MITA Information Series

1. What is MITA? An Overview
2. MITA and APDs
3. Planning for MITA — An Introduction to Transition Planning
4. What is a MITA Hub?
5. **Service-Oriented Architecture — A Primer**
6. The MITA Repository
7. The MITA Maturity Model
8. The MITA Operations Concept
9. The MITA Business Process Model
10. Comparing MITA and MHCCM

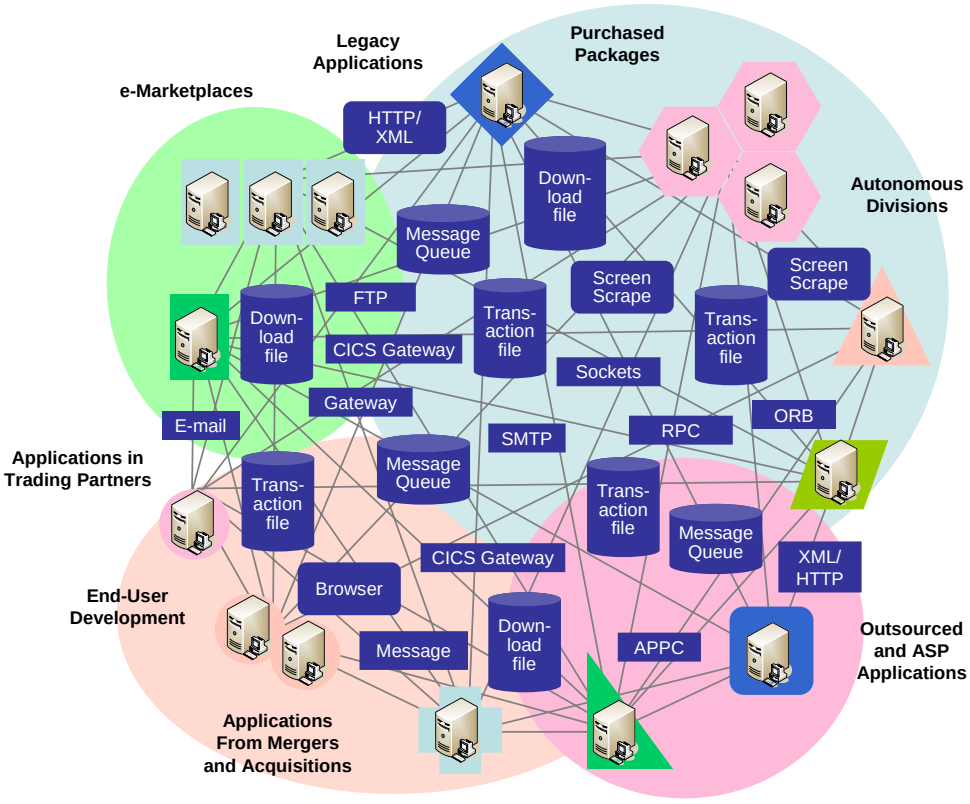
CMS
CENTERS for MEDICARE & MEDICAID SERVICES

Service-Oriented Architecture and the MMIS

Challenges Facing MMIS Today

Every state's Medicaid Management Information System (MMIS) has, as its core, automation for processing claims and other business functions. This core functionality includes handling baseline recipient, provider, reference, and sometimes, 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 what we call *tightly coupled* — each program, database, subsystem is highly dependent on another, and all are generally interconnected through individual, point-to-point interfaces, as represented in Figure 1.

Figure 1. Typical State of Current IT Systems



When business rules, policies, or legislation change, any and all parts of the system may be impacted, 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 a financial perspective (state and federal), as well as from an intellectual property perspective. As systems grow older, fewer resources are available that are both familiar with those systems and know best how to maintain or update them.

Additionally, different systems often run in their own platform-specific environment. Because system components do not communicate across functional or technical boundaries, a system upgrade creates a unique implementation environment. We have, as a result, lost the leverage for transporting certified systems from one state to another.

There are other challenges as well. For example, end users must know which subsystem performs which function. They might be required to sign on to multiple systems to perform a single task, such as verifying eligibility and enrollment in several programs. In addition, difficulties are encountered whenever new changes are requested or required. For example, a nonmandated change may take so long to implement that the end user has forgotten why the change was initially needed.

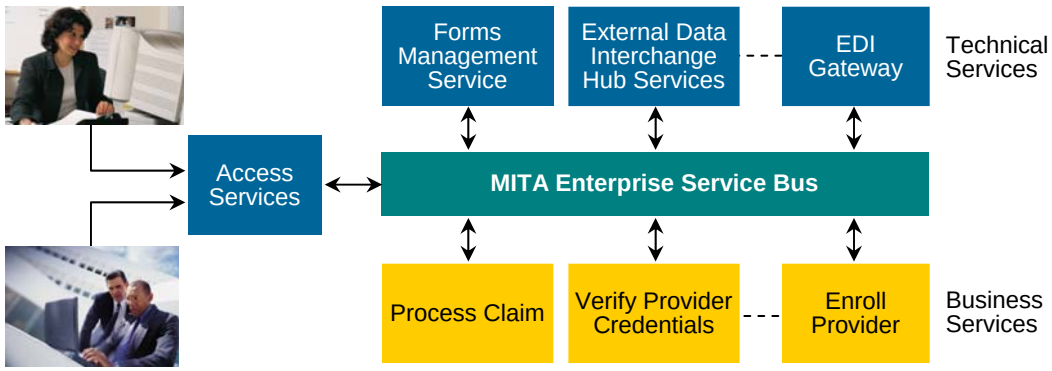
In the final analysis, the entire industry, states, federal government, and the vendor community are held hostage to obsolete and expensive software applications and the use of out-of-date technology. While they worked earlier, the prior methods of software integration are not sufficient for future use, as these methods do not provide an adequate foundation on which to build the rapidly changing Medicaid enterprise of tomorrow.

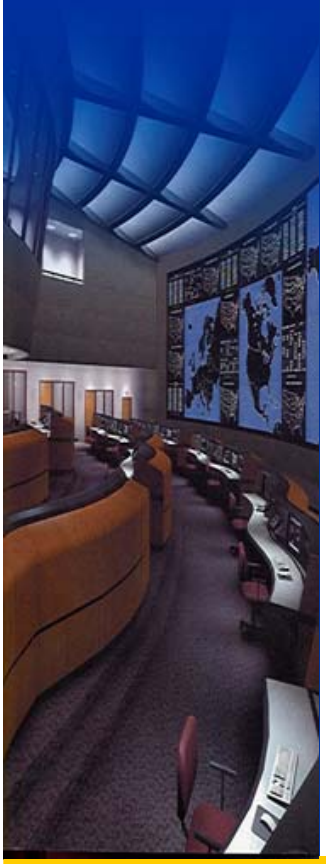
What Is a SOA?

This paper discusses an approach, known as *service-oriented* architecture that addresses these issues.

A service-oriented architecture (SOA) is an application architecture within which business functions and selected technical functions can be invoked 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.

Figure 2. MITA Service-Oriented Architecture





Description of SOA

Figure 2 shows a simplified view of a SOA with examples of Medicaid Information Technology Architecture (MITA) services. These 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, External Data Interchange Hub Services, or EDI Gateway, perform high-level **technical** services that are shared by many business services.

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

- **Message Management** is the reliable delivery of messages between services and built-in recovery.
- **Data Management** converts all messages between services to a common format, and in turn, converts messages from the common format to the application-specific format within a service. The MITA message format for interoperability is based on XML standards. Information sharing and event-notification standards are also defined for allowing information to be aggregated and integrated.
- **Service Coordination** orchestrates 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.

Many vendor implementations of an ESB exist, and the functions included in an ESB vary from one vendor to another. The list of functions mentioned previously are key functions needed for realizing an SOA, and for the purpose of simplification in this paper, we are including these functions in our definition of an ESB.

What Is a Service?

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. In SOA parlance, however, we use the term service to discuss 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, which is documented in a service contract.

Figure 3 shows a simplified example of business services; a business service is one that produces 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. (Note: This is not intended to be a comprehensive list of outputs.) Enroll Provider, in turn, invokes another business service, Verify Provider Credentials, in making its determination about a provider enrollment application.

Figure 3. Example of Business Services

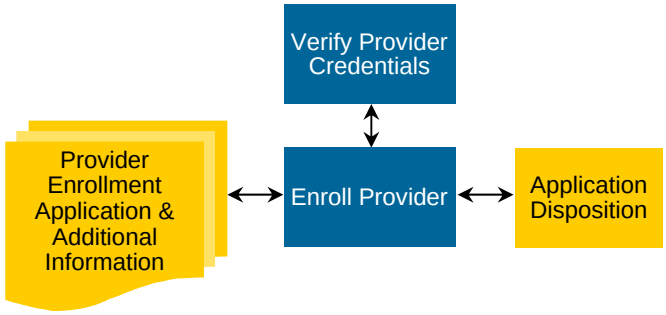


Figure 4 shows examples of technical services (shown in orange). Technical services do not produce business results by themselves, but help business services in producing business results. The example in Figure 4 shows a Forms Management service that allows forms to be completed online via Portal Services and made available electronically to business services, such as Enroll Provider, for processing.

Figure 4. Example of Technical Services and their Interaction with Business Services

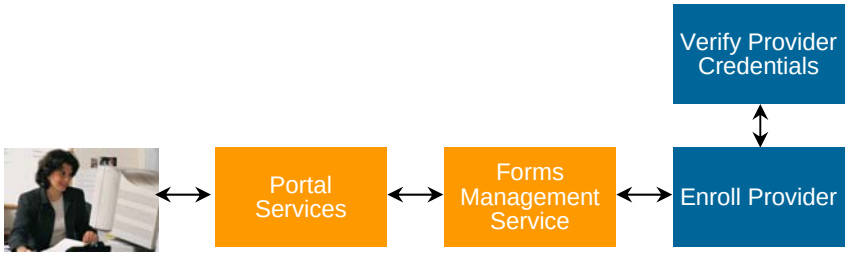


Figure 4 also illustrates another concept — the orchestration of a number of business and technical services into an end-to-end process that delivers a business result. In this example, the business result is to determine whether a provider should be enrolled to provide services to Medicaid beneficiaries on the basis of information supplied by the provider as well as information available from other sources available to Medicaid.

A service interacts with other services using messages. Services, in their purest sense then, are software building blocks with natural boundaries that allow us to organize business capabilities. 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 and 4, messages are the data requests and responses that flow between services and are used to invoke services. A message may be a request for a service, a series of requests, or a response. Where services are the bricks, messaging is part of the mortar between the bricks. 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-message interchange between business services and across organizational boundaries. XML messages are self-documenting; that is, each field in the message has a tag, which defines the field. For example, a field with the tag Last_Name contains a person’s last name. Consumers of the message look for (parse) and use fields that they need for their processing. If a new field called Middle_Initial is added, consumers that do not need to look at this field can ignore it, and therefore, the consuming service does not need to be modified. The service contracts and the XML-based messages are consistent, while the underlying application or technical capabilities may change. This minimizes the impact of changes on Medicaid IT systems.

Platform Independence

A key feature of a 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; each message results in the invocation of one of the documented functions supported by the service regardless of implementation details, as shown in Figure 5.

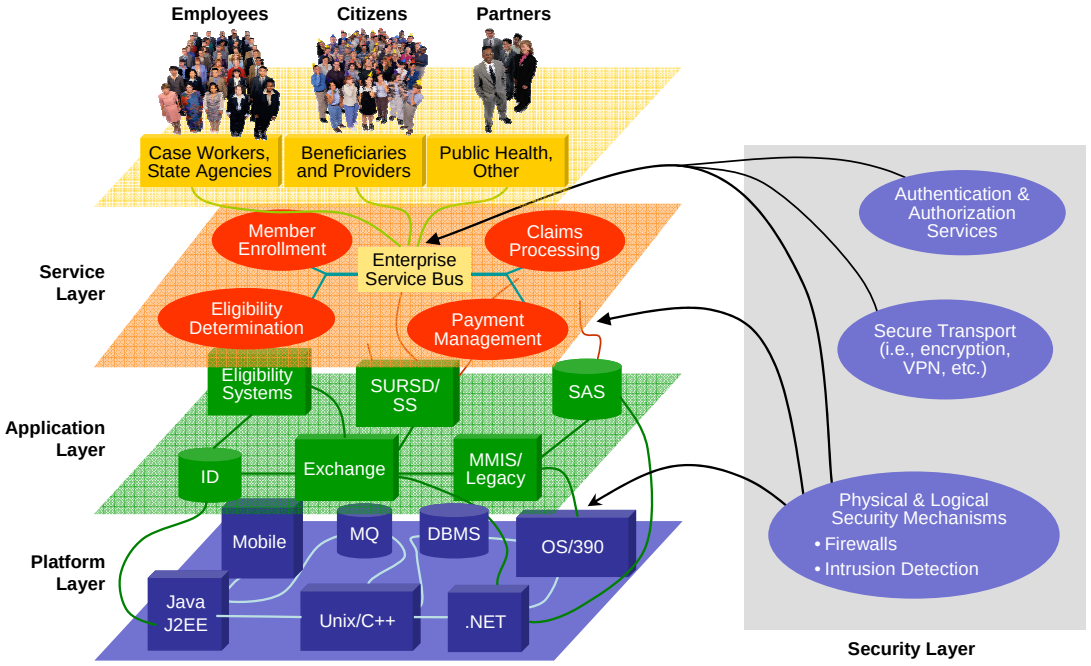


Figure 5. SOA Layers Provide Platform Independence

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

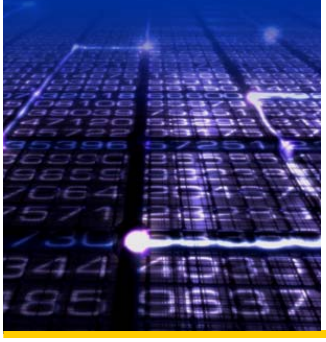
The Service Layer shown in Figure 5 provides a uniform and open standards-based way for end users to interact with a system built using SOA principles and for high-level business and technical services to interact with each other. Each service is implemented by applications at the Application Layer, but the Service Layer hides the nature of the application from the service consumers. By this, we mean that a service at the Service Layer could 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 (such as Unix, Windows, or OS/390), database management systems and data service interfaces (such as Oracle or DB2), message-oriented middleware (such as IBM WebsphereMQ, Microsoft's Message Queuing Server [MSMQ]), or an SOA platform (such as Iona Artix, Sonic ESB, or BEA Weblogic). These details, again, are transparent to the 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 runtime 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. As an example, it will 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.

Thus far, we have considered SOA from a purely technical perspective. SOA is about architecture, form, and structure, but it is also more encompassing than architecture. It also includes addressing issues such as business process design and service delivery processes. Properly implemented, SOA enables you to maximize your 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 (exposed) so that other services know how to interact with it. Services are assembled on basis of the specific needs of the business. The specific services are identified, as well as the order in which the services will be used (*sequence*), the number of times a service will be used (*occurrence*), and the number of times the entire sequence will be executed (*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 allows us to manage the use of these services (delivery, acquisition, consumption) in terms of, and in sets of, related services. This will have big implications for how we manage the software life cycle from now on — right from specification of requirements as services, design of services, acquisition and/or outsourcing of services, asset management of services, and so on. This is a new way of thinking in the software industry, and in Medicaid/MMIS in particular.

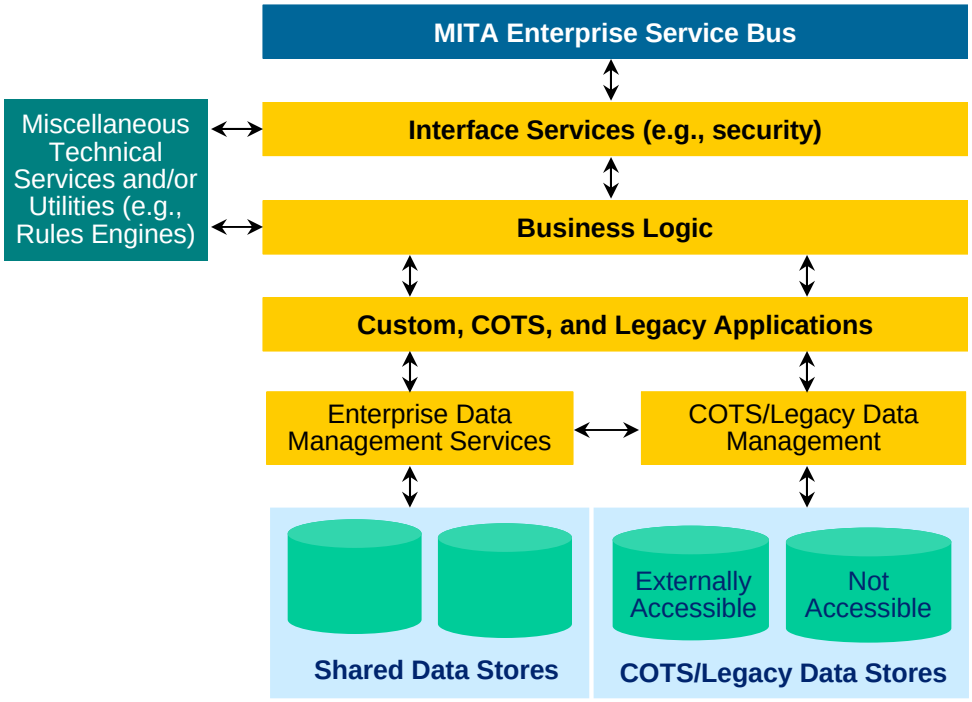
How Are Services Built?



Now that we understand some basic concepts of an SOA, we now describe how services are built — what's "under the hood". Figure 6 shows some of the key parts of a business service. Each service has an Interface Services Layer, which includes security services that look at incoming messages to verify that the message originator is authorized to invoke the service. As an example, only designated Medicaid staff members are authorized to approve claims.

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.

Figure 6. General Structure of Business Services



The Applications Layer accesses and, in turn, 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 requires 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 in many cases have proprietary data structures and data access routines, and can be accessed only by vendor-provided application programming interfaces (APIs). Besides, 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, as legacy systems are phased out or databases are restructured, there are no user/consumer-visible changes to the enterprise data management services.

So it is clear that services can be invoked at different layers. MITA will standardize (1) the service contract (service descriptions), (2) the invoking messages for all of the services connected to the ESB, and (3) as many of the lower level technical services as possible to maximize reuse across the Medicaid enterprise.

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 that include XML, SOAP, and other industry-recognized standards such as UDDI. While a service can be a Web service, and a Web service is a service, not all services are Web services. Use of Web services standards allows services to be discovered and invoked in a location-independent manner across a network; this could be either a state Medicaid's intranet or the Internet. Some MITA services will be Web services because they require or benefit from this flexibility. Other MITA services may be more economically implemented 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.

Should Future MMIS Move to SOA

Now that we have discussed what an SOA is, let's talk about the value proposition, or the business case, for using an SOA as the foundation for MITA.

Enables Increased Business Agility

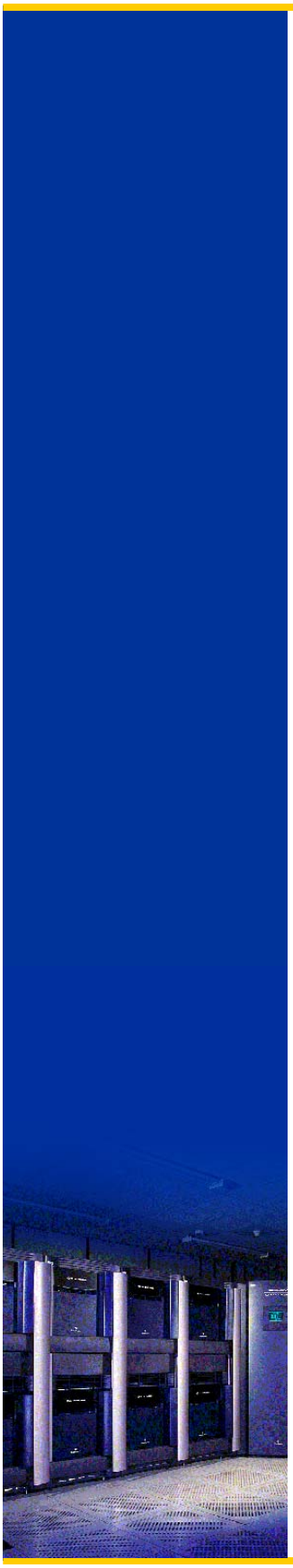
The basic doctrine under which every Medicaid enterprise operates remains the same year after year — business change and uncertainty are the rule, including changes in policies, procedures, laws, and regulations. Introducing new processes and technologies creates new tools and techniques that address the changes. It is the way in which a Medicaid enterprise addresses change and uncertainty that sets it apart from the crowd of other public-service programs. MITA must be "designed for change" and must include a built-in change-deployment process. Business innovation will be enabled by a service-oriented approach that allows new 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 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

In the Medicaid enterprise, many processes today involve the use of IT to some degree. The intent of SOA is to allow the business user to think in a business-centric way and not have to be concerned with the IT implications. Conversely, it allows IT to be introduced without upsetting the enterprise business "applecart". With an SOA, business needs will drive the enterprise, *not* technology.



Facilitates Greater Reuse

If the constituent organizations in a large enterprise follow the enterprise SOA standards set forth for the enterprise, business and technical services developed by one organization will be reusable by other organizations in the enterprise. Reuse typically has three benefits: lower cost, reduced development schedule, and lower implementation risk. By fostering collaborative development of services, the MITA SOA promises to reduce significantly the "time to market" for new MMIS capabilities, while lowering both the cost and risk of their implementation.

Facilitates Insertion of New Technology

The layers in an SOA shown in Figure 5 create an environment in which 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 a manner that is transparent to the consumers of the business and technical services in an SOA — minimizing the types of disruptions that are the rule rather than the exception in most organizations that do not use SOA principles.

Increases Operational Flexibility

An SOA provides the necessary infrastructure ("plumbing") to create an operational environment where services can be dynamically replicated or moved from one system node (such as a server) to others without breaking the application.

This flexibility is highly desirable for environments in which highly proactive management of the system operations is a necessity. This is particularly important in an organization where systems have requirements for high uptime/reliability and throughput, particularly 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 states MMIS to an SLA-based environment, thereby enhancing the quality of service provided to the stakeholder.

Let's now discuss how an SOA enables the Medicaid enterprise to meet the three key specific challenges facing MMIS systems described in Section 1.

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 between 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 are localized to a single set of interfaces. Furthermore, the use of XML-based messages minimizes the need for changes to consuming services. Only those services that depend on the changes to the messages need to be changed, which minimizes the number of services that require changes. 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 tend to be less pervasive; hence such systems can be changed more easily and more rapidly.

Challenge 2. Users must navigate through 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 together via messages and the user can be automatically navigated through the services needed to perform a complete task (such as the example described in Section 2, Enroll Provider). Users can be presented with a consistent user interface with single sign on to enable MMIS systems to become more automated and easy to use.

Challenge 3. MMIS, to a large extent, is platform dependent and does not communicate easily across functional or technical boundaries, which makes it difficult to share information or reuse functionality.

Business functions/processes can be invoked as services with standard, message-driven interfaces. If all MMIS systems play by these rules, services can be invoked or reused in a platform-independent manner by any service anywhere in the enterprise.

Getting Ready for SOA within Medicaid

Successful adoption of an SOA paradigm in the Medicaid enterprise will require states to make some important changes. Three key changes include:

- 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 belong to the same team that is the Medicaid organization. In the past, IT supported the business team — not quite at arm's length, but almost. Today, however, they *belong* to the team. These IT managers and their staff must become much more aware of what the business does and how it does it in order to participate in a meaningful way to help other team members develop a better understanding of what IT can do, and to plan to use it fully. This is where services come in. The people who design the services (IT) must understand which services are necessary, and how they are used. This information can only come from the business user. The business user must also learn what IT can do to help them recognize their maturity levels related to successfully completing their business processes. We define these levels as *capabilities*.

When these capabilities are integrated with business processes, business rules, knowledge bases, and other enterprise applications, an organization can have some confidence that it has the 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 to deploy) the best information, regardless of its location, is how these levels will be measured.

All of this leads to using technology to enable the Medicaid enterprise to prepare for its rapidly changing future. Changing it into a sleek, agile enterprise that can withstand the buffeting winds of change without jeopardizing the industry's very 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 it with the strength needed to survive.

Adjusting Skill Sets
MITA will assist the Medicaid enterprise in ensuring that a framework exists 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 to support 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.

An understanding of why the organization is in business and what tools it can use to make it succeed establishes the foundation for the development of capabilities that are very difficult to undermine. A strategy that is developed on this foundation of knowledge is strong, 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. To put it very 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.

Implementing Collaborative Governance for SOA Implementation

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