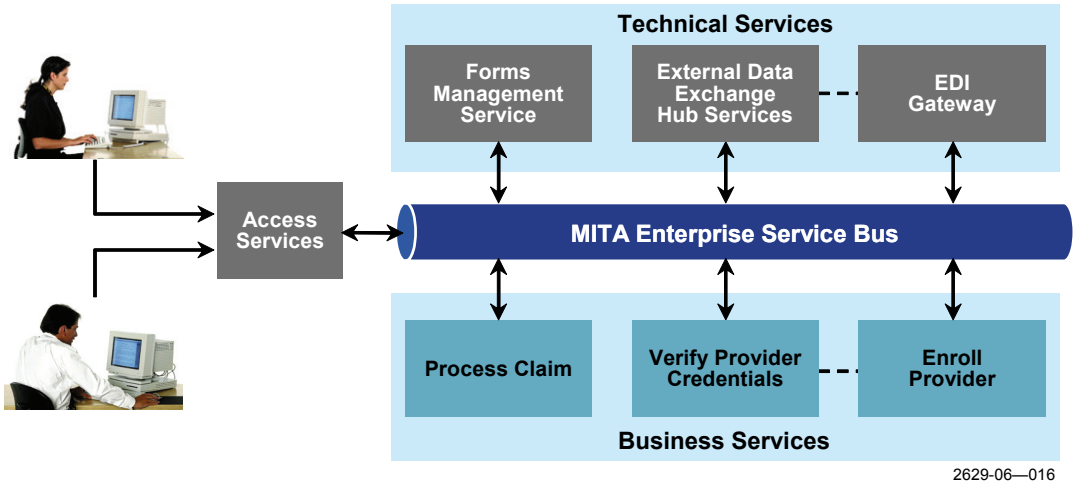

Figure 3-2. The MITA Service-Oriented Architecture

Simple or complex sets of services are interconnected by the Enterprise Service Bus (ESB). The ESB is a service layer that provides the capability for services to interoperate and be invoked as a chain of simple services that perform a more complex end-to-end process. The service layer is designed to handle normal conditions, respond to failures, and adapt to changes. The ESB provides the following functions:

- **Message Management.** This consists of reliable delivery of messages between services and built-in recovery.
- **Data Management.** This involves converting all messages between services to a common format and, in turn, converting messages from the common format to the application-specific format within a service. To ensure interoperability, the MITA message format is based on XML standards. Information sharing and alert/event notification standards are also defined for allowing information to be aggregated and integrated.
- **Service Coordination.** This consists of orchestrating the execution of an end-to-end business process through all needed services on the ESB. Services can adapt to changes in environments and are supported by a standards-based set of service management capabilities.

There are many different vendor implementations of an ESB, and the functions included in an ESB vary from one vendor to another. The list above identifies key functions needed for realizing an SOA, and for the purpose of this document, these functions are included in the definition of an ESB.

What Are Services?

In the Medicaid enterprise, the term *services* often describes the tasks that are performed on behalf of a beneficiary participating in the Medicaid (or other covered) program. For example, physicians provide medical services to beneficiaries. However, in SOA parlance, the term *service* is used to describe software services. Software services support the business needs, called *Business Services,* or provide commonly used technical capabilities, called *Technical Services*. Each software service performs a defined function that is documented in a service contract.

Figure 3-3 illustrates a simplified example of Business Services (i.e., services that produce a business result). In this example, the Business Service Enroll Provider receives as input an enrollment application for a provider, and it returns outputs such as More Information Needed, Application Accepted, or Application Rejected. Enroll Provider, in turn, invokes another Business Service Verify Provider Credentials in making its determination about a provider enrollment application. (**NOTE:** This example is not intended to be a comprehensive list of outputs.)

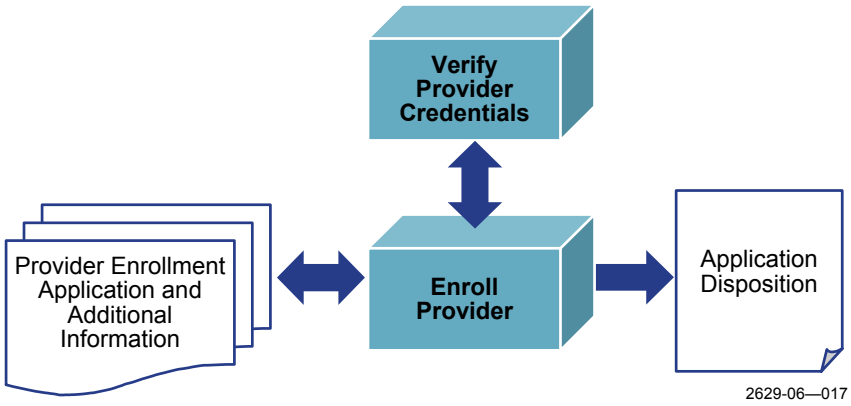

Figure 3-3. An Example of Business Services

Figure 3-4 illustrates possible Technical Services. Technical services do not produce business results by themselves, but they help Business Services produce business results. In this example, Figure 3-4 shows a Forms Management Service that allows forms to be filled online via Portal Services and made available electronically to Business Services such as Enroll Provider for processing.

Figure 3-4 illustrates another concept: The orchestration of a number of Business and Technical Services into an end-to-end process that delivers a business result. In this example, the business result is to determine whether a provider should be enrolled to provide services to Medicaid beneficiaries based on information supplied by the provider and from other sources available to Medicaid.

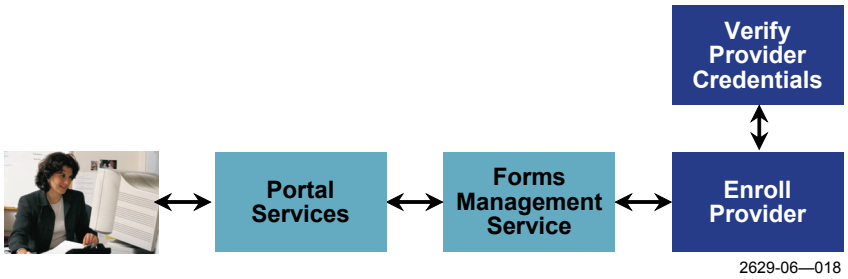

Figure 3-4. Examples of Technical Services and Their Interaction with Business Services

A service interacts with other services using messages. Thus, in their purest sense, services are software building blocks with natural boundaries that allow the organization of business capabilities. In order to be a service, a software component must be capable of performing one or more tasks or a defined piece of work. As a fundamental building block, a service demonstrates the following characteristics:

- Combines information and behavior
- Hides the internal workings from outside intrusion (i.e., operates as a "black box" with defined functionality)
- Presents a formally defined interface or interfaces

What Is a Message?

A *message* is a structured set of data that follows a prescribed method of delivery. The delivery method is called a *message protocol*. The message protocol results in a completely self-contained information exchange. In the examples shown in Figures 3-3 and 3-4, messages are the data requests and responses that flow between services and are used to invoke services (or, from another perspective, if services are the bricks, messaging is the mortar holding the bricks together). A message may be a request for a service, a series of requests, or a response. The input message for one service creates an output message that is the input to the next service.

MITA is standardizing the use of XML-based message interchange among Business Services and across organizational boundaries. XML messages are *self-documenting,* which means that each field in the message has a tag that defines the field (e.g., a field with the tag "Last_Name" contains a person's last name). Consumers of a message look for and use fields that they need for their processing, and they can ignore fields they do not need. Therefore, if a new field is added (e.g., "Middle_Initial"), the consuming service does not need to be modified. The service contracts and the XML-based messages remains consistent, while the underlying application or technical capabilities may change. This approach minimizes the impact of changes on Medicaid IT systems.

Platform Independence

A key feature of an SOA is that it can be implemented independent of the underlying platform. Each service is invoked in a standard way using one or more messages, and each message results in the invocation of one of the documented functions supported by the service, regardless of implementation details, as shown in **Figure 3-5**.

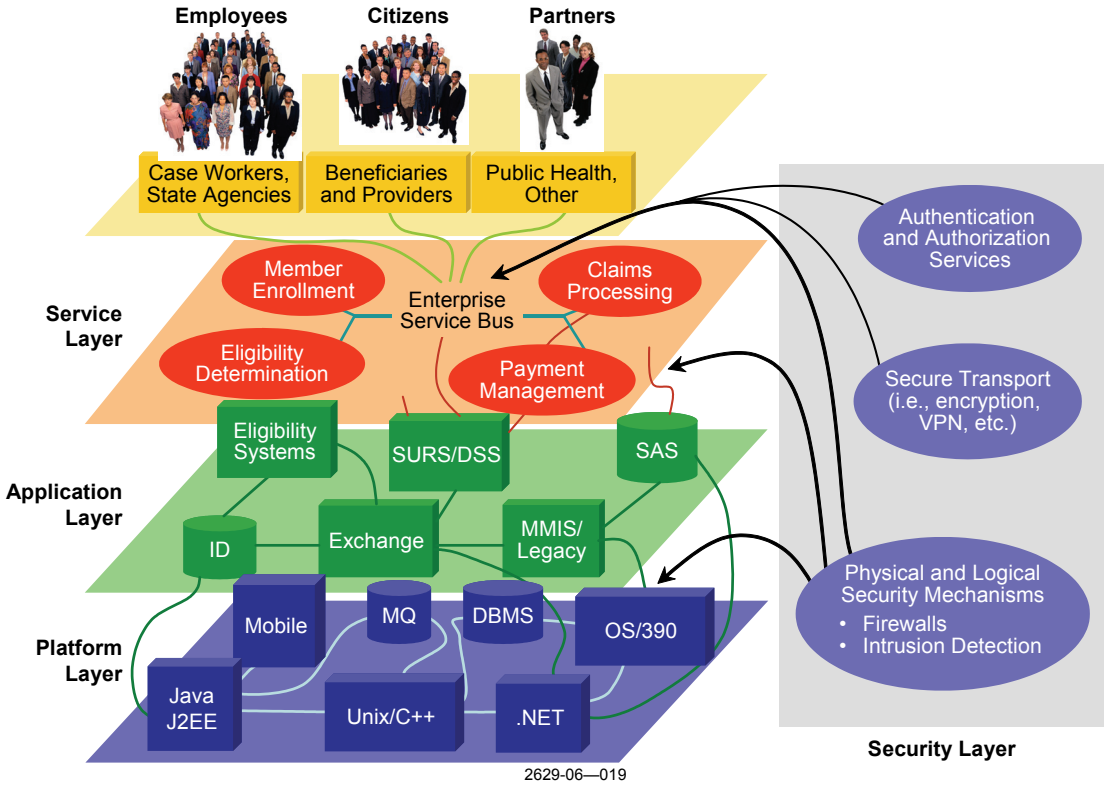

Figure 3-5. SOA Layers Provide Platform Independence

The top layer of Figure 3-5 is the Service Layer that is focused around the delivery of services to stakeholders using a variety of service access points (i.e., interfaces). The service interface to the case workers will be different from those to the citizens or provider community. The many special interfaces and preferences on content are handled within the connections to the Service Layer.

The Service Layer provides a uniform and open standards-based way for end-users to interact with a system built on SOA principles and for high-level Business and Technical Services to interact with each other. Each service is implemented by applications at the Application Layer, but the Service Layer hides the nature of the application from the service consumers. This means that a service at the Service Layer may be built using any combination of commercial off-the-shelf (COTS) packages, legacy systems, or custom code, but application-specific details are not evident to the service consumers. The applications, in turn, could be implemented on one or

more platforms — consisting of different operating systems (e.g., Unix, Windows, or OS/390), database management systems and data service interfaces (e.g., Oracle or DB2), message-oriented middleware (e.g., IBM WebsphereMQ or Microsoft's Message Queuing Server [MSMQ]), or an SOA platform (e.g., Iona Artix, Sonic ESB, or BEA Weblogic) — and these details, again, would be transparent to service consumers.

Existing systems can be wrapped and invoked as service-provider systems. The linking between service consumers and service providers can be established at run time via a service registry. This "loose coupling" means that an individual service can be replaced with a new implementation without impacting the rest of the enterprise. For example, it would be possible to replace a service that is currently a wrapped COBOL application with a COTS product or a J2EE, C#, C++ program, without changing any of the external interfaces.

From a purely technical perspective, SOA is about architecture, form, and structure. However, it is more encompassing than architecture. It also addresses issues such as business process design and service delivery processes. Properly implemented, SOA can maximize business capabilities by providing the flexibility to use services how and when they are needed, instead of within a rigid, prescribed format. Services will interact only through a well-defined interface, and this interface must be made known (i.e., exposed) so that other services know how to interact with it. Services are assembled based on the specific needs of the business. The specific services are identified, along with the order in which the services will be used (i.e., the sequence), the number of times a service will be used (i.e., the occurrence), and how many times the entire sequence will be executed (i.e., the iterations). Services have standards for quality of service, transaction management, security, and privacy. The technical and business concerns that were previously solved uniquely by each vendor are now being solved with vendor-independent standards.

SOA can manage the use of these services (e.g., delivery, acquisition, consumption, etc.) in terms of, and in sets of, related services. This capability will have big implications on how software life cycles are managed in the future (e.g., in terms of the specification of requirements as services, design of services, acquisition and/or outsourcing of services, asset management of services, etc.). This represents a new way of thinking in the software industry in general, and in Medicaid/MMIS in particular.

How Are Services Built?

To illustrate how services are built using SOA, **Figure 3-6** shows some of the key pieces of a Business Service. Each service has an Interface Services Layer that includes security services which inspect incoming messages to verify that the message originator is authorized to invoke the service. For example, only designated Medicaid staff members are authorized to approve claims.

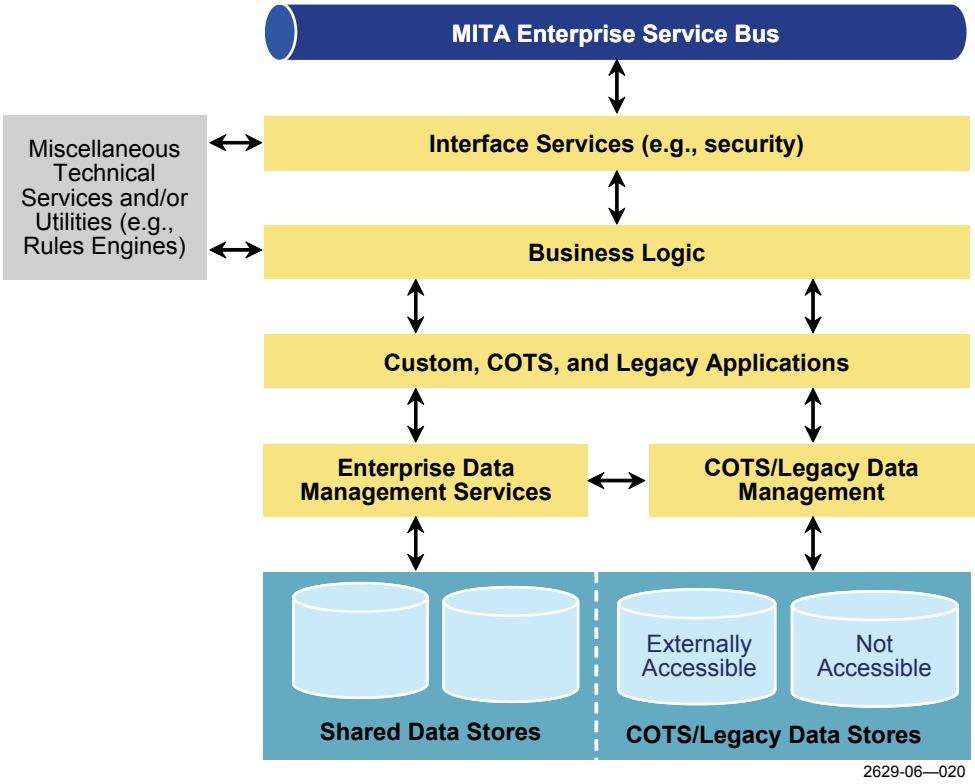

Figure 3-6. The General Structure of Business Services

The Business Logic Layer examines the received message and coordinates the execution of underlying custom, COTS, or legacy applications to provide the needed business results for the received message. The Business Logic Layer accesses miscellaneous Technical Services to perform its processing.

The Applications Layer accesses and uses a set of enterprise data management services to provide uniform access to data by using standard definitions for all shared and externally accessible data in the State Medicaid systems. Data contained in COTS and legacy systems require special handling. In the case of legacy data stores, data access routines tend to be specific to the underlying technology; data stores used by COTS packages often have proprietary data structures and data access routines and can be accessed only by vendor-provided application programming interfaces (APIs). In addition, not all COTS data stores are externally accessible, even via APIs. The enterprise data management services provide transparent data management services to all accessible/shared data stores in State Medicaid systems. The data access services are platform independent, so that as legacy systems are phased out, or databases restructured, no changes to the enterprise data management services are visible to the user (or the consumer).

Using SOA, services can be invoked at different layers. To maximize reuse across the Medicaid enterprise, MITA will standardize the service contract (i.e., service descriptions), the invoking

messages for all of the services that are connected to the ESB, and as many of the lower level Technical Services as possible.

What Is a Web Service?

A Web service is a reusable software service that interacts with other software components by exchanging messages *using Web service standards*, including XML; simple object access protocol (SOAP); universal description, discovery and integration (UDDI); and other industry-recognized standards. Web service standards enable services to be discovered and invoked in a location-independent manner across a network (i.e., via a State Medicaid's intranet or the Internet). While a service can be a Web service, not all services are Web services. For instance, some MITA services will be Web services because they require or benefit from this flexibility. Other MITA services may be implemented more economically as non-Web services. In architecting the MITA SOA, the MITA team will consider each service's usage characteristics in determining the best approach to structuring and invoking the service.

Why Should Future MMISs Move to a Service-Oriented Architecture?

The following sections present a compelling value proposition (i.e., the business case) for using an SOA as the foundation for MITA.

Enables Increased Business Agility

The basic doctrine under which every Medicaid enterprise operates remains the same year after year: Business change and uncertainty are the rule, including changes in policies, procedures, laws, and regulations. It is the way in which a Medicaid enterprise addresses change and uncertainty that sets it apart from other public service programs. Introducing new processes and technologies will help in creating new tools and techniques to address these changes. MITA must be "designed for change" and must include a built-in change deployment process. Business innovation enabled by a service-oriented approach will allow changes to be addressed with business-oriented tools that do not require arcane programming skills.

An SOA enables change in two dimensions: how processes are formed and how services are changed. Both dimensions provide mechanisms for promoting business agility.

With an SOA, responses to business changes will not be restricted or slowed down by technology. Rather, the SOA approach will be the foundation for proactive, rapid response to change.

Business Drives the Enterprise, Not Technology

With an SOA, business needs — not technology — will drive the enterprise. The intent of SOA is to enable the user to think in a business-centric way without being concerned with the IT implications. Conversely, it enables the introduction of IT without "upsetting the applecart" of the enterprise business.

Facilitates Greater Reuse

By following the SOA standards established by a large enterprise, constituent organizations can develop Business and Technical Services that will be reusable throughout the enterprise. Reuse typically has three benefits:

- Lower costs
- Reduced development schedules
- Lower implementation risk

By fostering collaborative development of services, the MITA SOA promises to significantly reduce time to market for new MMIS capabilities, while lowering the cost and risk of their implementation.

Facilitates Insertion of New Technology

The layers in an SOA shown in Figure 3-5 create an environment where platform- or technology-specific characteristics are hidden from the top-level Business and Technical Services. As a result, the impact of inserting new technology is localized to the layer at which the technology is used. New technologies can thus be inserted in an SOA in a manner that is transparent to the consumers of the Business and Technical Services, minimizing the types of disruptions typical in organizations that do not use SOA principles.

Increases Operational Flexibility

SOA provides the necessary infrastructure (i.e., plumbing) to create an operational environment where services can be dynamically replicated or moved from one system node to another (e.g., between servers) without breaking the application.

This feature is desirable for environments that require highly proactive management of the system operations. For example, an organization with systems that have high uptime/reliability and throughput requirements would find this feature particularly important, especially if these requirements are incorporated into a Service Level Agreement (SLA). The operational flexibility inherent in an SOA can be a powerful enabler for the movement of a State's MMIS to an SLA-based environment, thereby enhancing the quality of service provided to the stakeholder.

What Are the Challenges Facing MMIS?

SOA enables the Medicaid enterprise to meet the three key challenges facing MMIS.

Challenge 1: Highly interconnected systems using point-to-point interfaces require pervasive modifications to accommodate changes to business requirements, making them difficult to change. SOA is based on standardized messages being passed among services. MITA will leverage industry-standard message formats and create its own message formats with XML for special Medicaid needs. The plethora of individual point-to-point interfaces will be replaced by a set of standardized message-driven interfaces for each service. All interface

changes will be localized to a single set of interfaces. Furthermore, the use of XML-based messages will minimize the need for changes to consuming services because only those services that depend on the changes to the messages will need to be changed. Finally, if an internal change is made to a service that does not change the service functionality or its interface, no other services will need to be changed. The end result is that changes to systems built using SOA principles will tend to be less pervasive. Hence, such systems can be changed more easily and more rapidly.

Challenge 2: Users must navigate multiple functional systems to perform a single task. SOA uses a combination of Business and Technical Services to implement end-to-end processes. Services can be linked via messages, and the user can be automatically navigated through the services needed to complete a task. This linking of services is performed using an orchestration language. This presents users with a consistent user interface with single sign-on, making MMIS more automated and easy to use.

Challenge 3: MMISs are, to a large extent, platform dependent and do not communicate easily across functional or technical boundaries. Business processes/business functions are hard to reuse. With SOA, business functions/processes can be invoked as services with standard, message-driven interfaces. If all MMISs follow these rules, services can be invoked or reused in a platform-independent manner by any service anywhere in the enterprise.

How Can States Prepare for Service-Oriented Architecture Within Medicaid?

Successful adoption of an SOA paradigm in the Medicaid enterprise will require States to make the following three key changes:

- Training personnel for new approaches and roles
- Adjusting skill sets
- Implementing collaborative governance for SOA implementation

Training Personnel for New Approaches and Roles

IT managers and their staff are on the same team as the rest of the Medicaid organization. In the past, IT supported the business team — not quite at arm's length, but almost. Today, however, IT *belongs* to the team. IT managers and their staff need to become much more aware of business aspects of the enterprise (e.g., what it does and how it does it) in order to help other team members to better understand what IT can do and how to use IT fully. This is where services come in. The people who design the services (i.e., IT managers and their staff) must understand which services are necessary and how they will be used. This information can only come from the business user. The business user, on the other hand, must learn what IT can do to help them recognize their maturity levels related to successfully completing their business processes. These levels are defined as *capabilities*.

When these capabilities are integrated with business processes, business rules, knowledgebases, and other enterprise applications, an organization gains confidence in its ability to manage its IT assets and workforce in the most adaptive and cost-effective way possible. The ability to gain rapid and real-time access to (and deployment of) the best information, regardless of its location, is how these levels will be measured.

All of this leads to using IT to prepare the Medicaid enterprise for its rapidly changing future. Transforming the Medicaid enterprise into a sleek, agile organization that can withstand high levels of change without jeopardizing its existence depends on the synergy and flexibility of its people, processes, and technology. The interdependencies that result from an agreeable and harmonious relationship among these dynamic entities reinforce (and are simultaneously reinforced by) the government's resilience and provide the Medicaid enterprise with the strength needed to survive.

Adjusting Skill Sets

MITA will assist the Medicaid enterprise to establish a Framework for the planned use of technology and infrastructure to meet its changing business needs. Moving toward SOA will assist each organization to assess the most appropriate mix of skills, processes, and knowledge to support the continued development and enrichment of the core competencies needed for this level of flexibility. In addition to the specific technical skills required to run the IT operation, this obligation necessitates a mix of IT and business-related skills across the entire organization and at all levels.

To put it plainly, Medicaid and similar organizations must be able to align their available talent, processes, and knowledge with changing needs, almost on the fly, with limited financing. An understanding of why the organization is in business and what tools can be used to ensure success establishes the solid foundation for the development of capabilities. A strategy that is developed on this foundation of knowledge is difficult to undermine, and it benefits from the combination of business and technology foresight needed to provide a consistent, logical view of the enterprise, its context, and its environment.

Implementing Collaborative Governance for SOA Implementation

Although reuse of services is one of the contributors to the SOA value proposition, such reuse does not happen in a vacuum. Potential users of services need to define services collaboratively so that the resulting services are flexible enough to meet the requirements of as many users as possible. Achieving this collaborative environment across State Medicaid organizations will require new approaches and tools for governance, and MITA will provide the guidelines and tools for enabling this collaborative paradigm. States, in turn, will need to actively promote the new paradigm in their organizations to ensure that that the full value of SOA is leveraged to benefit the entire Medicaid enterprise.

Conclusion

This chapter has provided an introduction to the SOA concept, and readers should now be able to:

- Understand the concept of SOA
- Understand why MITA has decided to use SOA
- Understand what needs to be done next
- Read and understand the detail chapters of the MITA TA