Prepared for the:
Administration for Children and Families (ACF)

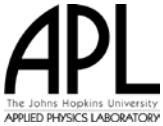
National Human Services Interoperable Architecture

Infrastructure Viewpoint

DRAFT Version D0.1

September 29, 2011

Prepared by:
The Johns Hopkins University
Applied Physics Laboratory (JHU/APL)



Draft Issue

It is important to note that this is a draft document. The document is incomplete and may contain sections that have not been completely reviewed internally. The material presented herein will undergo several iterations of review and comment before a baseline version is published.

This document is disseminated in the interest of information exchange. The Johns Hopkins University Applied Physics Laboratory (JHU/APL) assumes no liability for its contents or use thereof. This report does not constitute a standard, specification, or regulation. JHU/APL does not endorse products or manufacturers. Trade and manufacturer's names appear in this report only because they are considered essential to the object of this document.

Note: This document and other NHSIA-related documentation are available for review from the NHSIA SharePoint site. Updates and any additional documents will be published on that site. The URL for the site is https://partners.jhuapl.edu/sites/HSNIA. The version D0.1 documents may be viewed or downloaded from the document library named NHSIA_D0.1.

Review and comments to this document are welcome. To comment, either post your feedback in the NHSIA_D0.1_Comments library or send comments to:

Mr. John J. Tamer
The Johns Hopkins University Applied Physics Laboratory
11100 Johns Hopkins Road
Laurel, MD 20723
Phone: 240-228-6000
E-Mail: John.Tamer@jhuapl.edu

Table of Contents

List of Figures

List of Tables

This page intentionally blank

1 Introduction

1.1 NHSIA Overview and Objectives

The National Human Services Interoperability Architecture is being developed for the Administration for Children and Families (ACF) as a framework to support common eligibility determination and information sharing across programs and agencies, improved delivery of services, prevention of fraud, and better outcomes for children and families. It consists of business, information, and technology models to guide programs and states in improving human service administration and delivery through improved interoperability of business processes and information technology (IT).

The primary goal of the NHSIA Project is to develop a national architecture to enable information exchange and sharing IT services across currently independent federal, state, local, and private human service information systems. It is envisioned that the ultimate outcome for stakeholders following NHSIA guidance will be:

- Interoperability of IT elements and associated business processes
- Improved care provided to clients by holistically addressing their needs – e.g., "no wrong door"
- Comprehensive, integrated support for client-oriented case workers at point of service
- Incremental insertion of new services and technology
- More flexible, adaptive systems
- Reduced cost of operation and maintenance for all levels of government and the private sector through sharing and reuse of services, data, and IT resources
- Reduced fraud through automated and coordinated enrollment, verification and eligibility determination
- Greater availability of timely program data for evaluating program performance
- Better connections between human services and health and education services, and able to leverage advances made in those areas

1.2 Architecture Framework and Viewpoints

An **architecture** is a description of the components, structure, and unifying characteristics of a system. An enterprise architecture is a rigorous, comprehensive description of an enterprise, including mission and goals; organizational structures, functions, and processes; and information technology including software, hardware, networks, and external interfaces. NHSIA can be thought of as a multi-enterprise, or **community architecture**.

An **architectural framework** is a structure for describing an architecture. The NHSIA project has adapted the frameworks defined by the Federal Enterprise Architecture (FEA)[1] and the DoD Architectural Framework (DoDAF)[2], and has incorporated applicable features of the Medicaid IT Architecture (MITA) Framework[3]. DODAF has evolved over a decade to include multiple viewpoints. NHSIA has adapted DODAF to include the viewpoints shown in Figure 1–1. The adaptations include merging the DODAF Systems and Services viewpoints into a single Systems Viewpoint and pulling out an Infrastructure Viewpoint as a separate item from the systems viewpoint.

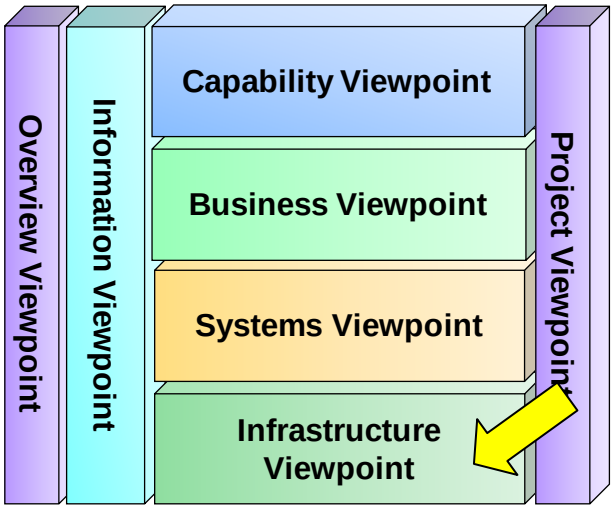


Figure 1–1. Architecture Viewpoints

1.3 Architecture Documentation

NHSIA is documented by a viewpoint description for each viewpoint. Each of these viewpoint descriptions is supported by more detailed documents including white papers, spreadsheets, diagrams, presentations, and products of specialized architectural tools. The viewpoint descriptions and associated products are referred to as architectural artifacts. This viewpoint description document addresses the Infrastructure Viewpoint.

[1] http://www.whitehouse.gov/omb/e-gov/fea/
[2] DoD Architecture Framework, version 2.0, Volume 1: Introduction, Overview and Concepts, Manager's Guide, 28 May 2009.
[3] https://www.cms.gov/MedicaidInfoTechArch/

2 Infrastructure Viewpoint Summary

The Infrastructure Viewpoint describes the technical underpinnings of the planned NHSIA architecture.

2.1 Infrastructure Viewpoint Description

The Infrastructure Viewpoint description includes the components necessary to facilitate interoperability among participants in the health and human services environment ranging from the Federal government to the individual beneficiary of services. Since the vast majority of human services agencies, organizations and providers will likely already have made significant investments in systems and technology infrastructure, this viewpoint presents an approach that works with the constraints of already established infrastructure environments, but is geared towards leveraging advances in technology with the goal of not only increasing interoperability, but reducing costs and improving process efficiency. The Infrastructure Viewpoint will include descriptions of patterns that can be used by government and private implementers to guide their solution architectures. An example of a pattern is shown in Figure 2–1.

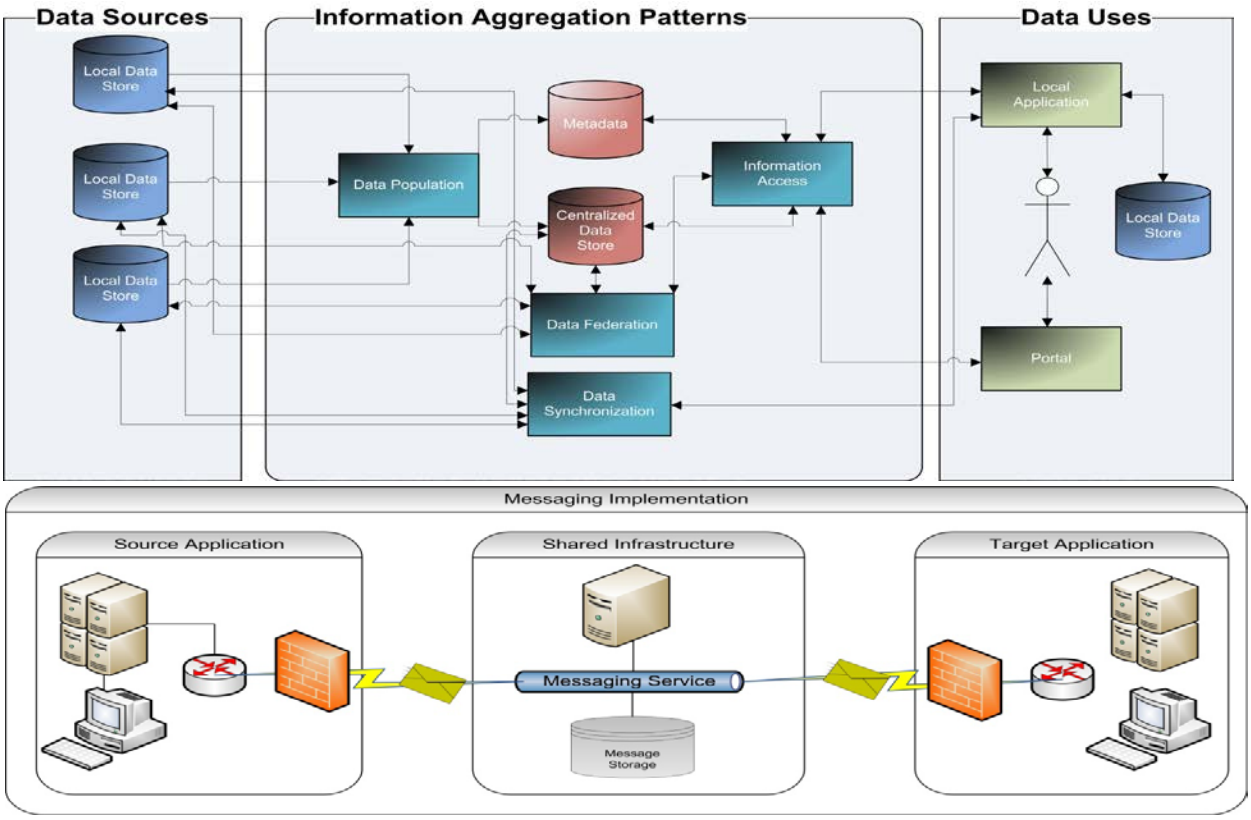


Figure 2–1 NHSIA Infrastructure Viewpoint Architecture Pattern

2.2 Description of Infrastructure Viewpoint Artifacts

The artifacts included in the Infrastructure Viewpoint are summarized in Table 2–1. These are all included in the sections of this document which follow.

Table 2–1. Infrastructure Viewpoint Artifacts

Artifact	Form: & Description
Key Infrastructure Concepts and Components	Form: Illustrations and descriptive text
	Description: Describes some of the key infrastructure technology trends and their uses and benefits in the NHSIA architecture.
Architecture Patterns	Form: Diagrams and descriptive text
	Description: Focuses on the types of infrastructure implementations needed to achieve interoperability and presents a series of patterns that address common cross-organizational interoperability and integration challenges.
Pattern Use Cases and Implementation Considerations	Form: Diagrams and descriptive text
	Description: Describes the possible uses of the architecture patterns in the NHSIA context and presents factors that may influence implementation decisions.

3 Fundamental Infrastructure Concepts

This section describes the fundamental concepts that are essential to achieving a high level of interoperability among NHSIA participants. Many organizations have already or in the process of implementing these concepts within their own infrastructure environments. The key for NHSIA interoperability will be coordinated efforts among participants. These fundamental concepts include:

- Service Oriented Architecture
- Enterprise Service Bus
- Cloud Computing
- Infrastructure Security

3.1 Service Oriented Architecture

Service-oriented architecture (SOA) is a methodology for systems development and integration where functionality is grouped around business processes and packaged as interoperable services[4]. SOA also describes an IT infrastructure which allows different applications to exchange data with one another as they participate in business processes. The aim is a loose coupling of services with operating systems, programming languages and other technologies which underlie applications.

SOA separates functions into distinct units, or services, which are made accessible over a network so that they can be combined and reused in the production of business applications. These services communicate with each other by passing data or by coordinating an activity between two or more services.

SOA defines a standard method for requesting services from distributed components and managing the results. Because the clients requesting services, the components providing the services, the protocols used to deliver messages, and the responses can vary widely, SOA provides the translation and management layer mediates these interactions. With SOA, clients and components can be written in different languages and can use multiple messaging protocols and networking protocols to communicate with one another. SOA provides the standards that transport the messages and makes the infrastructure to support it possible.

Within SOA, a service is a discrete and repeatable task within a business process. A service may be as granular as necessary. In fact, the more granular the service,

[4] Erl, Thomas (2005). Service-oriented Architecture: Concepts, Technology, and Design. Upper Saddle River: Prentice Hall PTR.

the more likely it is that it will be reusable. A combination of services that together comprise a complete business process is a composite service. A business process will access one or more services by passing messages containing both data and metadata to be acted upon. The service acts on that message and returns a response that the business process then uses for its own purpose. A response may be a simple status message or it may be the result of a query or system process. A common example of a message in an SOA is an XML file transported using a network protocol such as Simple Object Access Protocol (SOAP).

In the context of SOA, a service provider is the organization that develops the service, publishes it, and makes it available to service consumers. Service consumers are organizations or systems that discover the service and incorporate it into their own processes.

Usually service providers and service consumers do not pass messages directly to each other. Instead, implementations of SOA employ middleware software to play the role of transaction manager and translator. In addition, middleware can discover and list available services, as well as potential service consumers, often in the form of a registry. The Universal Description Discovery and Integration (UDDI) protocol is the one common type of registry used to broadcast and discover available Web services.

With respect to the NHSIA infrastructure, a Service-Oriented Architecture, then, is an architectural style for creating an IT infrastructure that exploits the principles of service orientation to achieve a tighter relationship between the business and the information systems that support the business. These information systems may be local, or cross-organizational.

Service-Oriented Architecture is one of the key features of the MITA 2.0 technical architecture. According the MITA framework, "service-oriented architecture (SOA) is a software design strategy that packages common functionality and capabilities (services) with standard, well-defined service interfaces, to produce formally described functionality that can be invoked using a published service contract. Service users need not be aware of "what's under the hood." A service can be built using new applications, legacy applications, COTS software, or all three. Services will be designed so that they change to support State-specific implementations.[5]"

In establishing a business case for use of SOA, MITA lists a number of benefits. While these are specific to the Medicaid enterprise, the same benefits will accrue to NHSIA. The following are adapted from the MITA framework:

[5] Service-Oriented Architecture — A Primer, Medicaid Information Technology Architecture (MITA), Center for Medicare and Medicaid Services, No Date, https://www.cms.gov/MedicaidInfoTechArch/Downloads/mitasoa.pdf

- Enables Increased Business Agility—for many health and human services organizations, business change and uncertainty are common occurrences. Business innovation to respond to these occurrences 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—most business 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, SOA allows IT to be introduced without upsetting enterprise business processes. With an SOA, business needs will drive the enterprise, not technology.
- SOA Facilitates Greater Reuse—common services for shared business processes can be developed and reused across many organizations. Reuse typically has three benefits: lower cost, reduced development schedule, and lower implementation risk. By fostering collaborative development of services, SOA can reduce significantly the time required to develop new capabilities, while lowering both the cost and risk of their implementation.
- Facilitates Insertion of New Technology—SOA requires 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.

3.2 Enterprise Service Bus

An **enterprise service bus (ESB)** is fundamental component of the SOA infrastructure. Its presence in the architecture should be transparent but an ESB is fundamental to simplifying the task of invoking services, making the use of services wherever they are needed, independent of the details of locating those services and transporting service requests across the network to invoke them wherever they reside.

While it is a fundamental component of the SOA infrastructure, there is no standard definition of an ESB. However, for the purposes of NHSIA, an ESB can

be defined as a software-based infrastructure component that acts as an intermediary layer of middleware to address integration and interoperability among web services and eliminate the need for multiple point-to-point connections. An ESB adds flexibility to communication between services, and simplifies the integration and reuse of services. An ESB makes it possible to connect services implemented in different technologies, acting as a mediator between different, otherwise incompatible, protocols and data formats. An ESB usually provides capability to route the messages to different services based on their content, origin, or other attributes and a transformation capability to transform messages before they are delivered to services. A primary benefit of using an ESB is that it provides a scalable way to connect a large number of applications without the need for multiple point to point connections.

Adapters are another important component of an ESB. Adapters exist at the edges of the ESB and allow it to interact with different input and output formats. This allows the ESB components to receive input from any protocol and send output to any protocol. Depending upon the ESB, adapters will be provided for a variety of common transformations, such as XML documents or SQL databases. In addition, adapters are often available for industry-specific transactions such as HL-7. Use of adaptors will be a key component in achieving semantic interoperability in NHSIA. Architecture patterns that employ data mediation will make use of adapters.

Most COTS ESB solutions include the capability to automate the execution of business processes by combining existing Web services into a **composite service**. A business process, in this context, is defined as a collection of related, structured activities or tasks that produces a specific product or service for a particular customer. A process begins with a customer's need and ends with the fulfillment of that need. An example of a NHSIA business process is Enrollment and Eligibility.

In an SOA environment, automation of a business process workflow is accomplished by means of **orchestration**. Through orchestration tools, Web services can be aggregated, integrated, and nested. Sequential, conditional, parallel, time-based execution and other dependencies can be defined among orchestrated web services. The **Business Process Execution Language (BPEL)** is a language for describing a business processes as a representation of coordinated Web service invocations and related activities that produces a result, either within a single organization or across several organizations. An orchestration engine, running in concert with the ESB will execute the BPEL which will invoke associated Web services necessary to complete the process. This will provide the capability to automate business process workflows across organizational boundaries.

A **business rules engine** is another component typically employed as a part of, or in conjunction with an ESB. A business rules engine is a software system that executes one or more business rules in a runtime environment. These rules might

come from legal regulations, company policy, or other sources. In any case, a business rules engine enables such rules to be defined, tested, executed and maintained separately from application code. This provides a mechanism to adapt to changing rules without the need for major system updates.

A business rules engine typically includes a repository that permits rules to be externalized from application software. Most also include a rule definition tool that allows business experts to define and manage decision logic that was previously buried in software. Finally, the rules engine allows complex and inter-related rules to be executed based on specific business context, using a combination of data inputs, sets of applicable rules, and execution algorithms that define how to process the data and rules to provide an output.

Finally, a **services registry** is one of the fundamental pieces of SOA for achieving reuse. Typically a component of the ESB, it is a place in which service providers can document information about their offered services and potential service consumers can search for services. The reuse of services greatly depends on the ability to describe and publish the offered functionality of the services to potential consumers. A service registry organizes information about services and provides facilities to publish and discover services. **Universal Description Discovery and Integration (UDDI)** and the **Web Services Description Language (WSDL)** are standards for describing services and their providers, as well as how services can be consumed. In particular, UDDI provides an infrastructure that supports the description, publication, and discovery of service providers; the services that they offer; and the technical details for accessing those services.

3.3 Cloud Computing

The Federal Cloud Computing Strategy [6] begins with the premise that "cloud computing has the potential to play a major part in addressing … inefficiencies and improving government service delivery." It goes on to say that the "cloud computing model can significantly help agencies grappling with the need to provide highly reliable, innovative services quickly despite resource constraints."

To achieve the intended benefits of cloud computing, the Federal government has instituted a "Cloud First" policy to accelerate the pace of cloud adoption by requiring agencies to evaluate cloud computing alternatives prior to making any new technology investments. Therefore, by leveraging shared infrastructure and economies of scale, cloud computing presents a compelling business model in which organizations "will be able to measure and pay for only the IT resources they consume, increase or decrease their usage to match requirements and budget

[6] Vivek Kundra, Federal Cloud Computing Strategy, February 8 ,2011

constraints, and leverage the shared underlying capacity of IT resources via a network." This same compelling business case applies to NHSIA as well.

The Burton Group defines cloud computing as "The set of disciplines, technologies, and business models used to deliver IT capabilities as an on-demand, scalable, elastic service [7]. The National Institutes of Standards and Technology (NIST) put forth what is quickly becoming the industry standard definition of cloud computing. According to NIST, "Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models[8]." NIST definitions are included in italics throughout the following subsections.

3.3.1 Essential Characteristics

NIST has defined five essential characteristics of Cloud computing. These are:

- **On-demand self-service.** *A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service's provider. In other words, a user can request computer resources as needed.* These resources are deployed in the Cloud and made available to the user as needed.
- **Broad network access**. *Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, laptops, and PDAs). In*

[7] Drue Reeves, "Cloud Computing: Transforming IT" Burton Group, April 2009, http://www.burtongroup.com/Client/Research/Document.aspx?cid=1681

[8] Peter Mell, Tim Grance, The NIST Definition of Cloud Computing, NIST Special Publication 800-145, January 2011, csrc.nist.gov/publications/drafts/.../Draft-SP-800-145_cloud-definition.pdf

other words, Cloud-based resources are accessible via a network connection. This may be via the public internet or via dedicated network infrastructure.

- **Resource pooling.** *The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or datacenter). Examples of resources include storage, processing, memory, network bandwidth, and virtual machines.* In this case, the provider may be a Cloud service provider or the internal IT organization. Consumers, or tenants, may be members of the same organization or may span organizational boundaries.
- **Rapid elasticity.** *Capabilities can be rapidly and elastically provisioned, in some cases automatically, to quickly scale out, and rapidly released to quickly scale in. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be purchased in any quantity at any time. While capabilities may appear to be unlimited, in fact, virtual resources are limited by the availability of physical resources.* The key point is that resources can quickly be deployed or retracted to meet changes in demand without the typical lags associated with procurement and deployment.
- **Measured Service.** *Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service.* This provides an accurate means to apply a "pay as you go" model for pricing of Cloud services.

3.3.2 Service Models

NIST has also defined what it calls "service models" for Cloud computing. These are essentially the layer at which one enters the Cloud. This may be at the infrastructure level, at the platform level, or at the application level. Each of these is defined further below.

- **Cloud Software as a Service (SaaS).** *The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through a thin client interface such as a web browser (e.g., web-based email). The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.*

  This is the highest layer at which an organization may enter the Cloud. An organization is essentially renting the use of a software application. In addition, this layer also includes the next two layers in that the application must run on a platform and the platform must exist on an infrastructure.

- **Cloud Platform as a Service (PaaS).** *The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly application hosting environment configurations.*

  In this case, the user is entering the Cloud stack one layer below the application and is given access to a platform upon which to develop or implement organization-specific software. In other words, PaaS is the middle layer of the software stack "in the cloud." Platforms provide technology that intermediates between the underlying system infrastructure (operating systems, networks, virtualization, storage, etc.) and overlaying application software. A PaaS will likely include application containers, application development tools, database management systems, integration brokers, portals, business process management and many others.

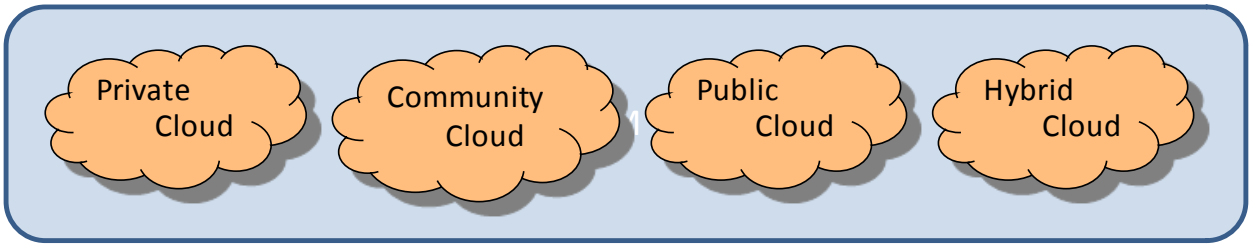
- **Cloud Infrastructure as a Service (IaaS).** *The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly limited control of select networking components (e.g., host firewalls).*

  This is the lowest entry point into the Cloud stack and constitutes the most basic and fundamental form of cloud computing services. Infrastructure services include system-level capabilities, such as server/computing, server operating systems, client operating systems, storage or networking, on which

the users can develop or install and run applications. Virtualization is often a key enabling technology for infrastructure as a service architectures.

In addition to the three service models defined by NIST, an additional service model, Software Infrastructure as a Service (SIaaS) has begun to emerge as well. SIaaS is a stand-alone cloud service that provides a specific application support capability, but not the entire application software platform service. For example, Microsoft SQL Data Services is a SIaaS offering that is available as a stand-alone infrastructure service separate from Azure, Microsoft's PaaS. The intended users of SIaaS are organizations who want to create applications that do not have dependencies on internal infrastructure components, which can be too expensive to license, slow to deploy, and complex to maintain and support. Other examples of SIaaS include messaging services, such as Amazon Simple Queue Service, integration services, such as Cast Iron Systems, and content distribution services such as Akamai.

3.3.3 Deployment Models



Finally, NIST [MEL2011] has defined three "deployment models" that describe the location and tenancy of a Cloud environment. These are:

- **Private cloud.** *The cloud infrastructure is operated solely for an organization. It may be managed by the organization or a third party and may exist on premise or off premise. This is also referred to as an enterprise cloud.*

  A private cloud will usually be in-sourced and run on-premises using equipment owned by the organization, but that is not always the case. Private cloud computing can also be outsourced and externally hosted. This is often called a "virtual private cloud."

- **Community cloud.** *The cloud infrastructure is shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be managed by the organizations or a third party and may exist on premise or off premise.*

  Again, this Cloud environment may be built and hosted by one of the community organizations or it may be hosted by a Cloud provider.

- **Public cloud.** *The cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services.*

- **Hybrid cloud.** *The cloud infrastructure is a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds).* In other words, a hybrid cloud is any combination of the previous three cloud deployment models. An example of hybrid cloud deployment may consist of an organization deploying noncritical software applications in the public cloud, while keeping critical or sensitive apps in a private cloud, on the premises. In addition, hybrid clouds often employ a "cloudburst" concept. A cloudburst generally refers to the ability to redirect resources from a private cloud to a public cloud in the event of a spike in demand.

3.3.4 Benefits of Cloud Computing for NHSIA

The starting point for examining the benefits of cloud computing for NHSIA is the fact that a shared information technology environment is a key to effective interoperability. Therefore, the advantages of cloud computing can be characterized as follows:

3.3.4.1 Enhanced Interoperability

- A common environment, hosted in a cloud, eliminates the need for multiple point-to-point connections for secure exchange of information and access to applications and services. Organizations will require only one connection to the shared environment.
- A common environment with adequate storage provides a standardized means for storing and accessing shared data. Data can be sent to the common environment, transformed into a standard format and stored for subsequent retrieval. Combining data in a common environment means that organizations have a centralized starting point for access to shared data. Further, data can be retrieved without imposing a processing burden on the source system.
- Storing data in a common environment provides a repository which supports multi-organization and cross-organization data analysis and reporting.
- Provides a secure, controlled environment for social networking and collaboration. With secure communication between partner organizations and the common environment, collaboration can be conducted in a private and secure manner.
- A shared or multi-tenant cloud environment provides NHSIA with the ability to develop and deploy common applications and services. That is,

rather than running an application instance for each organization, as is done for on-premises, multiple organizations may use a single instance of the application simultaneously. This is analogous to the START concept of deploying a common provider management system in a multi-tenant cloud environment. This approach allows fixed application costs to be amortized over a large number of customers. In a single-tenant instance, each organization pays for its own application development and management. For example, in an on-premises instance, the same activities, such as applying software patches, are performed multiple times, once for each instance. In a multi-tenant cloud environment, that cost is shared across a large set of organizations.

3.3.4.2 Economies of Scale

- Power, space and cooling costs represent a significant portion of data center operating costs. With cloud computing minimized power, space and cooling as well as other and operational costs are gained through large-scale dynamic data centers built on commodity hardware and managed through best practices.
- In many organizations, utilization rates for IT assets tend to be low relative to available capacity. In most data centers, each application/workload typically runs on its own physical server. This means the number of servers scale linearly with the number of server workloads. The result is that organizations invest more capital than is necessary in assets that are underutilized. With cloud computing, high utilization rates as well as the ability to manage variability in resource demand are achieved via aggregation of customer demand across participating organizations in multi-tenant cloud environments.
- The ability to manage variations in resource demands comes from the elasticity provided by a cloud environment. In other words, a cloud facilitates the ability to scale up and down based on peaks and valleys in demand. Potential gains are achieved through lower overall costs over the course of the year because capital is not invested to meet peak demands.
- Establishing a cloud computing environment reduces both capital expenditure and risk. Organizations avoid large upfront capital investments while gaining the ability to deploy new capabilities quickly with minimal financial risk.
- In a cloud environment, organizations can rapidly provision and integrate of services without the need to go through traditional procurement processes, approvals and deployment chains. This allows projects to be completed in less time and speeds time necessary to realize a business benefit.

3.3.4.3 Limited Financial Risk

- Because organizations do not need to make up-front capital investments to create a Cloud infrastructure, there is a low barrier to entry. Organizations gain access to systems and infrastructure for a relatively small investment compared to building a similar capability. Further, infrastructure can be made available more quickly to allow organizations to begin realizing benefits more quickly.
- A typical pay-as-you-go Cloud model matches resources to need on an ongoing basis. The means that capacity can grow to meet peak demand without the need to anticipate peaks or invest in extra, underutilized capacity.

3.3.5 The Economics of Cloud Computing

Cloud computing presents the opportunity for organizations to develop new or enhanced capabilities with the potential for cost savings compared to developing those same capabilities individually. For NHSIA, cloud computing presents an opportunity to create an infrastructure that can be shared by multiple organizations to provide a foundation for sharing of information, use of common applications and services, enhanced collaboration, and other types of interoperability. The actual business case for cloud computing will depend highly on the desires and capabilities of the participating organizations as well as the cloud computing deployment model selected. This section discusses the relevant cost and business case factors.

The four deployment models for cloud computing (see Section 2.3) each have distinct trade-offs to be considered.

3.3.5.1 Private Cloud

With a private cloud deployment, the owning organization or organizations will incur an initial capital investment for the facility and infrastructure as well as costs for development, transition and ongoing operations. These ongoing costs will typically include licensing, maintenance contracts, support staff, electricity for IT hardware, and heating, ventilation and air conditioning (HVAC) for the facility.

The assumption here with respect to a private cloud is that an organization will establish its own private cloud infrastructure rather and establish a hosted private cloud. The costs associated with a hosted private cloud will be similar to those associated with a public cloud as discussed below.

Private clouds can offer significant savings through reductions in hardware and associated expenses. This is based on the assumption that, through sharing of resources, utilization of existing assets is increased thereby reducing the total number of assets required. Consequently, this reduces capital expenditures and operating costs. In addition, a smaller hardware footprint will result reduced maintenance as well as reduced use of electricity, building space and HVAC.

Another savings can come from the reduction in administrative staff that results from a fewer and more standardized environments to manage.

It is further possible that savings will be realized in certain fixed costs, such as those associated with data center building, but this will likely occur over time.

3.3.5.2 Community Cloud

The business case for creating a community cloud is similar to that for creating a private cloud. The fundamental difference is that, in the case of a community cloud, multiple organizations invest in and use the cloud environment. Costs will primarily be driven by the need for new or increased infrastructure capacity and addition support staff. Of course these will vary based on projected usage requirements as well as the level of support required.

As with a private cloud, the assumption here is that the participating organizations will jointly fund the development of the community cloud. Funding to be provided by individual organizations may be determined by projected usage relative to total capacity. Alternatively, a community cloud may be hosted by a commercial cloud provider. In this case, the business model will be similar to that for a public cloud. The primary difference is that, with a hosted community cloud, costs will be shared among a multiple organizations and may therefore generate economies of scale.

3.3.5.3 Public Cloud

The most significant change in the business case for a public cloud is the fact that, when acquiring cloud services, the acquiring organizations do not need to make the significant capital investments necessary to build their own data center capabilities. Since most commercial cloud providers are able to provision resources out of existing capacity, there is no need to make capital investments far in advance of needing or using the resources.

The primary cost of using commercially available could services will include the usage fees for the service as well as the cost for staff to manage the relationship with the cloud vendor. These fees will depend upon the nature of the contract and the resources consumed. While fees will vary according to usage, organizations will not incur the fixed costs associated with data center infrastructure and facilities.

As with any of the cloud deployment models, there is a cost associated with migration applications and services to the new cloud environment. With private or enterprise-hosted clouds, the migration costs may be less since the cloud infrastructure can be designed to accommodate existing applications. With a public cloud, use of proprietary environments may require additional effort. Regardless, moving applications and services to the cloud will require some level of integration and testing. In addition, migration of data and any required transformations must also be accounted for.

3.3.5.4 Other Considerations

A recent study by Booz Allen Hamilton[9] analyzed the costs and benefits associated with migrating to each of the cloud deployment models as compared to maintaining a status quo data center environment. The study showed that the net present value and the benefit to cost ratio for each of the scenarios to be significant compared to the status quo. Specifically, the study suggested that "once cloud migrations are completed, the model suggests annual operating and sustainment savings in the 65-85 percent range." The variability in the range is due to the type of deployment model (i.e., private versus public) and the percentage of current workloads migrated to the cloud environment. Additional key findings included:

- Economic benefits increase as virtualized cloud servers replace underutilized servers in a non-cloud environment.
- The less time required to migrate to the cloud environment the greater the benefit due to the reduced cost of parallel operations.
- Greater scale efficiencies are achieved from grouping multiple small data centers into as large a cloud as possible.

3.4 Infrastructure Security Considerations

3.4.1 Building a Trusted Environment

Any discussion of security must begin with the concept of trust. Trust, as applied to security consists of three key elements:

- Predictability—the ability to consistently produce an expected (positive) result. The more predictable the security, service, and quality NHSIA, the easier it will be for organizations and clients to participate.
- Assets—whether physical or logical, must be at risk of being damaged or lost. The greater the value, the more trust becomes a requirement.
- Uncertainty—increases the need for trust. In general, if all information is known about the parties and the transactions involved, the need for trust is greatly reduced. However, this is unlikely to be the case when multiple organizations and clients are involved.

3.4.1.1 Trust Policy

One of the most effective, methods of establishing trust among human service organizations and clients is to establish well-thought-out, transparent policies of trust. These policies should cover the following:

---

9 Ted Alford, Gwen Morton, "The Economics of Cloud Computing," Booz Allen Hamilton, October 2009, http://www.boozallen.com/insights/insight-detail/42656904

- Privacy—privacy policies must be designed to ensure that a human services client understands the impact of sharing personal information within NHSIA. Given the likely dissemination of that information, it is important for a client to understand how personal information will be used. In general, a privacy policy will create a binding contract between the client and NHSIA.
- Proper Use of Information—another aspect a trust policy is the proper use of information. For example, can information be used against a person, such as in determining benefits or eligibility for a program.
- Recourse in the Event of Breach of Trust—given that it is not possible to maintain a 100 percent trusted environment all the time, recourses for disputes regarding trusted systems must be established.
- Continuity of Trust—internal mechanisms must be in place to ensure that trust is not an external promise, but also a vital part of NHSIA operations. Examples might include employee background checks, secured physical facilities, and ongoing employee training.
- User Consent—a mechanism for consent from human service clients will be required. This will include appropriate technology and systems to monitor and comply with opt-in or opt-out preferences.

3.4.1.2 Trusted Infrastructure

In addition to a comprehensive trust policy, trust in a digital environment is created by establishing a trusted infrastructure. Trusted infrastructure is the implementation of systems, applications, and processes that allow for reliable, safe processing of transactions. Components of this infrastructure include firewalls, routers, virus scanners, vulnerability assessments, and hardened servers. Individual organizations will assume the responsibility for establishing and maintaining their own trust infrastructures. The security infrastructure of the shared NHSIA environment will be the responsibility of the hosting organization. If the NHSIA shared environment is hosted by a Cloud service provider, that provider will assume responsibility for providing the necessary security infrastructure. While several commercial Cloud service providers have attainted HIPAA certification, the actual security infrastructure provided will vary among providers and should be a key determining factor in selecting one.

3.4.2 NHSIA Security Concepts

The ability to securely share and protect information is a key aspect of NHSIA. While organizations are likely to have invested significantly in securing their own environments, NHSIA complicates an already complicated area by bringing users and data together in a new, shared environment. Because of this, NHSIA must provide mechanisms to verify the identities of individuals who will access the environment, must authorize their admittance into the environment, and must control the applications and information to which those individuals have access.

NHSIA must ensure that data is communicated into and out of the shared environment securely and that it is adequately secured and protected while in the NHSIA environment.

These security concerns are applicable to NHSIA regardless of the deployment approach used. In other words, the same security issues must be considered with NHSIA is deployed in a Cloud environment or in a non-cloud shared environment. These concerns revolve around three key areas. These are:

- Identity and Access Management
- Network and Infrastructure Security
- Data Security and Privacy

The following discussion presents recommendations and considerations for each.

3.4.2.1 Identity and Access Management

Identity management and access control are fundamental functions required for secure cloud computing. The simplest form of identity management is logging on to a computer system with a user ID and password. However, true identity management for a multi-organizational environment such as NHSIA, requires more robust authentication, authorization, and access control. It should determine what resources are authorized to be accessed by a user or process and determine when a resource has been accessed by unauthorized entities.

The following are the principal elements of an identity management system:

- Authentication—confirmation that a user corresponds to the user name provided.
- Authorization—granting access to specific services and/or resources based on the authentication.
- Accounting—a process for logging access and authorization.
- Provisioning—the enrollment of users in the system and the assignment of credentials

There are three primary assumptions that will drive the approach to security in the cloud environment. These are:

- Human services workers (including providers) will require access to shared applications and resources in the cloud environment
- Human services clients will require access to their own information in the cloud environment
- Data integrity and confidentiality must be ensured. In addition, health-related privacy concerns must be addressed.

Each of these will be discussed in turn.

3.4.2.1.1 Human Service Worker Access

Human service workers will most likely have credentials (i.e., a userid and password) that allow them to access their own organization's computer network and resources. Further, most organizations have some type of directory that contains information pertaining to that organization's computer resources and users. Most directories follow a hierarchical database format, based on the X.500 standard, and a type of protocol, as in Lightweight Directory Access Protocol (LDAP), that allows both users and software applications to interact with the directory. The directory controls who has access to what within the internal IT infrastructure.

The objects within the directory are managed by a directory service. The directory service allows an administrator to configure and manage how identification, authentication, authorization, and access control take place within the network. In a Windows environment when a user logs in, he is logging in to a domain controller (DC) which has a hierarchical directory in its database. The database is running a directory service (Active Directory (AD)), which organizes the network resources and carries out user access control functionality. Once successfully authenticated to the DC, certain network resources will be available to as determined by the configuration of AD.

Human service workers should be able to use these same authentication credentials to access shared applications and resources in the cloud environment. This will be achieved via a federated identity approach. A federated identity is a portable identity, and its associated privileges, that can be used across organizational boundaries and allow a user to be authenticated across multiple IT systems.

3.4.2.1.2 Federated Single Sign-On

Single sign-on (SSO) addresses the cumbersome situation of logging on multiple times to access different resources. In the case of NHSIA, users should not be required to maintain separate sets of logon credentials to access their local and shared resources. When users must remember numerous passwords and IDs, they are more likely to take shortcuts in creating them that could leave them open to exploitation.

In SSO, a user provides one ID and password per work session and is automatically logged on to all the required systems or applications. The advantages of SSO include having the ability to use stronger passwords, easier administration of changing or deleting the passwords, and less time to access resources. The major disadvantage of many SSO implementations is that once users obtain access to the system through the initial logon, they can freely roam the network resources without any restrictions unless other safeguards are in place.

Federation takes SSO to the next level, providing a secure, standard, Internet-friendly way to share identity among multiple organizations and applications. Users

sign on once (the SSO) using their standard network login. Their identity is then transparently and securely shared with the requested application, thereby removing the additional login requirement. Since the employee's organization authenticates him or her, and the application provider can verify the authenticity of the provided federated identity, application passwords are obviated and users are able to access to applications.

Federated identity management allows multiple organizations to securely exchange user information for access to common resources. Federation bridges segregated silos of identity systems to provide organizations with the ability to secure their cross-boundary interactions. Identity federation is based on standards that provide interoperability between security domains and systems. Although there are multiple identity federation protocols, the three most widely adopted are the Security Assertion Markup Language (SAML), the Liberty Alliance protocols (ID-FF) and the WS-* ("WS-star") suite of specifications.

Different COTS vendors take different approaches to federated SSO. Virtual directories are essentially proxy servers that can assemble identity data from multiple, heterogeneous repositories. Synchronization servers focus on harmonizing data between identity repositories and are characterized by change detection mechanisms that detect when user information changes in a source repository.

3.4.2.1.3 Human Services Client Access

It is likely that clients of a human services system will also require access to cloud resources. This is based on the assumption that some combination of self-service applications and portals will be used for services such as intake, enrollment and status inquiries. Since private individuals, rather than human services employees, will require access, and since access will be via the Internet, a different approach will be required.

Web access management (WAM) software controls what users can access when using a web browser to interact with web-based applications. In this case, those applications may be running in the Cloud environment with the potential for reach back to local organizational applications via Web services.

WAM software typically requires that a user register in the cloud environment to obtain the appropriate credentials. Some WAM products are implemented to allow users to reset their own passwords. This does not mean that the users have any type of privileged permissions on the systems to allow them to change their own credentials. Instead, during the registration of a user account, the user can be asked to provide several personal questions (school graduated from, favorite teacher, favorite color, and so on) in a question and answer form. When the user forgets his password, he may be required to provide another authentication mechanism (smart card, token) and to answer these previously answered questions to prove his

identity. If he does this properly, he is allowed to change his password. If not, his login is denied.

A key concern with client access will be ensuring that the correct person is given access to the correct data. A variety of means may be employed to provision credentials to clients and will likely depend on the characteristics and capabilities of the organizations involved. One possible scenario is to provide credentials to a client as a part of the intake or enrollment process as an adjunct to the verification process.

3.4.2.2 Network Security

Secure cloud communications involves the structures, transmission methods, transport formats, and security measures that provide confidentiality, integrity, availability, and authentication for transmissions over private and public communications networks. Secure communications should ensure the following:

- Confidentiality – that only those who are supposed to access data can retrieve it. Loss of confidentiality can occur through the intentional release of private company information or through a misapplication of network rights.
- Integrity — that data has not been changed due to an accident or malice. Integrity is the guarantee that the message sent is the message received and that the message is not intentionally or unintentionally altered. Integrity also contains the concept of nonrepudiation of a message source. Nonrepudiation means that the originator of a message cannot deny the origin of the message.
- Availability — that data is accessible when and where it is needed, and that connectivity is accessible when needed, allowing authorized users to access the network or systems. Also included in that assurance is the guarantee that security services for the security practitioner are usable when they are needed.

There are a number of options available to ensure secure communications in the NHSIA environment and the appropriate option will depend on the organizations participating. However, it is a given that each participating organization will require a network to connect to the shared environment. For large organizations with multiple users requiring access to the common environment, a dedicated network line can be put in place. Alternatively, a virtual private network (VPN) can be established between a participating organization and the common environment.

A VPN is created by building a secure communications link between two nodes by emulating the properties of a point-to-point private link. A VPN can be used to facilitate secure remote access into a cloud, securely connect two networks together, or create a secure data tunnel within a network. The portion of the link in which the private data is encapsulated is known as the tunnel. To emulate a point-to-point

link, data is encapsulated, or wrapped, with a header that provides routing information. Most often the data is encrypted for confidentiality.

A VPN can also be configured to provide remote access to shared resources over the public Internet to maintain confidentiality and integrity. This configuration enables a remote user to use the Internet to create a secure connection. In other words, the VPN software creates a virtual private network between the remote user and the central VPN server across the Internet.

3.4.2.3 Data Security and Privacy

Willingness for organizations and individuals to share information in a common environment is predicated on the fact that such information will be protected against unauthorized use or disclosure. While NHSIA extends beyond health care, it is nevertheless important to consider the restrictions placed on health data by the Health Insurance Portability and Accountability Act (HIPPA). HIPPA establishes national standards to protect individuals' electronic personal health information that is created, received, used, or maintained by a covered entity.

The HIPAA Security Rule requires appropriate administrative, physical and technical safeguards to ensure the confidentiality, integrity, and security of electronic protected health information (PHI). The HIPAA Privacy Rule establishes national standards to protect individuals' medical records and other personal health information and applies to health plans, health care clearinghouses, and those health care providers that conduct certain health care transactions electronically. The Rule requires appropriate safeguards to protect the privacy of personal health information, and sets limits and conditions on the uses and disclosures that may be made of such information without patient authorization. Further, the privacy rule prohibits entities from transmitting PHI over open networks or downloading it to public or remote computers without encryption. Encryption is a key component to protect data in any environment. This is true of a Cloud environment as well and includes encryption of data both at rest (i.e., stored on site) and in motion (i.e., while being transmitted).

3.4.2.3.1 Securing Data at Rest

There are multiple ways of encrypting data at rest. Following is an outline of various forms of encryption that may serve as methods for securing data at rest in the cloud:

- Full Disk Encryption of data at the disk level—the operating system, the applications in it, and the data the applications use are all encrypted simply by existing on a disk that is encrypted. This is a brute-force approach to encrypt data since everything is encrypted. However this may impact system performance since everything must also be unencrypted to be used.
- Directory Level Encryption— the use of encryption on entire data directories. Access to files requires use of encryption keys. This approach can also be used

to segregate data of identical sensitivity or categorization into directories that are individually encrypted with different keys.
- Application Level Encryption—requires the specific application to manage encryption and decryption of its own data.

Critical to implementing any of these forms of encryption is the need to manage the keys that are used to encrypt and decrypt data. In addition, identifying recovery methods for when encryption keys are lost needs to be considered.

3.4.2.3.2 Securing Data in Motion

The two goals of securing data in motion are preventing data from being tampered with (integrity) and ensuring that data remains confidential while it is in motion. Other than the sender and the receiver, no other party observing the data should be able to either make sense of the data or alter it. The most common way to protect data in motion is to utilize encryption combined with authentication to create a conduit in which to safely pass data to or from the cloud.

Encryption is used to assure that if there was a breach of communication integrity between the two parties that the data remains confidential. Authentication is used to assure that the parties communicating data are who they say they are. Common means of authentication themselves employ cryptography in various ways. Transferring data via programmatic means, via manual file transfer, or via a browser using HTTPS, TLS, or SSL are the typical security protocols used for this purpose.

3.4.2.3.3 Public Key Infrastructure

In addition to the physical and hardware aspects of the security infrastructure, one of the key aspects of representing trust in a digital format is the ability to show some type of digital credential for that trust. Whether the need is to authenticate a user of an Internet resource or to maintain the privacy of a communication, a digital credential is required on the Internet. Generally, the most accepted form of this digital credential has been built on the foundation of a technology called Public Key Infrastructure (PKI).

Further given the requirements of integrity, nonrepudiation, and confidentiality that are imposed by HIPAA, use of a PKI presents a viable solution. "Only digital signatures, using current technology, provide the combination of authenticity, message integrity, and nonrepudiation which is viewed as a desirable complement to the security standards required by the law...The use of digital signatures requires a certain infrastructure (Public Key Infrastructure) " (45 CFR Part 142 on 43260).

PKI is an infrastructure that allows the creation of a trusted method for providing privacy, authentication, integrity, and nonrepudiation in online communications between two parties. A PKI is the combination of software, encryption technologies,

processes, and services that enable an organization to secure its communications and business transactions. The ability of a PKI to secure communications and business transactions is based on the exchange of digital certificates between authenticated users and trusted resources. A PKI will typically ensure:

- Confidentiality through encryption of data that is stored or transmitted
- Integrity through use of a PKI digital signature to identify whether another user or process modified the data
- Authenticity by passing data through hash algorithms to produce a message digest which is then digitally signed by using the sender's private key to prove that the message digest was produced by the sender
- Nonrepudiation via the digital signature which provides proof of the integrity of the signed data and proof of the origin of the data

Despite the variances in different PKI systems, they all consist of several essential components that control how digital certificates are issued and managed. As with any security system, a series of checks and balances ensures the right mix of flexibility and control necessary in an operating environment. APKI system can be broken down into three main elements [NIST2001]: a certification authority, a registration authority, and a repository.

A certification authority (CA) is similar to a notary. The CA confirms the identities of parties sending and receiving electronic payments or other communications. A registration authority (RA) is an entity that is trusted by the CA to register or vouch for the identity of users to a CA. Finally, a repository is a database of active digital certificates for a CA system. The repository provides data that allows users to confirm the status of digital certificates for individuals and businesses that receive digitally signed messages. These message recipients are called relying parties. CAs post certificates and certificate revocation lists (CRLs) to repositories.

The CA issues a public key certificate for each identity, confirming that the identity has the appropriate credentials. A digital certificate typically includes the public key, information about the identity of the party holding the corresponding private key, the operational period for the certificate, and the CA's own digital signature. In addition, the certificate may contain other information about the signing party or information about the recommended uses for the public key. A subscriber is an individual or business entity that has contracted with a CA to receive a digital certificate verifying an identity for digitally signing electronic messages.

CAs must also issue and process certificate revocation lists (CRLs), which are lists of certificates that have been revoked. The list is usually signed by the same entity that issued the certificates. Certificates may be revoked, for example, if the owner's private key has been lost; the owner leaves the company or agency; or the owner's name changes. CRLs also document the historical revocation status of certificates. That is, a dated signature may be presumed to be valid if the signature date was

within the validity period of the certificate, and the current CRL of the issuing CA at that date did not show the certificate to be revoked.

PKI users are organizations or individuals that use the PKI, but do not issue certificates. They rely on the other components of the PKI to obtain certificates, and to verify the certificates of other entities that they do business with. End entities include the relying party, who relies on the certificate to know, with certainty, the public key of another entity; and the certificate holder, that is issued a certificate and can sign digital documents. Note that an individual or organization may be both a relying party and a certificate holder for various applications.

4 Infrastructure Components

This section provides a brief summary of the infrastructure components that are recommended as a part of the NHSIA infrastructure environment.

- **Adapters**—pre-defined mappings and transformations used to move data in one format to another. Adapters are typically a component of an enterprise service bus.
- **Application Servers**—a software framework that provides an environment where applications execute. It is an infrastructure component dedicated to the efficient operations between users and an organization's backend business applications or databases. An application server is typically used for complex transaction-based applications.
- **Business Intelligence/Analytics COTS Tools**—a suite of software tools that provide the capability to generate queries, reports or displays of business intelligence data. Some tools also include built-in capabilities to extract, transform and load data. These tools usually operate in conjunction with a data base management system for storage of data and a portal or web server for interactive dashboards or displays.
- **Collaboration/Social Networking COTS Software—**a suite of commercially available software that provides the tools for collaboration and social networking.
- **Data Integration Servers**—data integration involves combining data residing in different sources and in different formats into a consistent and unified view. A data integration server is a component of the infrastructure that executes the process of transforming data from one format into another.
- **Data Warehouse/ETL COTS Software**—performs the extract, transform and load (ETL) processes that involve extracting data from outside sources, transforming it to fit operational needs and loading it into the end target, typically a data warehouse.
- **Enterprise Service Bus**—an infrastructure component that acts as an intermediary layer of middleware to address integration and interoperability among web services and eliminate the need for multiple point-to-point connections. ESBs typically provide additional capabilities that include data transformation and business process management.
- **Federated Single Sign-on**—is a means of linking a person's electronic identity and attributes, stored across multiple distinct identity management systems, in such a way as to allow that person to use the same credentials to log in to a system other than the one in which the identity is established.
- **Media Server—**a specialized type of server optimized for the distribution of electronic content, such as streaming video or audio.

- **Message-Oriented Middleware (MOM)**—software that connects separate systems in a network by carrying and distributing messages between them. The messages may contain data, software instructions, or both together. MOM infrastructure is typically built around a queuing system that stores messages pending delivery, and keeps track of whether and when each message has been delivered. MOM may be deployed as a separate component or included as part of an ESB.
- **Orchestration Engine**—a component of middleware that executes business process orchestrations or automated workflows. They process the routine logic, message passing and the request and responses of Web services. Orchestration engines typically have the capability to execute Business Process Execution Language (BPEL), a language used to define automated business processes.
- **Portal Servers**—a type of application server that provides end-user access to system resources via a Web-based interface. A portal server provides a mechanism to run "portlets," small, self-contained programs that provide content to the user through the portal.
- **Rules Engine**—a software system that executes business rules in a runtime environment. A business rules engine rules to be defined, tested, executed and maintained separately from application code. This provides a mechanism to adapt to changing rules without the need for major system updates.
- **Service Registry**—acts as a directory of available Web services and service providers. Registries, using the Universal Description Discovery and Integration (UDDI) standard, can be queried to find the locations of definitions of Web services.
- **Web Access Management COTS—**a type of identity management software that controls access to Web resources, typically providing services such as authentication management, which involves determining a user's identity. This is normally done by prompting for a user name and a password. Web access management may also include policy-based authorization that can be used to control access to resources based on identity or some associated attribute.
- **Web Servers**—is typically a process running on a server that is designed for access by HTTP clients and to host both presentation and business logic. The Web server provides the technology platform and contains the components to support access to information by users employing Web browser technology.
- **Web Services**. The W3C defines a "Web service" as "a software system designed to support interoperable machine-to-machine interaction over a network.[10]" In other words, a Web services is a software application that

[10] "Web Services Glossary". W3C. February 11, 2004. http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/.

is designed to perform a specified function, be callable from another software application, and provide a response back to the calling application.

5 Architecture Patterns for Interoperability

This section focuses on the types of infrastructure implementations needed to achieve interoperability and presents a series of patterns that address common cross-organizational interoperability and integration challenges. Five categories of patterns are discussed:

- Information Aggregation
- Collaboration
- Self-Service
- Extended Enterprise
- Business Intelligence and Analytics

5.1 Information Aggregation

5.1.1 Introduction

The ability to access and understand data regardless of its origin, location and format are keys to achieving interoperability in health and human services.

Information Aggregation patterns serve to integrate data used by multiple applications in preparation for access to that data by down steam applications or end users. The basic premise is that existing data is available in both structured and unstructured forms in application data repositories managed by other applications and is accessible via a network connection.

The Information Aggregation patterns allow users to access data that is aggregated from multiple sources.

Source systems will most likely contain data in a variety of formats, so the Information Aggregation pattern must account for these differences via data translation or mediation. In addition, the Information Aggregation pattern must extract relevant metadata and store that metadata locally while maintaining links to the source data. Finally, the Information Aggregation pattern must return data from the metadata store, any centralized data stores and source local data stores in order to present a unified and complete view of the data to end users or downstream applications.

Figure 5–1, below, presents a high-level view of the Information Aggregation patterns. To accomplish the functions above, the pattern also includes sub-patterns for Population, Data Federation, Data Synchronization and Information Access. These are discussed further below.

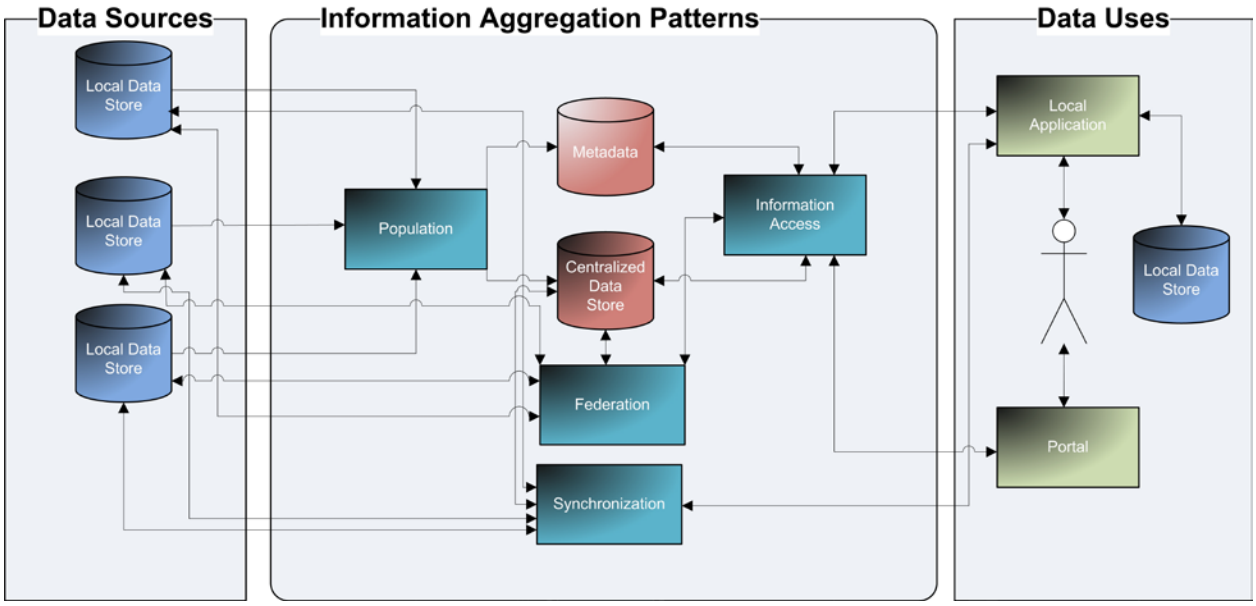


Figure 5–1. Information Aggregation Patterns

5.1.2 Federation

Federation is a fundamental pattern that can be used in a number of ways to bring together information from a variety of sources. Federation is a data integration pattern that provides access to multiple data source while giving the appearance of a single, logical data source. Target data may be local, remote, or distributed. The Federation application pattern accounts for the fact that applications may require access to many diverse data sources with the appearance that these sources are a single logical data store.

Federation may be required in any business process where the data needed exists in a number of different locations in a number of source systems of record and in a number of formats. Such diversity may be the result of historical, technical or organizational factors. When called by an application, Federation uses its metadata store to determine where and in what format the required data is stored. Metadata than maps source data to common data structures also enables the decomposition of the query into requests that each individual source is able to process. The information returned appears as one unified data set to users.

5.1.3 Population

Population involves data being copied and, if necessary, transformed, from one place to another in advance of when it is required by a user or application. Population is a one-way process where one data store is the source of the process and the other the target. The Population pattern gathers data from one or more sources, processes

that data in an appropriate way, and applies it to some data target store. The primary driver for population is to gather and reconcile data from multiple data sources in advance of a need to use the data.

The Population pattern may gather data (pull) or receive data from the source system (push). Once data is received, it is translated from diverse sources (mediation) into a common, usable format. The Population pattern also includes the necessary steps to extract metadata, match metadata to existing data, and store metadata in a central location. The metadata would most likely also contain a pointer or reference to the source record in the source system of record.

In some cases, it may also be desirable to store a sub-set of the local data centrally. Such centralized storage of data may be used to improve response time or query performance. Alternatively, data can be requested from its source for query purposes and not stored locally.

5.1.4 Synchronization

The Synchronization pattern enables a coordinated flow of data in a federated or virtual data environment. This is also known as replication. While the use of this pattern will depend largely on the data topology (see Implementation Considerations, below) and the particular requirements, the general goal of the synchronization pattern is to ensure that updates to source data are propagated to existing data replicas.

5.1.5 Information Access

The Information Access helps to structure a system that provides access to aggregated information. It is most often used, in conjunction with one of the Data Integration patterns discussed above, to provide users access to an aggregated repository created by the data population pattern. For example, the Information Access pattern would receive a service call to provide data access. The Information Access pattern would formulate the query and call a Data Federation service (provided by the Data Federation pattern) that would in turn query the central metadata store, retrieve the information from the related source systems, and present the results to the Information Access service.

5.1.6 Implementation Considerations

The notional implementation that follows is based on the assumption that a shared infrastructure will be available to host the processing and storage capabilities necessary to support the Information Aggregation patterns.

The Population pattern must be implemented in such as way as to support source data from a variety of sources in a variety of formats. Figure 5–2, below, illustrates the fact that data may come from source systems in a variety of formats. These are

the Data Source indicated on the left of the figure. Data can be created by source organizations in a variety of formats, including files, spreadsheets, XML or other messaging formation. Data can be transmitted to the shared environment by the source organization calling a Web service, sending a file using file transfer protocol (ftp), accessing a common applications system via an API, or by direct entry into a Web application.

Regardless of the means, extracting the relevant data will be the responsibility of the source organization. However, the common infrastructure must be capability of processing, and if necessary, transforming the incoming data into a useable format. This will likely be accomplished through the use of the data mediation capabilities and adaptors provided by the enterprise service bus. Figure 5–2 Population Pattern, shows several example adaptors that could be used for SQL data from a relational database, XML transactions, files, or other formats. In fact, most commercial Enterprise Service Buses (ESBs) provide a number of adaptors for this type of scenario. In addition, most also provide specialized adaptors that can be used to facilitate the intake of industry standard data formats, such as HL-7 or NIEM.

The servers show in Figure 5–2 will provide a variety of functions. An application server will process the business and transaction logic involved in accepting and storing incoming data, the Web server will manage presentation of and interaction with browser-based sessions, and the data integration server will process the data transformations. These are show as separate servers connected to a common ESB. In practice, the servers could be combined into one physical server or multiple virtualized servers. The notional infrastructure also includes storage for centralized data and metadata. This is discussed further below under Data Topology.

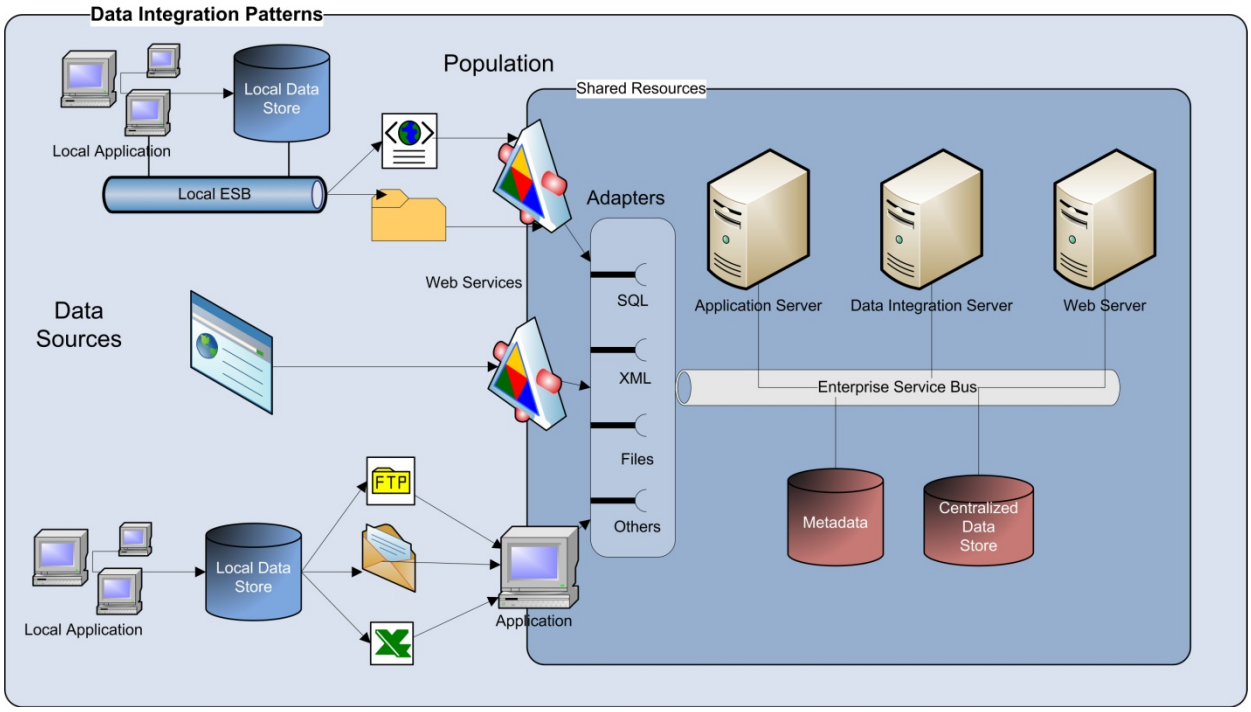


Figure 5–2 Population Pattern

Federation is essentially the reverse of the Population pattern. The actual implementation will depend highly on the technology solution. For example, most COTS database management systems provide the capability to federate data across multiple instances.

Figure 5–3 Federation Pattern, depicts a notional implementation of the federation pattern. The components shown are similar to the population pattern. For example, data may be provided by a relational database (e.g., Oracle, SQL Server), XML transaction, or some other format.

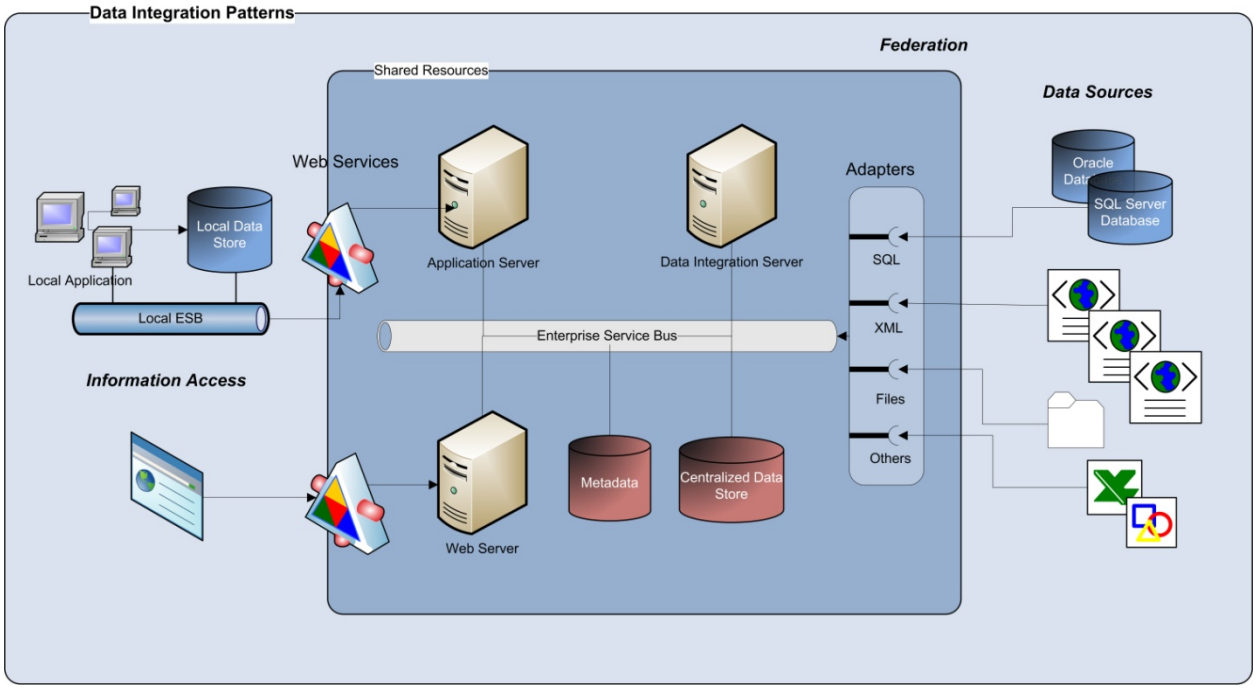


Figure 5–3 Federation Pattern

In addition, a Federation pattern will typically include a data integration server. Applications requesting access to the data via the Information Access pattern typically use standard relational interfaces and protocols to interact with the data integration server, SQL and JDBC/ODBC for example. The data integration server in turn connects through various adaptors, or wrappers, to a variety of data sources such as relational databases, XML documents, packaged applications and content management and collaboration systems. The data integration server is essentially a virtual database with all of the capabilities of a relational database.

The requesting application or user, depicted on the left of Figure 5–3, can perform any query requests within the scope of their access permissions. In this example, requesting applications interact with the data integration server via web service calls. They may be through a local application or directly via a Web browser.

When the data federation server exposes integrated information as a web service provider, a service consumer can access the integrated information through a service interface such as WSDL. The data federation server can also consume services provided by multiple information sources in order to integrate that information.

5.1.6.1 Data Topology

One of the key technology decisions will be determining the most efficient data topology. In other words, the infrastructure must be designed to achieve the most efficient access to data by providing the proper combination of local and centralized

storage. Options can range from a fully centralized model in which all data is stored centrally, to a federated model in which all data is stored locally and retrieved as needed. A hybrid model combines both central storage and a mechanism for obtaining data from participating nodes, so it is a combination of the centralized and federated models.

In the centralized model, data that is not in the central repository can never be available; in the federated model, no data can be obtained without a real-time request/response to the respective node. The hybrid model, however, provides a flexible approach, where certain categories of frequently needed data can be stored centrally for quick and reliable access directly from the single store, while further details, rarely used, or large in size data can remain in the originating nodes and can be requested when needed.

As shown in the information aggregation infrastructure above, a centralized data store could be used to contain basic client demographic information. This demographic information could be expanded to contain additional case or medical information. However, beyond that, the next level of detail would be available via pointers to the full details available on demand from their respective local storage. This pattern will provide the foundation of a master person index.

5.1.6.2 Data Transformation

Semantics concerns the study of meanings. 'Semantic interoperability' is defined by the National Alliance for Health Information Technology (NAHIT) as "the ability of different information technology systems, software applications and networks to communicate and exchange data accurately, effectively and consistently so providers can use the information as they care for patients." (http://www.nahit.org) However, beyond the technical exchange of data, *semantic interoperability* implies that the meaning of data can be comprehended unambiguously by both humans and computer programs, and that information can be processed in a meaningful way. The need for semantic interoperability requires that the infrastructure support the ability to transform data from incoming formats into the form needs by the ultimate user of the data. This is typically accomplished using an enterprise service bus and appropriate adapters.

5.1.7 Infrastructure Components Used by the Info Aggregation Pattern

- Enterprise Service Bus
- Web Services
- Adapters
- Web Servers
- Application Servers
- Data Integration Servers

5.2 Collaboration

5.2.1 Introduction

Human services workers not only need to use the information about clients and cases to make decisions, but they also need to collaborate with other participants, including service providers, family members and others associated with a case. Some of this collaboration is likely to take place through the documents and information contained in a physical (or electronic) case folder. This will include case notes, documents added to the case, and documented decisions made. Other collaboration will likely take place through person to person communication. For example, case workers should be able to communicate with others working the same case through case notes, instant messaging, email, and phone conversations. Technology supporting a case management solution needs to provide these types of collaboration technologies, and the capability to maintain the collaborations as part of the case folder for case resolution and audit purposes.

In general, collaboration can occur asynchronously, synchronously, via multicasting, or through a social networking application. Asynchronous collaboration typically involves e-mail or may involve one worker leaving case notes in a coordinated case file. Synchronous communications typically occurs in real-time and may make use of instant messaging or chat capabilities. Multicasting involves the dissemination of a message from one party to multiple other parties simultaneously. This may include such things as podcasts or videos but may also include targeted messages such as alerts and notifications.

From an infrastructure standpoint, much of the required networking and communications components are likely to be in place. These will include corporate intranets and the Internet. Also, certain key collaboration tools, such as e-mail, will most likely be in place. However, specific application software is likely to be required for more advanced collaboration activities. In addition, privacy and security are major concerns as is the ability to preserve collaborations as a part of a client's case records, where appropriate.

Figure 5–4 Collaboration Patterns, below, presents a high-level view of the Collaboration patterns. To accomplish the functions above, the pattern also includes sub-patterns for asynchronous collaboration, synchronous collaboration, multicasting, and social networking. These are discussed further in the sections that follow.

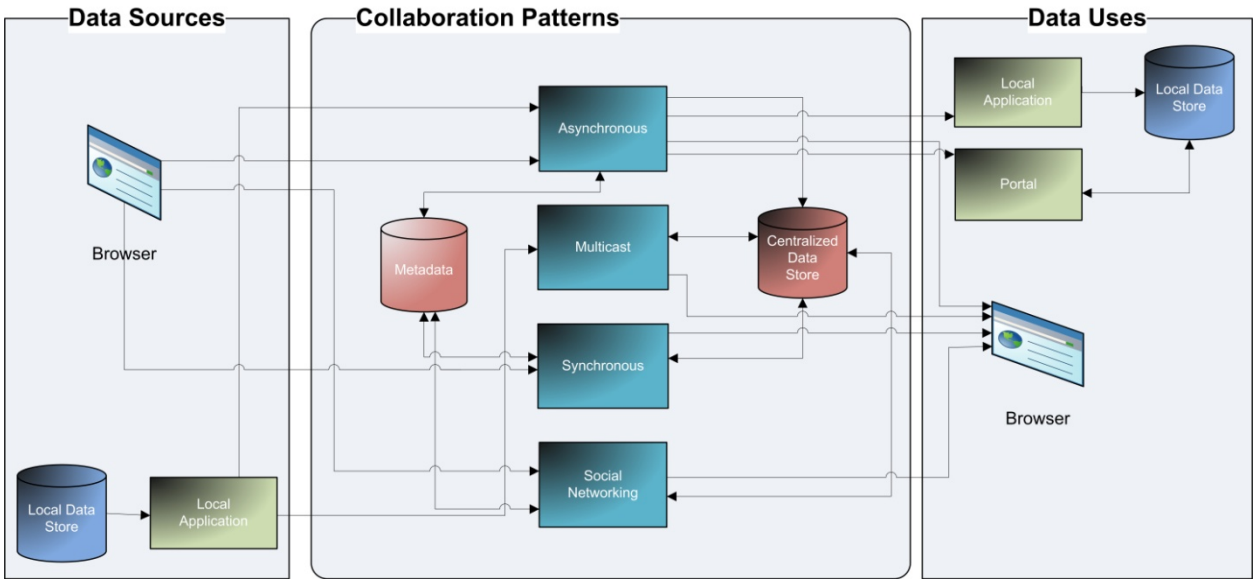


Figure 5–4 Collaboration Patterns

5.2.2 Asynchronous Collaboration

Asynchronous collaboration involves and interaction between one sender and one or more recipients, in which a sender addresses a message to another user or group of users. This may take place within an organization, on a company intra-net, or outside of the organization via the Internet. This message is essentially sent to a collector (e.g., a local e-mail server) where the intended recipient of the message picks it up. This type of communication is typically seen in traditional e-mail systems that are based on SMTP (Simple Mail Transfer Protocol) and POP (Post Office Protocol) as well as in collaboration or social network Web sites that include the ability to send messages between participants.

5.2.3 Synchronous Collaboration

Synchronous, or Interactive collaboration, is a type of collaboration in which a user collaborates with one or more other users by sharing information synchronously. This type of communication is typically implemented through services such as interactive chat rooms, bulletin boards and instant messaging services. This may take place within an organization, on a company intra-net, or outside of the organization via the Internet.

5.2.4 Multicast Collaboration

Multicast or Broadcast collaboration is a type of collaboration in which a user sends a message or a sequence of messages to multiple recipients. This includes support for broadcasting rich media such as audio and video, streaming media, as well as alerts and notifications.

5.2.5 Social Networking

Social network can be defined as “web-based services that allow individuals to (1) construct a public or semi-public profile within a bounded system, (2) articulate a list of other users with whom they share a connection, and (3) view and traverse their list of connections and those made by others within the system. The nature and nomenclature of these connections may vary from site to site.”[11]

Social network sites (SNSs) enable users to articulate and make visible their social networks to others. While this can result in connections between individuals that would not otherwise be made, in common practice, users are primarily communicating with people who are already a part of their extended social network. While SNSs have implemented a wide variety of technical features, their backbone typically consists of visible profiles that display a list of “Friends” who are also users of the system. In addition, most contain “Profiles” where one can "type oneself into being."

The visibility of a profile varies by site and according to user discretion. For example, LinkedIn controls what a viewer may see based on whether she or he has a paid account. Sites such as MySpace allow users to choose whether they want their profile to be public or "Friends only." Facebook takes a different approach—by default, users who are part of the same "network" can view each other's profiles, unless a profile owner has decided to deny permission to those in their network. Structural variations around visibility and access are one of the primary ways that SNSs differentiate themselves from each other.

The public display of connections is also key component of SNSs. The Friends list contains links to each Friend's profile, enabling viewers to traverse the network graph by clicking through the Friends lists. Most SNSs also provide a mechanism for users to leave messages on their Friends' profiles. This feature typically involves leaving "comments," although sites employ various labels for this feature. In addition, SNSs often have a private messaging feature similar to webmail. While both private messages and comments are popular on most of the major SNSs, they are not universally available. Beyond profiles, Friends, comments, and private messaging, SNSs vary greatly in their features and user base. Some have photo-sharing or video-sharing capabilities; others have built-in blogging and instant messaging technology.

11 Boyd, D. M., & Ellison, N. B. (2007). Social network sites: Definition, history, and scholarship. Journal of Computer-Mediated Communication, 13(1), article 11. http://jcmc.indiana.edu/vol13/issue1/boyd.ellison.html

5.2.5.1 Social Network Technology

Many commercial collaboration, communication, and content management systems contain social network fragments that users can leverage to interact with other people in an ad hoc manner. These systems allow participants to define social structures representing a relation to that group or to a particular individual.

A user can leverage that capability to define lists of friends, coworkers, or team members, making it easier to interact with those relations. Beyond commercial software solutions, specific destination websites can act as social networking hubs for people to connect with each other. These sites could be considered as a means to create a "corporate Facebook."

5.2.5.2 Social Network Considerations

Security is a key consideration for social network collaboration. This includes authorization, access control, and permissions. Technologies that enable social networking must support an organization's existing mechanisms to protect corporate resources and satisfy confidentiality needs, or they must extend separation of duty constructs to social network participants.

Records management and related audit trail concerns are topics that social networking projects need to identify when they assess infrastructure impacts. Social networking systems may be required to provide mechanisms to support discovery requests or provide logging features based on the nature of the social networking system. The extent to which social network interactions become a part of the permanent case record must also be determined.

5.2.6 Implementation Considerations

Social networks can be implemented in a variety of ways. An organization could purchase a COTS social network platform, such as IBM's Connections, and deploy the components on an internal or cloud-based infrastructure. Alternatively, an organization could make use of a cloud-based social network platform such as Salesforce.com's Chatter (see Figure 5–5). Chatter is a collaboration application that helps users connect with coworkers and share business information securely and in real time. It is hosted in Salesforce.com's Internet-based cloud environment and provides a number of pre-built collaboration and social network capabilities. Jive.com is another cloud-based provider of social network software and services.

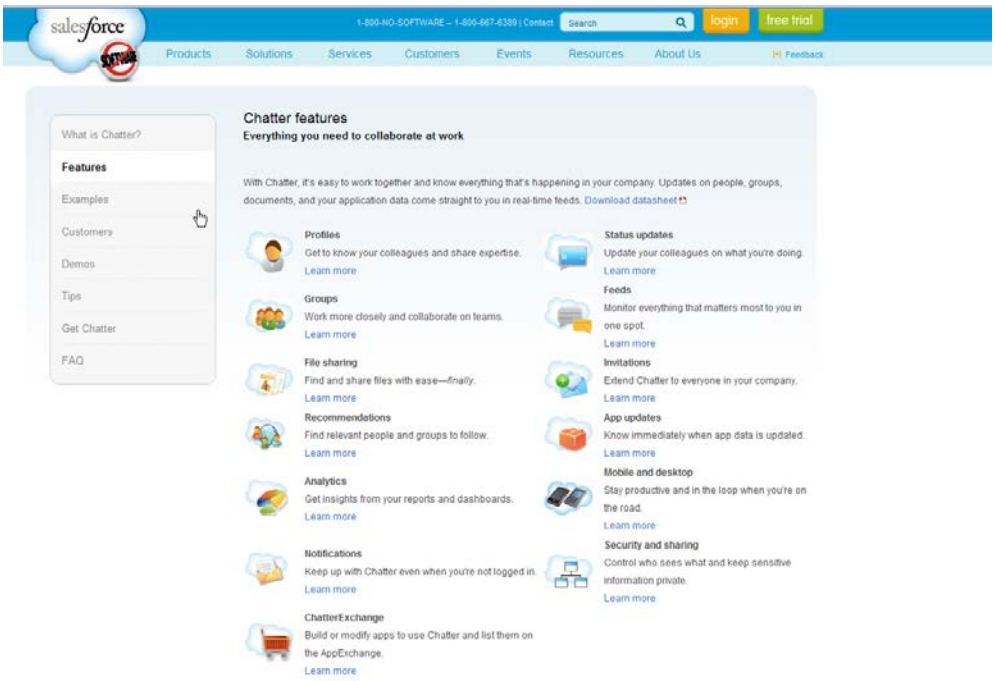


Figure 5–5 Chatter-Hosted Collaboration

Alternatively, organizations can choose to implement social network software themselves using COTS software such as IBM's Connections (see Figure 5–6), which provides similar collaboration and social network capabilities, but hosted by an organization on its own infrastructure or on a public or private cloud-based infrastructure.

Figure 5–6 IBM Connections Social Software

The notional implementation that follows is based on the assumption that collaboration COTS tool is deployed using shared resources and is accessible via the internet by parties interested in collaboration. Collaborators will connect to the shared infrastructure via the Internet, using a Secure Socket Layer (SSL) connection or via a virtual private network (VPN). The COTS software will provide the necessary collaboration applications as well as common storage areas for shared files. Additional storage will provided for media (e.g., podcasts, videos) and can be streamed via a media server. Finally, access to the shared infrastructure, and in particular, the collaboration software applications will be managed through federated single sign-on or Web access management software, depending upon the characteristics of the collaborator. This is depicted by a virtual directory server in the diagram.

Figure 5–7 depicts a variety of "collaborators" along the bottom of the figure. The access method used will depend on the type of collaboration. A Web browser is the most common means for interacting on social network sites. A chat window, either on a local desktop or in a Web browser, is a common means of synchronous collaboration. Many collaborative exchanges can also take place using smart phones or other portable computing devices. Finally, the figure contains traditional corporate e-mail servers for asynchronous communications. Alternatively, Web-based e-mail systems may be used.

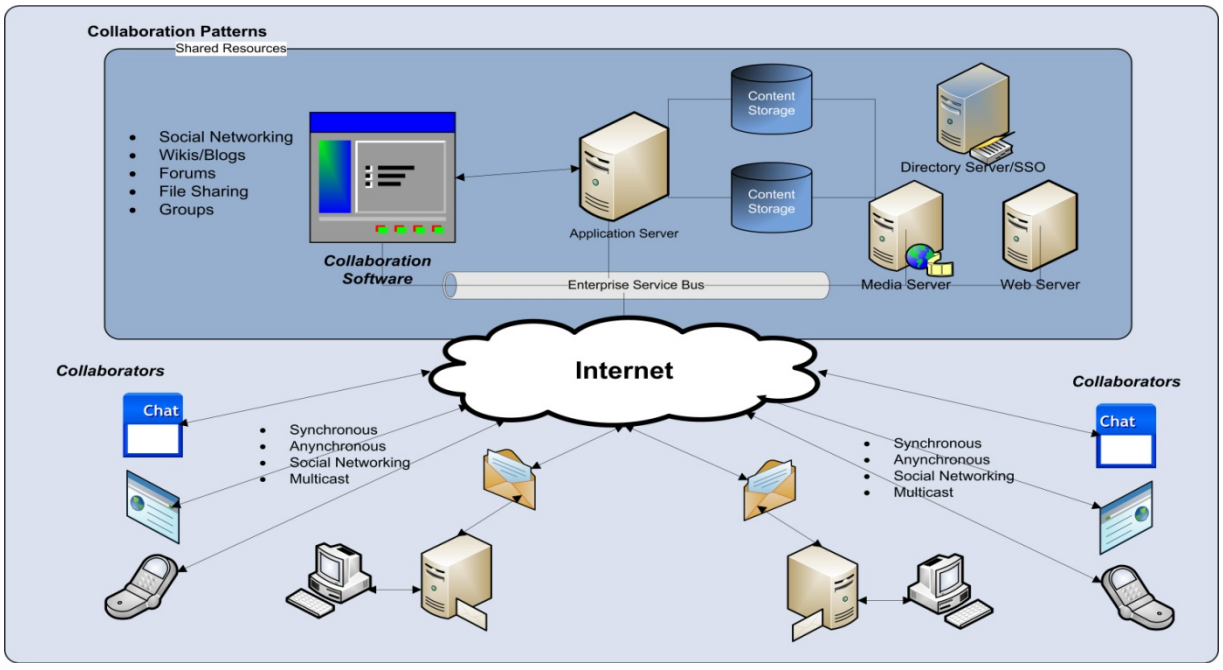


Figure 5–7 Collaboration Implementation

5.2.7 Infrastructure Components Used by the Collaboration Pattern

- Enterprise Service Bus
- Federated Single Sign-on

- Collaboration/Social Networking COTS Software
- Web Servers
- Application Servers
- Media Servers

5.3 Self-Service

5.3.1 Introduction

The Self-Service business pattern covers a wide range of uses. Applications of this pattern can range from the simple function of allowing users to view data built explicitly for one purpose, to taking requests from users, decomposing them into multiple requests to be sent to multiple data sources, personalizing the information, and recomposing it into a response for the user.

Self-service patterns are appropriate when:

- The end-users and customers need to directly interact with business processes.
- The business process needs to be integrated with existing business systems and information.
- The business process must be reachable in a common, consistent, and simplified manner through multiple access and delivery channels.

A Self-Service pattern is likely to consist of all or some of the following:

- Users may be within the enterprise, in partner organizations, or external to the enterprise in any geographic location
- Users will typically require access via a web browser or a browser-based mobile device.
- Access will be provided from any location across the Internet via TCP/IP or other Internet protocols
- Access will be provided via a dedicated LAN connection or a broadband connection
- Users may interact with systems internal to the enterprise or across multiple enterprises
- A set of interactions that represent business processes are provided to users

Figure 5–8 presents a high-level view of the Self-Service architecture pattern.

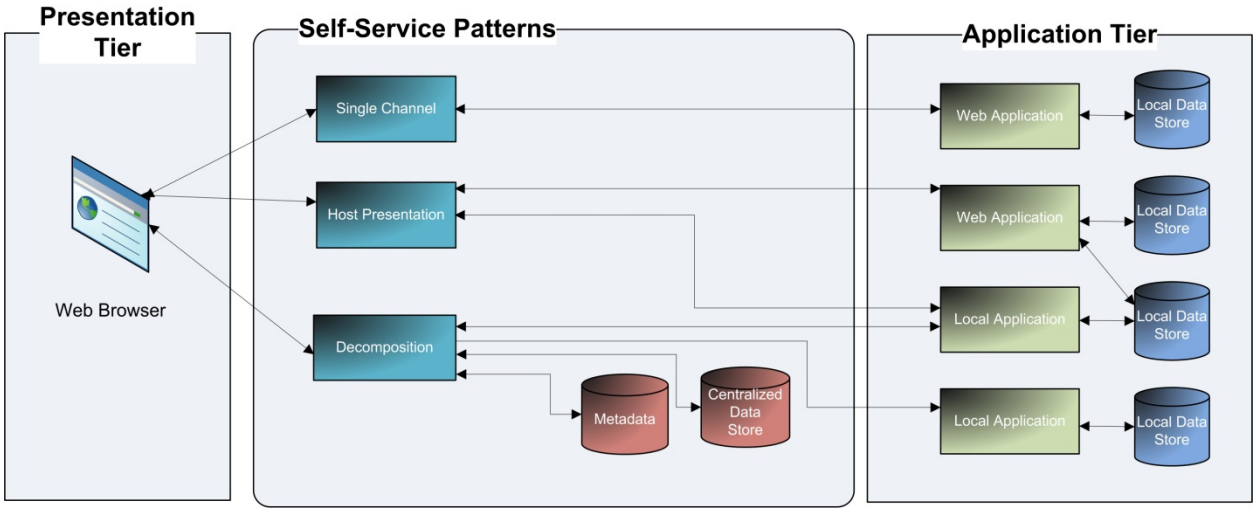


Figure 5–8 Self-Service Patterns

5.3.2 Single Channel

The Single Channel application pattern provides a structure for applications that have no current need for integration with multiple systems and need only focus on one means of delivery. While the Single Channel pattern can be used to implement a variety of the delivery channels, the assumption here is that delivery will focus primarily on the Web using a point-to-point connection to local applications and databases.

The Presentation Tier is responsible for all the presentation logic of the application. In addition, a Web application tier is typically responsible for implementing some of the business logic and for accessing back end application logic and data. Finally, the Application Tier may exist as a new application, a modified current application, or an unmodified current application. The data will typically also reside on this tier and is most likely accessible only through the existing back end application.

Interaction between the first and second tier is synchronous, involving an end user and a browser session. The interaction between the Web application and the application tier can be either synchronous or asynchronous. The chosen interaction between these tiers depends on the characteristics and capabilities of the back end system. For example, if the back-end system is primarily batch-oriented, the interaction will be asynchronous. If the back-end system is interactive, the interaction will be synchronous. Web application servers or message-oriented middleware are often used to implement this pattern.

The single channel pattern works best with applications that have simple integration requirements. However, it can increase the processing efficiency and reduce the latency of business events by providing real-time access to business data and business logic resident in legacy systems. Further, direct access to backend

applications reduces the duplication of business logic across multiple tiers. As a result, changes to business logic can be made in one tier rather than in multiple applications.

However, because this pattern relies on point-to-point interfaces, it becomes relatively inflexible and cumbersome since a user could be required to access multiple systems from multiple web sites. Under such circumstances a more advanced patterns, such as the Decomposition patterns would be more appropriate.

5.3.3 Host Presentation

The Host Presentation pattern is intended to provide Intranet access to existing host applications that may previously have only been available to employees through green screen devices or PCs with terminal emulators. The primary driver for choosing this pattern is to minimize costs by providing browser-based access to existing legacy applications without rewriting or significantly modifying them. When this pattern is used, the existing application remains as is. Even though the delivery mechanism used is browser-based, the presentation continues to look and behave the same as existing green screens. It uses synchronous interaction between the presentation tier and the existing application tier. Both presentation and business logic continue to run inside the existing host application.

Legacy applications often require a significant amount of training. Therefore, this presentation style is normally suited for deploying green screen applications through the Intranet to internal users. This same strategy may be used to deploy legacy applications to external users and business partners. However, with this pattern, the legacy user interface continues to look and feel the same. Because of this, the presentation does not take advantage of the user-friendly features of that a browser interface could provide and will most likely not be suitable for general access by clients.

5.3.4 Decomposition

The Decomposition pattern provides a structure for applications that require the intelligent routing of requests from user access to one or more application systems. Access channels are likely to include the Internet via a Web browser, the Internet via a personal computing device (e.g., smart phone), an integrated voice response system, or a walk-up kiosk. The Decomposition pattern decomposes a single request from a client into several, simpler requests as needed and intelligently routes them to multiple applications as necessary. For example, this pattern could support a system designed to gather client information in one transaction and then submit eligibility or enrollment information to multiple systems. A major benefit of the Decomposition pattern is that it provides a customer-centric view rather than a product- or organization-centric view of information and processes.

A presentation tier is responsible for the user interface, including data formatting, validation and navigation. The presentation tier should be designed to support multiple devices or interfaces. The decomposition, or middle, tier will support most of the services provided by pattern, including intelligent routing of requests, protocol conversion, and security. In addition, the decomposition tier must also implement the intelligence to break down a single request received from a client into several requests which it then routes to multiple back end applications. This may or may not include some local storage of data to allow users to return to a previous session and complete input before submitting the data to the relevant back end applications.

With this approach, the majority of the product and function specific business logic is still concentrated in the application tier. The presentation tier primarily gathers and validates input and then determines the proper back end application for further processing.

Interaction between the presentation and decomposition tier is synchronous. The interaction between the decomposition tier and the backend application tier can be either synchronous or asynchronous. A synchronous interaction is required when the presentation client expects an immediate response, such as a social security number verification or previously determined eligibility verification.

An asynchronous interaction is appropriate when the presentation client does not expect an immediate response. For example, a user may submit a transaction containing intake information that must then be evaluated by an intake worker prior to a decision. In a case such as this, the decomposition tier could initiate a batch transaction to store the information in the appropriate system and, at the same time, send an e-mail to the intake worker notifying him or her of the pending transaction. Alternatively, the decomposition tier could send a transaction to the intake worker's processing queue using an automated workflow process.

Self-service sites can use specialized information to target certain services that match a customer's demographics. For example, consider a self-service site that collects basic intake information from potential clients. A common eligibility system could then use this intake information to determine services for which a client might be eligible. The system could then prompt the user for additional information as need and then pass an eligibility or enrollment message to the appropriate system or systems.

This pattern allows the work in progress to be stored. This would provide a means to a user to fill in the intake information over time, save it periodically, and return to complete the information prior to submitting a request to the back end application. In other words, such a site would allow users to save their work in progress and complete the application at a later time. Such work in progress is not submitted to the processing systems until the user has completed the entire

application. This approach can be effectively used to store such long running transactions before committing a completed transaction to the back end application.

5.3.5 Implementation Considerations

In many ways, the implementation of the Self-Service infrastructure is similar to the Extended Enterprise infrastructure (Section 5.4) in that it provides a mechanism for external users to access information, systems or processes that may span organizational boundaries. The primary difference is that the external users in this case are typically individuals who are not employees of the organizations involved. In addition, access is achieved through a Web browser or a browser-based mobile device using Wireless Application Protocol (WAP).

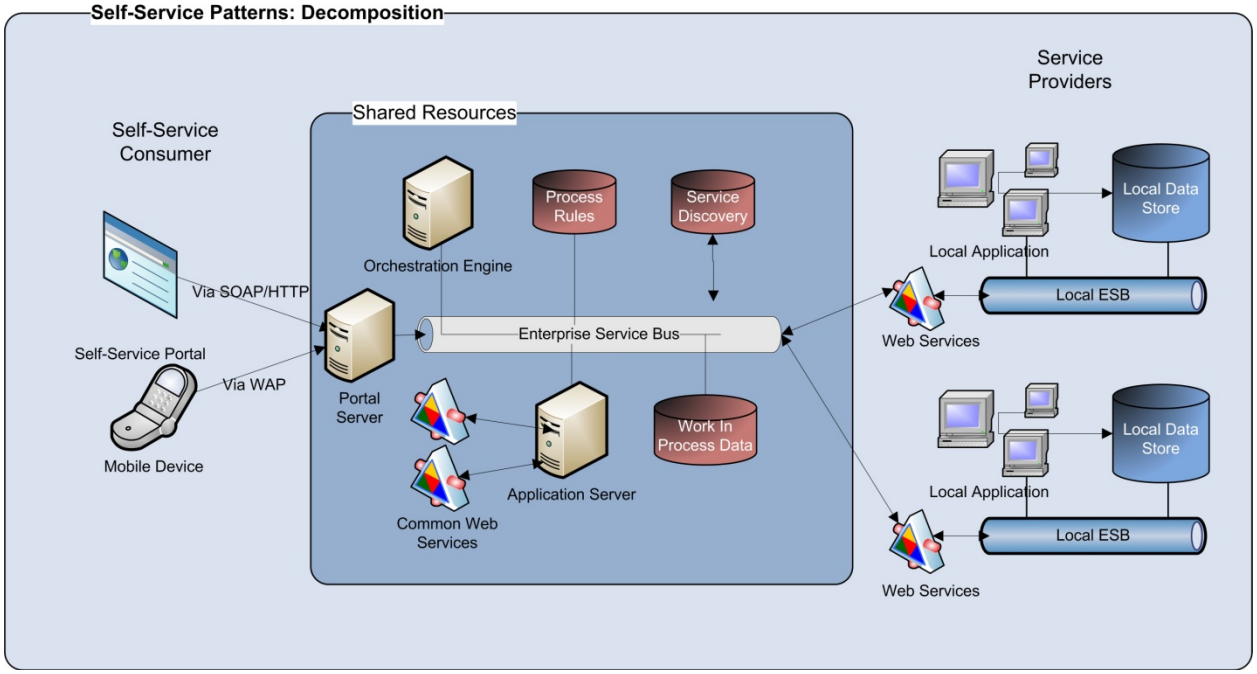


Figure 5–9 Decomposition Pattern Implementation

The architecture presented in Figure 5–9 represents a notional implementation of the Decomposition pattern. The left of the diagram depicts a self-service consumer accessing the environment via a Web portal or mobile device. The center of the diagram represents the shared infrastructure necessary to implement the pattern. Finally, the right of the diagram represents individual, organization-specific applications that are available via Web service calls.

A main feature of the notional architecture presented in Figure 5–9 is a self-service portal and a portal server. A portal server is a type of application server that runs portal software, a type of development tool used to create a portal, or starting point for access to information and processes.

A portal can be as simple as a Web page or series of pages that collect basic demographic and intake information on a potential client and store that information in a local system for subsequent evaluation and processing. A more sophisticated approach may be to use the same Web pages to collect information but to then e-mail the information to an intake or eligibility worker for processing. A yet more sophisticated approach would be to use the same Web pages to collect information but to then, using an orchestration engine, call a series of Web services to verify address information against state motor vehicle records or income against the Internal Revenue Service systems. As verifications are completed, a series of rules to be executed to determine appropriate services and the information could then be routed to the appropriate application systems via messaging or Web service calls to trigger processing in local application systems, notify intake or eligibility workers and to store the relevant data in a local system. The potential client could be notified of his or her status via the portal or later via e-mail.

The self-service portal would likely provide the capability for a human service client to inquire into the status of his or her eligibility or enrollment processing. This would require implementation of the data access and federation patterns to retrieve information from appropriate sources and to present the information to the client via the portal. Similar portals could be constructed to allow human service workers to access all data related to an individual, case, or series of cases.

5.3.6 Infrastructure Components Used by the Self-Service Pattern

- Enterprise Service Bus
- Web Access Management COTS Software
- Portal Servers
- Web Servers
- Application Servers
- Message Oriented Middleware
- Web Services
- Orchestration/BPM Engine
- Rules Engine

5.4 Extended Enterprise

5.4.1 Introduction

The Extended Enterprise pattern addresses the interactions and collaborations between business processes in separate enterprises. This pattern is appropriate to situations in which applications must be connected programmatically and across organizational boundaries. The pattern does not include applications that are accessed directly via a user interface. The Extended Enterprise pattern contains several sub-patterns, discussed below.

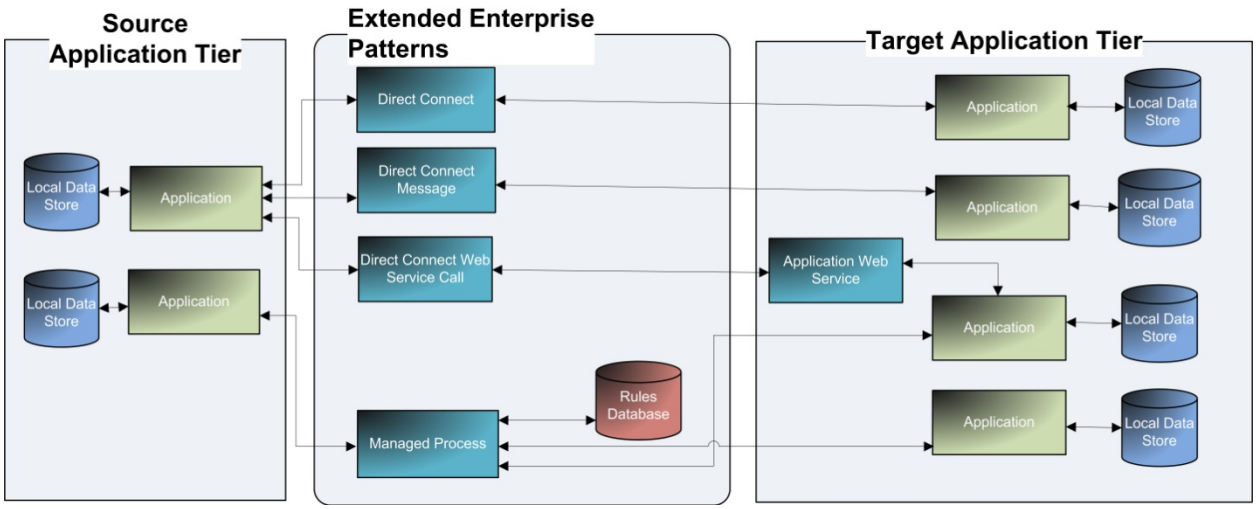


Figure 5–10 Extended Enterprise Patterns

5.4.2 Direct Connection

The Direct Connection application pattern represents the simplest interaction type based on a one-to-one relationship between two systems. It allows a pair of applications to directly communicate with each other across organization boundaries. Interactions between a source and a target application can be rather complex. However, complexity can be addressed by breaking down complex interactions into more elementary interactions. The primary goal of the Direct Connection application pattern is to permit an application to gain direct and real-time access to another application that is outside the organization in order to reduce the latency of business events.

Because an application is directly exposed across organizational boundaries, it must implement or exploit the necessary security features such as authentication, authorization, confidentiality, integrity, and logging for non-repudiation purposes.

The use of this pattern allows the complete integration of applications belonging to different organizations. Source and target applications are decoupled from one another, as are business logic and communication details. Therefore, it is possible to develop different parts of the whole system independently.

However, this pattern does implements a direct connection between the source and target application. Therefore, changes to one application may have unexpected effects on the other application. This is especially true when integrating across organizational boundaries. Any changes to the exposed target application might require changes to many partner applications. Such changes can be both expensive and time-consuming.

5.4.3 Direct Connection via Message

The Direct Connection via Message pattern is similar to the Direct Connection pattern (above) except that with this pattern, the source application or business process does not require a response from the target application within the scope of the interaction. Essentially, this pattern represents a one-way message flow.

Messaging is a general term used to describe the exchange of transaction data between two distinct applications. The main benefit of this type of exchange is that it decouples the source and target applications and hides the technical implementation details of each system from the other. Messages can consist of records, files, alerts, requests and responses, among other things.

In most situations, this pattern will employ message-oriented middleware. This provides organizations with the ability to send and receive messages asynchronously via what is known as a store and forward mechanism. This mechanism and be employed to create a reliable messaging environment in which one application can send a message to another application even though the receiving application may not be available at the time the message is sent.

5.4.4 Direct Connection via Web Service

The Direct Connection via Web Service pattern is similar to the Direct Connection via Messaging pattern except that with this pattern, the source application or business process initiates the interaction via a call to a Web service that has been exposed by the target application. In addition, the source application may require a response from the target application within the scope of the interaction.

5.4.5 Managed Process

The Managed Process pattern applies to situations where a source application starts an interaction that is distributed to multiple target applications across organization boundaries. The Managed Process pattern facilitates the sequential execution of business services hosted by a number of applications that may be hosted by different organizations. In other words, it enables the orchestration of a business process across enterprise boundaries, in response to an interaction initiated by the source application. The primary driver for selecting this pattern is to support the automated coordination of business process flow between partners.

Applications that are based on a service-oriented architecture (SOA), which have well-defined and coarse-grained business services that represent a unit of work, are better suited for this pattern. These business services are composed into an end-to-end process flow.

5.4.6 Implementation Considerations

The Extended Enterprise pattern is intended to provide guidance relevant to creating interoperability and interaction between independent systems across multiple organizations. In all likelihood, such systems will include a variety of technologies, some modern and some legacy. The sub-patterns presented here can be applied to any combination of technologies to achieve integration.

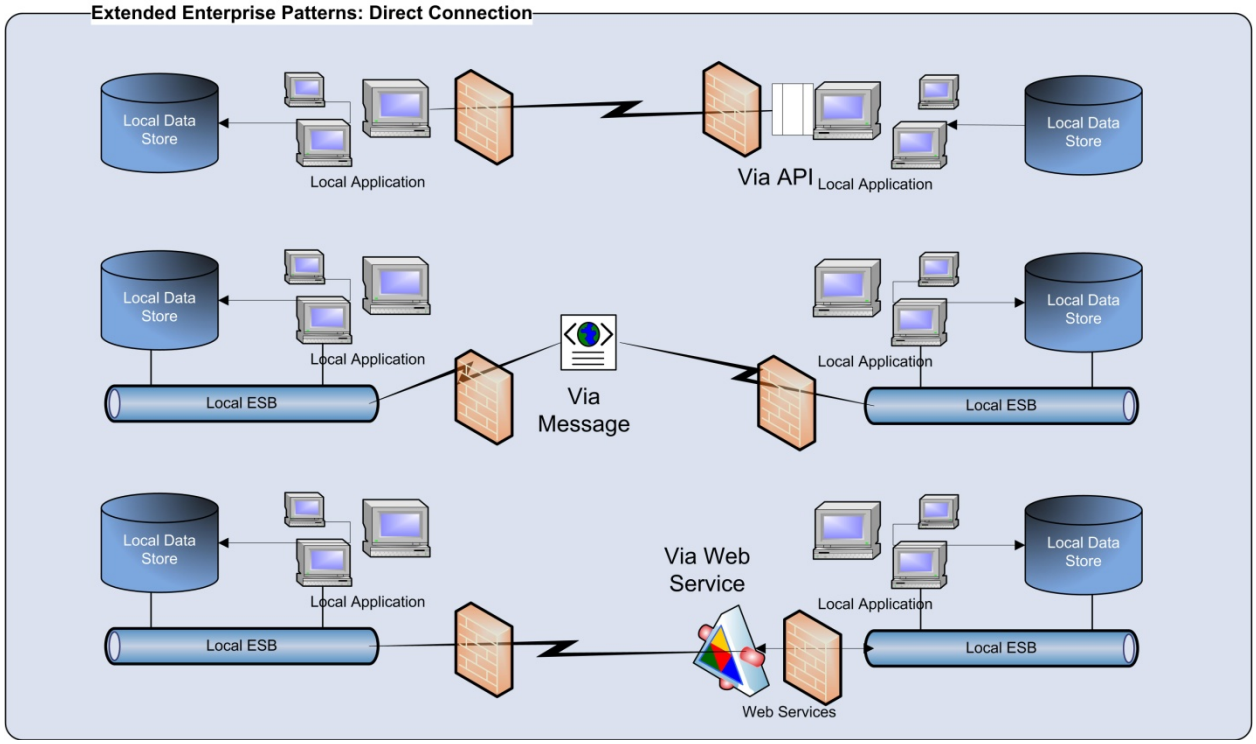


Figure 5–11 Direct Connection Implementations

Each of the Direct Connect patterns can be implemented using technology appropriate to the situation. In the case of a single direct connection between two local applications, the source application may access the target application via a host application programming interface (API). In this case, the target program makes the API available and the source program develops the software to integrate the API into its own logic. While this is a relatively simple means of integration, it has the disadvantage of creating a static point-to-point interface that will require software modifications should either side of the transaction change. Further, if either the communications or the target application is unavailable, the transaction will fail.

The Direct Connect pattern can also be implemented using a message. This will involve the source system formatting the data, generating the message, and transmitting it to a message queue. The message queue will most likely be implemented in the target environment using either message oriented middle ware

(MOM) or the organization's enterprise service bus (ESB). This type of implementation has several advantages. First, the message queue is able to stage the message in the event the target application is unavailable. The middle ware (MOM or ESB) may also provide the capability to transform the incoming message into the format required by the target application. However, this type of implementation still requires a point-to-point connection for each pair of applications exchanging information.

A further variation of the Direct Connect pattern is to implement the exchange of information using a Web service. This is similar to the message implementation in that it isolates the details of the exchange from the source system. However, this type of implementation still requires a point-to-point connection for each pair of applications exchanging information.

The Direct Connection sub-pattern is most appropriate for point-to-point connections involving legacy applications. This pattern can be relatively simple to implement via hard-coded logic in the source program, but in most cases does not achieve any significant integration between systems. The Direct Connection via Message sub-pattern is similar to Direct Connection in that it provides a point-to-point interface. However, this pattern is most often implemented using Message Oriented Middleware (MOM), which provides a loose coupling between the two systems. This approach is most applicable to situations requiring integration with legacy systems. Again, the disadvantage of this sub-pattern is that multiple point-to-point connections may be required to achieve integration among multiple systems.

Most commercial SOA platforms include some type of messaging. Therefore, an SOA platform could be used to support a messaging pattern or a service-oriented pattern. The main difference between MOM and an SOA platform is in the manner in which the source application interacts with the messaging middleware. In other words, with MOM, source applications interact with message queues which provide no information about the target application or service. With an SOA platform, services are described and discoverable, they are based on WSDL technologies, and they use XML formats for message translation and transformation. This provides a means for source applications to discover a needed service. The ESB will the route requests to a service implementation behind the bus which matches the request via the interface.

In the point-to-point integration, independent connections between each pair of systems must be created. For example, if there are six systems to be interconnected, there must be thirty separate connections. Over time, as more and more systems and applications are added to the organization, the number of interconnections to be defined between these systems rises exponentially. The Managed Process sub-pattern is intended to eliminate the point-to-point complexity through the use of a shared infrastructure.

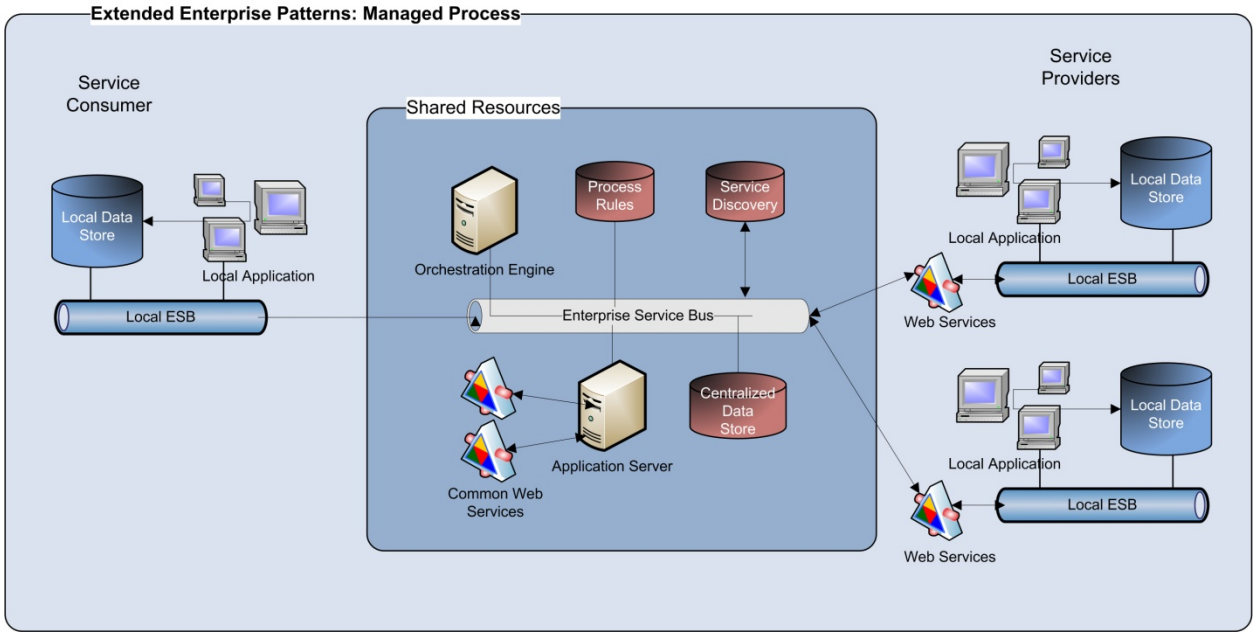


Figure 5–12 Managed Process Implementation

The notional implementation presented here makes use of a number of components within a shared infrastructure. This shared infrastructure will likely contain an enterprise service bus with capabilities to mediate interactions among Web services hosted both within the shared infrastructure and in external organizations. In addition, the ESB will provide business process management (BPM) or orchestration capabilities. The will allow a source organization to establish and execute business processes that make use of common web services hosted in the shared infrastructure as well as web services hosted by provider organizations. Web services will be discoverable via a shared service directory.

Since business processes can span multiple applications and organizational boundaries, the orchestration engine maintains state and tracks sequencing through the process flow. In doing so, it may use a storage repository to store intermediate results. It is also responsible for invoking target applications as necessary through their associated connectors on the ESB. The Orchestration engine can support serial processes in which there is a sequential execution of process steps and parallel processes where process steps or sub-processes can execute concurrently.

The Managed Process sub-pattern increases the flexibility and responsiveness of an organization. It does this by implementing end-to-end process flows across organization boundaries and by externalizing process logic from individual applications. In addition, it provides a foundation for automated support for BPM that enables the monitoring and measurement of the effectiveness of business processes.

5.4.7 Infrastructure Components Used by the Extended Enterprise Pattern

- Enterprise Service Bus
- Web Access Management COTS Software
- Portal Servers
- Web Servers
- Application Servers
- Orchestration Engine
- Service Registry

5.5 Business Intelligence and Analytics

5.5.1 Introduction

Business intelligence systems focus on improving the access and delivery of business information to both information providers and information consumers. This section addresses the architecture patterns relevant to implementing a business intelligence infrastructure to support reporting and analysis of information combined from multiple organizations as well as individual reporting requirements that are shared among organizations.

Business intelligence systems are typically designed to process and analyze large volumes of information using a variety of different tools. A business intelligence system must, therefore, provide scalability to support growing information volumes and be able to support and integrate products from multiple vendors.

A business intelligence system may also provide access to business information through the use an information catalog that documents decision support objects that can be employed by information consumers to answer the main business questions that arise in everyday business operations. These objects may be combined and presented via a user interface called a dashboard. Some systems also provide information consumers with the ability to subscribe to the information they require, and the system delivers it to them at predefined intervals.

To support reporting of information combined from multiple organizations, the Business Intelligence and Analytics pattern presupposes the existence of a common data warehouse to store the required information. The information stored in a data warehouse will be sourced from operational databases most likely via the Population pattern and will be structured in such as way as to facilitate inquiry, reporting and analysis. This will thereby reduce the impact on an organization's operational transaction systems.

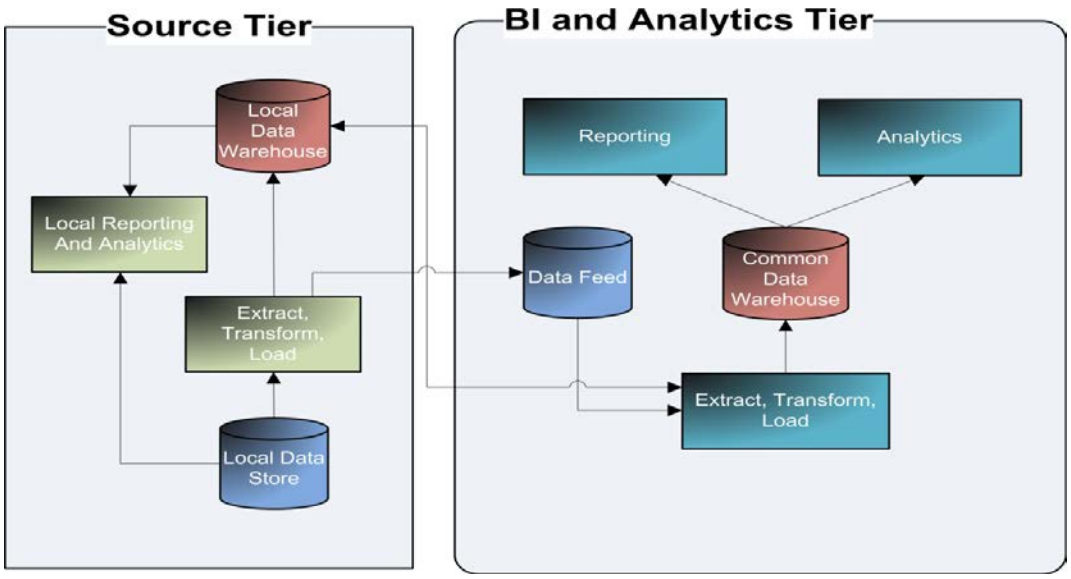


Figure 5–13 Business Analytics Patterns

5.5.2 Extract, Transform, Load

The process of moving data from its location in an operational system involves transforming it into the required format for the target data warehouse, and loading it into the target data warehouse. Even though the three steps may be performed on different systems or in different locations, there are often referred to together as extract, transform and load, or ETL.

Individual organizations will most like assume responsibility for providing data feeds to the central data warehouse. These may be created as a part of the organization's own ETL process or by some other means that will depend highly on the application software and database technology of the organization. For example, many commercial hospital systems provide the capability to generate extracts of HL-7 data that could be used as input to the central warehouse. The format of the data and the communications means will again be up to the individual organization.

The transform process takes the incoming data feed and processes that data according to predetermined criteria. For example, name and address data may be normalized before being loaded. In addition, data may be aggregated or analyzed prior to loading. Finally, the load phase writes the data to the local data warehouse.

5.5.3 Reporting

Business intelligence reports will draw on the data loaded into the common data warehouse and will be presented to the information consumer via a web browser. Reporting will mostly likely consist of predefined reports and answer specific business questions although an online query capability may be employed as well.

In addition, reports may be available via subscription and will most likely be delivered via e-mail.

5.5.4 Analytics

Analytics typically refers to the processing, summarization, and categorization of data necessary to produce actionable insights from that data. With source data available in a data warehouse, apart from operational systems, sophisticated routines can be created using COTS business intelligence and analytics tools. The output of these routines can be presented to users in a way such that additional analysis can be performed in real-time using online analytical processing tools (OLAP).

5.5.5 Implementation Considerations

The intent of this pattern is to provide a common infrastructure to support business intelligence and analytics that draw upon data from multiple organizations or are common across multiple organizations. It is not intended to replace an organization's existing business intelligence and analytics infrastructure or functionality.

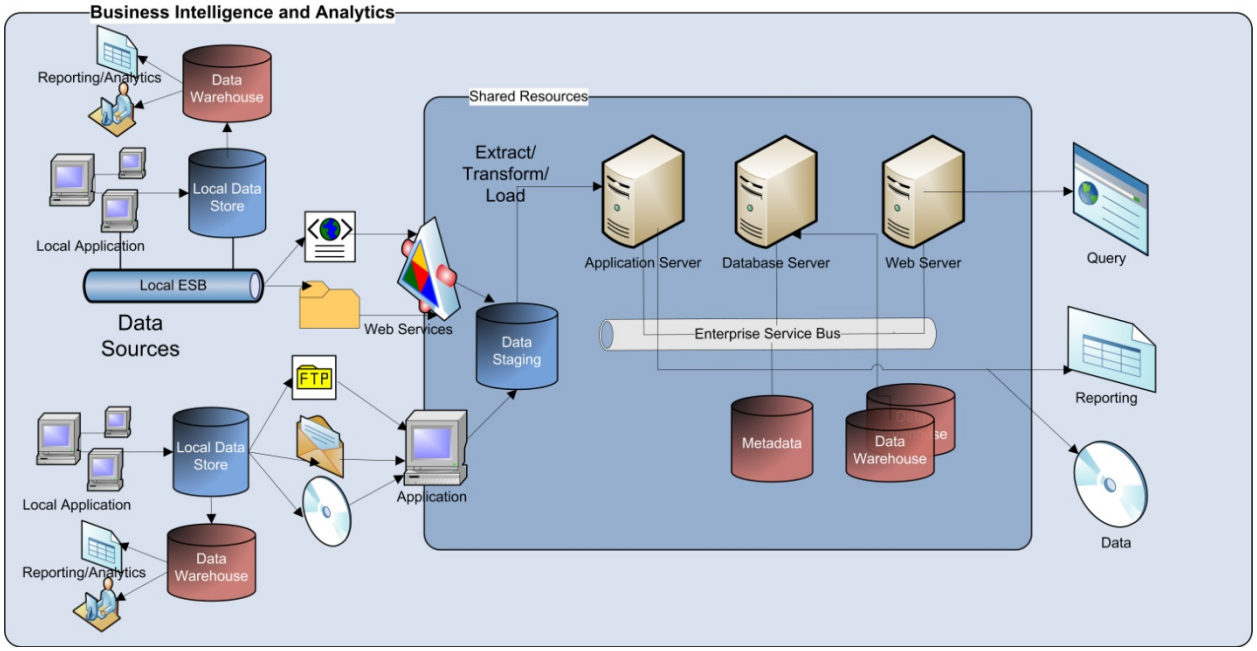


Figure 5–14 Business Intelligence and Analytics Implementation

The primary assumption is that the infrastructure will include some form of shared storage, shown above as a common data warehouse. Data will be provided by source applications or local data warehouses via a variety of means. For example, organizations that are able can invoke common web services to transmit files or XML documents to the shared infrastructure. This is depicted in the notional

infrastructure as a local system invoking a Web service via its own Enterprise Service Bus and transmitting data in this fashion.

Other organizations may provide data via file transfer. This is depicted in the notional infrastructure as an organization providing an extract of its operational data and sending it to the shared infrastructure by some means. This may include use of file transfer protocol (ftp), e-mail or some means of manually sending the data to the shared infrastructure (e.g., FedEx).

All data will be copied initially to a staging are in the common infrastructure. From there data will be extracted, transformed and loaded (ETL) into the common data warehouse. COTS tools are available that provide a facility to map incoming data against its data targets, perform necessary transformations, such as harmonization or aggregation, and load the data into storage.

It is also likely that COTS business intelligence and analytics tools will be hosted in the shared infrastructure. Such tools will be the primary means for display, inquiry and reporting of data from the common data warehouse. Most COTS tools provide the capability to present data via a web browser. In addition, COTS tools typically provide the ability to query and analyze data online via a web browser. Finally, the capability should exist to generate structured extracts or aggregations of the data that can be sent to other organizations for additional processing and reporting. For example, data may be aggregated across a state and extracted for reporting to the Federal government.

5.5.6 Infrastructure Components Used by the Business Intelligence and Analytics Pattern

- Enterprise Service Bus
- Federated Single Sign-on
- Data Warehouse/ETL COTS Software
- Business Intelligence/Analytics COTS Tools
- Web Servers
- Application Servers