VIRGINIA
IT AGENCY

Commonwealth of Virginia

Enterprise Technical Architecture [ETA]

Integration Domain Report

vita.virginia.gov

April 20, 2023

Integration Domain Report Requirements: Version History

Revision	Date	Description
1.0	May 2001	Initial document (Formally known as Middleware Domain Report)
2.0	July 2006	The 2006 *Integration Domain Report* – Updates 2001 *Middleware Domain Report*
3.0	July 2008	The 2008 *Integration Domain Report* – Updates 2006 Report and includes new sections on Instant Messaging (p. 43) and Mashup (p. 47). **NOTE:** Substance updates in the 2008 Report are in "blue" text in the body with "legal black-line" in the left margin next to the text location.
3.1	April 20, 2023	Administrative update for accessibility

Review Process
This requirements document was posted on VITA's Online Review and Comment Application (ORCA). All agencies, stakeholders, and the public were encouraged to provide their comments through ORCA. All comments were evaluated, and individual commenters were notified of action(s) taken.

Technology Applications, Architecture, and Strategy Directorate Review
The domain report was reviewed and approved by: Jerry Simonoff, Director and Chuck Tyger, the Associate Director of VITA's Strategic Management Services Division.

Online Review
The revisions to the domain report were available on ORCA, the Online Review and Comment Application, for a 30-day review and comment period. All the comments were considered, and many resulted in modifications to the final document.

Standards and Agency Exceptions
These standards are incorporated within the COV Enterprise Architecture Standard (EA-225), and the requirements defined within this document are mandatory for Executive Branch agencies. Agencies deviating from these requirements must request an exception for each desired deviation, and receive an approved *Enterprise Architecture Exception* via Archer, prior to developing, procuring, or deploying such technology, or not complying with a requirement specified in this document.

Glossary
As appropriate, terms and definitions used in this document are in the COV ITRM IT Glossary. The COV ITRM IT Glossary is available on the ITRM Policies, Standards, and Guidelines web page at the VITA website: https://www.vita.virginia.gov/it-governance/glossary/cov-itrm-glossary/

Contents

Purpose

The intent of these requirements is to guide the purchase, design, implementation, and on-going operation of COV IT services and utilized technologies. For further information on the perspectives, please reference the most recent version of the Enterprise Technical Architecture (ETA) Requirements document.

For further information on the perspectives identified in this document, please reference the VITA Enterprise Architecture Standard (EA-225)

Authority

- Code of Virginia, §2.2-2007. Powers of the CIO
- Code of Virginia, §2.2-2007.1. Additional duties of the CIO relating to information technology planning and budgeting
- Code of Virginia, §2.2-2009(A). Additional duties of the CIO relating to security of government information
- Code of Virginia, §2.2-2012(A). Additional powers and duties related to the procurement of information technology.

Scope

This standard is applicable to all Executive Branch state agencies (hereinafter collectively referred to as "agencies") that are responsible for the management, development, purchase and use of information technology resources in the Commonwealth of Virginia. This standard does not apply to research projects, research initiatives, or instructional programs at public institutions of higher education.

[Is there a scope associated with this document? Put the information here. State what this document addresses/covers and also state what it does not]

In addition to the requirements below all COV IT technology solutions comply with the standards found on the VITA Policies Standards & Guidelines page.

Executive Summary ETA Integration Domain

This report updates the 2006 Integration Domain Report. The updates include new sections on Instant Messaging and Mashup.

State agencies continue to be faced with the challenge of integrating disparate systems and islands of automation. Often, information needed by knowledge workers is spread across agencies or throughout various departments within agencies. Knowledge workers need to be able to access all the information they need in a transparent and seamless fashion. To accomplish this, programmers must know how to connect information to applications and customers no matter where it resides on the network. Integration technology enables agencies to address these connection needs in a consistent and useful manner. Integration allows organizations to share data between systems that do not communicate easily. Integration is the enabler of application communications in a distributed system and is the tool that improves the overall usability of an environment made up of products from many different vendors on multiple platforms.
In Virginia, the Enterprise Architecture team has divided the architecture into eight domains. To include every aspect of the information technology architecture in one domain or another, domain teams break down architecture into components. The integration domain team identified a number of components to cover as part of the integration domain. In considering these components, the integration domain team discussed whether the components belong in the integration domain or in neighboring domains including the networking and telecommunications, application, security, database, and enterprise systems management domains.

Additionally, the integration domain team has identified eight integration component types: Database Integration, Message Integration, Distributed Transaction Processing Monitors, Application Integration, Enterprise Service Bus (ESB), Service-Oriented Architecture (SOA) Governance, Instant Messaging and Mashup. The various types of integration products overlap in some services provided. For example, all play a role in sending messages (i.e., between applications across the network) and in accessing data. Table 1 provides an overview of many of the services that might be provided by one or more integration products. The service management tools on the left are examples of functions partitioned from applications and databases for provision centrally in the computing environment (between rather than within the application and databases). The communication services on the right are partitioned from the total "protocol stack" that is required for communication between senders and receivers on a network. So, a big picture look at integration technology would emphasize the bringing to the center certain services so that they can be created one time, managed centrally, and used many times by the distributed network applications.

Table 1: Example Integration Tools and Services

Service Management Tools	Communications Services
1. Environment description tools: example tools might include • business rules/workflow definition tools • distributed environment definition tools • object reuse location repository • message type definitions • protocol translation information for the environment • event registry • distributed location information 2. Diagnostic and analysis tools for monitoring transactions: • monitoring metrics • load balancing services • metric reporting/viewing services 3. Scripting tools for configuring each middleware component. 4. Pre-configured message types for metrics and analysis (e.g., directory entries to aid in counting important parts of financial transactions)	• Directory Lookups • Security Services (e.g., encryption) • Translation Services (e.g., decryption) • Database to Database Interconnections • File Transfers • Asynchronous Messaging (one-way) • Synchronous Messaging (two-way) • Application Interfaces • Message Publish and Subscribe Services (e.g., like news services) • Message Store and Forward Services (e.g., like email)

The Integration concept is difficult to understand from an enterprise viewpoint without having some understanding of how the introduction of the client-server environment and distributed environments affect the complexity of computer programming. Integration vendors are trying to address some of these complexities by centralizing certain functions that may have been embedded in the tools of the network architect, the application architect, and the database architect. Appendices A and B provide related history and communication service information in non-technical terms.

All agency business applications distributed over a two- or three-tiered environment or involving network communications between clients and servers require some of the functionality that may be provided in a bundled integration product. In agencies without a bundled integration product, these services are provided through integration tools acquired along with the operating systems. In database products these services are provided as part of shelf-ware, as separate tools and/or through functionality coded into local applications. The acquisition of an ESB product would enable addressing many integration service needs.

Overview

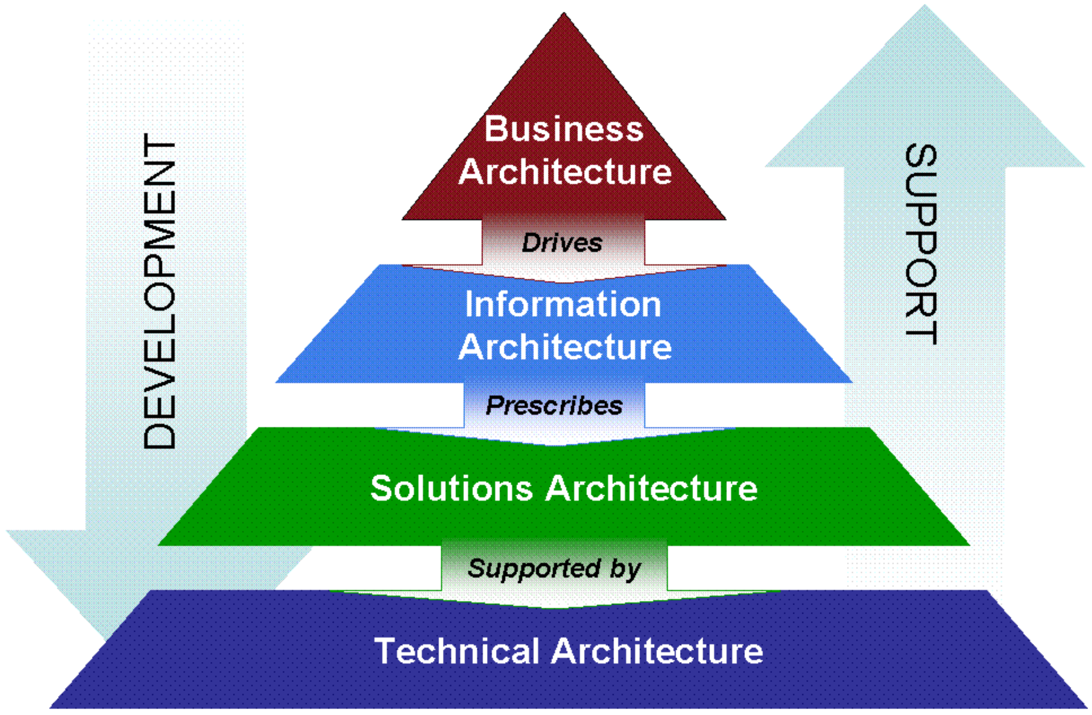
The Commonwealth's Enterprise Architecture (EA) is a strategic asset used to manage and align the Commonwealth's business processes and Information Technology (IT) infrastructure/solutions with the State's overall strategy.

The EA is also a comprehensive framework and repository which defines:

- the models that specify the current ("as-is") and target ("to-be") architecture environments,
- the information necessary to perform the Commonwealth's mission,
- the technologies necessary to perform that mission, and
- the processes necessary for implementing new technologies in response to the Commonwealth's changing business needs.

The EA contains four components as shown in the model in Figure 1.

Figure 1
Commonwealth of Virginia Enterprise Architecture Model



The Business Architecture drives the Information Architecture which prescribes the Solutions Architecture that is supported by the Technical (technology) Architecture.

The Enterprise Technical Architecture (ETA) shown in Figure 2 consists of eight technical domains that provide direction, recommendations and requirements for supporting the Solutions Architecture and for implementing the ETA. The ETA guides the development and support of an organization's information systems and technology infrastructure.

Each of the domains is a critical piece of the overall ETA. The Networking and Telecommunications and Platform Domains address the infrastructure base and provide the foundation for the distributed computing. The Enterprise Systems Management, Database, Applications, and Information Domains address the business functionality and management of the technical architecture. The Integration Domain addresses the interfacing of disparate platforms, systems, databases and applications in a distributed environment. The Security Domain addresses approaches for establishing, maintaining, and enhancing information security across the ETA.

This report addresses the Enterprise Technical Architecture's Integration Domain and includes requirements and recommended practices for Virginia's agencies[1, 2].

This report was developed by the Integration Domain team, which was commissioned to identify domain related requirements and recommendations. Identified requirements and technology product standards from this domain report will be combined with requirements and technology product standards from other technical domain reports into a single ETA Standard for review and acceptance by the Information Technology Investment Board (ITIB).

[1] This report provides hyperlinks to the domain report Glossary in the electronic version. In the electronic and printed versions, the hyperlinks will have the appearance established by the preferences set in the viewing/printing software (e.g., Word) and permitted by the printer. For example, the hyperlinks may be blue and underlined in the screen version and gray and underlined in the printed version.

[2]The Glossary entry for agency is critical to understanding ETA requirements and standards identified in this report and is repeated here. **State agency or agency -** Any agency, institution, board, bureau, commission, council, or instrumentality of state government in the executive branch listed in the appropriation act. ETA requirements/standards identified in this report are applicable to all agencies including the administrative functions (does not include instructional or research functions) of institutions of higher education, unless exempted by language contained in a specific requirement/standard.

Agency Exception Requests

Agencies that desire to deviate from the requirements or the technology component standards specified in this report must request an exception for each desired deviation and receive an approved *Enterprise Technical Architecture Change/Exception Request Form* prior to developing, procuring, or deploying such technology or not complying with a requirement specified in this report. The instructions for completing and submitting an exception request are contained within the ***Commonwealth Enterprise Architecture Policy***.

Integration Domain Scope

The Integration Domain identifies the component technologies that constitute the domain, defines the tools and services that may be provided in integration products, and provides additional decision-relevant information to help agencies and other responsible parties make informed decisions regarding their integration architecture.

Overall Integration Domain Scope

The following technology topics are addressed in the Integration Domain:

- Database Integration
- Message Integration
- Transportation Processing Monitor Integration and Services
- Application Integration Servers and Services
- Enterprise Service Bus (ESB)
- Service-Oriented Architecture (SOA)

Scope of this Report

This report will address all of the technology topics identified above.

Future Integration Domain Initiatives

Future domain topics include the development of an Integration Competency Center to manage several different types of integration products as a single entity.

Domain-wide Principles, Recommended Practices and Requirements

The following principles recommended practices and requirements pertain to all components in all situations and activities related to the ETA Integration Domain. Component specific principles recommended practices and requirements will be discussed in the next section of the report.

Domain-wide Principles

The ETA Integration Domain team identified the following three domain-wide principles:

INT-P-01: The Commonwealth should provide seamless access to data and services.

Rationale:
There is increasing emphasis on the implementation of a single Commonwealth of Virginia portal for citizens to use to obtain data and services from the State. There currently are portals for individual State agencies, which provide a separate means to access data from those agencies.

INT-P-02: Agencies should strive for inter-operability.

Rationale:
There is an increasing need for systems to inter-operate within and across agencies. Integration solutions can help in providing the inter-operability needed within the enterprise.

INT-P-03: Integration solutions should provide flexibility, portability, and cost effectiveness in the implementation of enterprise architecture.

Rationale:
State agencies have limited resources with which to implement enterprise architecture. State agencies must be able to react quickly to change. State agencies must continue to use legacy systems.

Domain-wide Recommended Practices

The ETA Integration Domain team identified the following three domain-wide recommended practices:

INT-RP-01: Agencies should use integration technologies to support logical partitioning and boundaries.

INT-RP-02: Agencies should use technologies that support open interfaces, are persistent, and are non-proprietary whenever possible.

INT-RP-03: Integration software can play an important role in enabling a single sign-on for all applications and services.

Domain-wide Requirements

The ETA Integration Domain team identified the following seven domain-wide requirements:

INT-R-01: Agencies must implement integration applications/ solutions in adherence with all security, confidentiality and privacy policies and applicable statutes.

INT-R-02: The version/release levels of all integration software tools must have vendor or equivalent quality level support available.

INT-R-03: Before acquiring a central integration solution, agencies must map their present integration sources and uses, and shall develop a plan in consultation with the Virginia Information Technologies Agency (VITA) Integration Competency Center (ICC) for migration to the central integration solution.

INT-R-04: Agencies must use integration solutions that are scalable, extensible, and maintainable.

INT-R-05: Agencies must carefully define their interfaces and interface business requirements.

INT-R-06: Integration tools and services must be thoroughly tested. Consideration must be given to the need to maintain a separate environment for testing modifications.

INT-R-07: Before acquiring integration solutions, agencies must contact the VITA ICC (Integration Competency Center) to determine if similar integration solutions exist that could be a shared resource across several agencies. To reach the VITA ICC, contact the VITA Customer Care Center (VCCC) by phone 1-866-637-8482, or 804-786-3932 in Richmond, or by Email: vccc@vita.virginia.gov or go online: http://www.vita.virginia.gov/vccc/incident/vcccincident.cfm.

ETA Integration Domain Technical Topics

Integration Architecture defines the functions that enable communications in a distributed system and the tools that improve the overall usability of architecture made up of products from many different vendors on multiple platforms. Integration tools and products are software that allows organizations to share data between disparate systems that do not communicate easily. Integration Architecture has been described as the software “glue” that ties different applications together.

Database Integration
Database tools and products enable applications to communicate with one or more local or remote databases. It does not transfer calls or objects. For example, database integration does not allow for two-way communication between servers and clients. Servers cannot initiate contact with clients, they can only respond when asked. The discussion of database integration is broken into Directory Services, Metadata, Access Services, and related guidance. Guidance information may direct the reader to other domains when those documents become available.

Directory Services
A directory may be described as a specialized database of lists. Directories serve a wide variety of functions in a computing environment and are used by applications including email, security, and naming services. Directory services are important as tools in the communications process and decisions about directory services are one of the most important foundational decisions an agency can make in planning a distributed architecture and integration strategy. Deciding on a desired external directory strategy (e.g., external to the database system or network management system) before looking at integration products will allow an agency to be more critical of how integration components are integrated, especially in bundled, multi-vendor products. Having a directory strategy is an integral part of promoting interoperability and location transparency and lowering future maintenance costs in a distributed environment. Some directory services can be configured with strong security by attribute so that everyone could see a user email address, for example, but only the user could update a password or see other personal information. Some sample uses of a directory to support government functions are provided below:

- Certificate authority information and public keys for digital signatures
- Single sign-on password information for employees and other authorized individuals
- A statewide citizen-changeable address store that could be accessed by subscribing agencies.
- Encrypted agency PIN numbers for citizen access to services
- Object naming for reuse by programmers.
- Employee address, office phone or email information for updating by employees.

Lightweight Directory Access Protocol or LDAP is based on the X.500 open standard. LDAP specifies the access method and protocol, not the storage structure. LDAP enables extensible access to directories. Using LDAP, directory organization can be configured and extended to add additional categories and attributes. Active Directory Server from Microsoft and Netscape Directory Server are two LDAP compliant directory servers for the NT server networks but LDAP compliant access and storage methods are becoming available on most platforms. Initial implementation of the Microsoft Active Directory Server with Windows 2000 was slowed due in part to changes in the way copies of the directory are replicated and the need for careful planning in organizing the directory structure.

Two additional related directory standards that have been very important to the growth of the Internet are: Domain Naming Service (DNS)--A distributed directory service that may be used on the Internet along with Global Directory Service (GDS) to provide a worldwide hierarchy. This is what enables Internet users to access a Web site by typing a friendly name in the format “www.site-name.com” instead of requiring users to remember complicated series of physical Internet Protocol (IP) addresses with port numbers in the format “127.127.127.127:9999”.

(Note: DNS is criticized for its lack of extensibility and its inflexibility in the area of searching. LDAP has both search and extensibility features).

The Open Group's Distributed Computing Environment (DCE) maintains the LDAP standard[3]. For a guide to additional information on LDAP and related standards work, see http://www.opengroup.org/directory/, the Directory Interoperability Forum.

Following are the two recommended practices for Directory Services:

INT-RP-04: Agencies should consider implementing separate directories for Internal use and external (i.e., beyond the firewall) use.

INT-RP-05: Since 2000 many universities use the EDUCAUSE/Internet2 eduPerson task force effort as a vehicle for coordinating directory standards for faculty and student access. The EDUCAUSE/Internet2 eduPerson task force has the mission of defining an LDAP object class that includes widely used person attributes in higher education. The URL for the Higher Education LDAP Schema work is: http://www.educause.edu/eduperson. VITA should continue to investigate the utility and applicability of this work to government-wide person attributes.

Rationale:
- Security directories facilitate reuse of security and access integration components.
- Reuse of security and access integration components saves time and money.

Following is the requirement for Directory Services:

INT-R-08: Agencies must employ Lightweight Directory Access Protocol (LDAP)-compliant directory services. This lays the groundwork for uniform decentralized lists that can be aggregated centrally for use by the Commonwealth.

Rationale:
- Directory services facilitate reuse of integration components.
- Reuse of integration components saves time and money.
- Directory services promote consistent enterprise solutions.

[1] DCE provides a complete Distributed Computing Environment infrastructure. It provides security services to protect and control access to data, name services that make it easy to find distributed resources, and a highly scalable model for organizing widely scattered users, services, and data. DCE runs on all major computing platforms and is designed to support distributed applications in heterogeneous hardware and software environments. DCE is a key technology in three of today's most important areas of computing: security, the World Wide Web, and distributed objects. (From the Open Group DCE Web site)

Directory Services Standards

Technology Component Standard INT-S-01 provides technology ratings. In general, the technologies listed as strategic are based on open standards.

Table INT-S-01: Directory Services Technology Component Standard
Strategic:
• LDAP, DNS & GDS • Sun JDAP; • MS Active Directory (ADSI)
Emerging:
• None
Transitional/Contained:
• X.500 DAP
Obsolescent/Rejected:
• Novell NDS
Exception History:

Database Metadata Services

Database metadata services are repositories of data about data. The purpose of the metadata repository is to provide a consistent and reliable means of access to data. The repository itself may be stored in a physical location or may be a virtual database, in which metadata is drawn from separate sources. Metadata may include information about how to access specific data, or more detail about it, among a myriad of possibilities. An example of improper metadata comes from NASA: *A Mars spacecraft crash was attributed to using the wrong interpretation of the type of measurement data in a calculation (metric vs. standard).*

Metadata is all the descriptive information that lets us make sense of the data. It tells us things about the data such as: how to interpret it; the business and technical definitions and descriptions; how it may be used; what constraints there are in its use; where it originated; who creates it; who is responsible for it; what business processes it supports; where it is used; its frequency of use; and any other information deemed valuable by the organization. English language or business names for the data, synonyms (other database table or attribute names that refer to the same data), table relationships and the physical database locations should also be included in a comprehensive metadata repository. Policies should be developed for uniform abbreviations, uniform classifications and access types.

The Object Management Group (OMG) and the Metadata Coalition (MDC) are developing a joint metadata model. In the past, metadata tools followed different formats. A subset of these open metadata repository formats and access tools enable designers of systems to find and utilize existing data and services. Attaining reuse of data and services has been an elusive goal of the architecture for years. In the future, both application designers and applications will be able to use integration tools to communicate with metadata repositories and find existing data, services, functions, message format descriptions, etc.

Extensible Markup Language (XML) is a popular method for formatting data message exchange over the Internet. So that agencies and localities use the same set of formats to describe essentially the same data, sharing and reuse of definitions are important. This kind of new information can be placed into an accessible meta-metadata repository to allow developers to share and reuse solutions.

Database Metadata Services Standards
There are no Database Metadata Services requirements or recommended practices at this time.

Technology Component Standard INT-S-02 provides technology ratings. In general, the technologies listed as strategic are based on open standards.

Table INT-S-02: Database Metadata Services Technology Component Standard
Strategic:
• OMG's UML, MOF • MDC's XMI (XML, DTD, Schema) • OIM's exchange format XIF (XML) • Accessible, computer aided metadata documentation (e.g., ERwin modeling tool) and a metadata repository
Emerging:
• Active metadata repository
Transitional/Contained:
• Configurable metadata separate from appellation but proprietary to system.
Obsolescent/Rejected:
• Business rules and meaning hard coded into applications. • Hard copy only documentation of metadata.
Exception History:

Database Access Services
Database access services refer to software applications that are designed to arrange and store data for ease and speed of search and retrieval.

1) ODBC

The term Open Data Base Connectivity (ODBC) is often used to refer in a general way to a group of integration database connectivity drivers and services. The drivers are written to open specifications for accessing data called Structured Query Language (SQL). This includes ODBC, JDBC (for Java), and OLE-DB. Most relational databases support this method of access natively. This method is commonly used for reporting programs to access application tables and for doing lookups in other databases or obtaining data to extract or import into another database. However, since the access is direct and not through the application business rules, ODBC data access should not be used to modify data in different applications by anyone other than the owner of the data. This access method bypasses any security roles and policy maintained through the application interface instead of through database security.

2) Database Gateways / Adapters

Database gateways enable data access and sharing between heterogeneous databases. In order to access non-relational, or legacy databases that do not natively support SQL access, translation database access software needs to be installed on a device with access to the source database. In the past, these gateways enabled the requesting system to utilize the proprietary commands of the host system to access the data, instead of SQL commands. This approach may be advantageous for an application that is being ported to a new platform where the need is to maintain compatibility with the existing application. It is also advantageous when there is a performance penalty for using the open SQL command instead of the proprietary native command. For other generic systems, however, the preferred method is to use the SQL open standard access method.
Adapters are essentially pre-built interfaces for connecting one application to another common business application. In addition to providing access to the data, adapters may also be application program

interfaces, object request protocols, etc. Adapters provide a way to utilize the security and business rules embedded in the application logic. The programmer should be able to extend or modify the adapter if the target application is modified. Even if customization is needed, an adapter can provide a starting point and framework for the programmer.

There are no Database Access Services requirements or recommended practices at this time.

Database Integration Guidelines
ODBC drivers may not support all versions or extended features of the host databases. Agencies should use certification labs or groups to test new database ODBC drivers and new databases especially if any non-standard features are being utilized.

Performance monitoring has multiple benefits. It helps identify what sources of data are getting accessed, and how long it takes users to run their queries on those data sources. Queries that exceed a threshold time limit can be logged for further analysis. This information is important in helping database administrators to reorganize data and create indexes to speed data access. Also, the information may be used to identify priority data for migrating to data warehouses thus decreasing transactions in busy data stores.

There are no Database Integration Guidelines recommended practices at this time.

Database Access Services Standards
Technology Component Standard INT-S-03 provides technology ratings. In general, the technologies listed as strategic are based on open standards.

Table INT-S-03: Database Access Services Technology Component Standard
Strategic:
• DB Adapters or Drivers: ODBC, JDBC, xDBC, OLE-DB (platform specific) • XML point to point contracts (e.g., for Schemas) • ODBC/SQL compliant gateways • XML messaging
Emerging:
• None
Transitional/Contained:
• OLE (replaced) • Screen Scrapers as a mainframe access • Non-ODBC/SQL compliant Gateways • Translators for non-standard SQL, XML, etc.
Obsolescent/Rejected:
• None
Exception History:

Message Interrogation

Message-Oriented Middleware also known as Message Brokers, MOM, and Messaging Broker, provides an interface between applications or application parts, allowing for the transmission of data back and forth intermittently. Messaging integration technology is similar to an e-mail system that transfers messages between people, except that it sends information between applications. MOM is typically asynchronous and peer-to-peer, but most implementations support synchronous message passing as well. In general, a message-oriented middleware has one of two architectures: the hub-and-spoke model or the network-centric bus model, also called the message-bus model. If the destination application is not available because of connection failure or because the application is busy, the integration product stores the data in a message queue until the application becomes available.

Message Formats

In this section, the term “messages” will be used in the broadest sense to encompass transaction-based messages as well as entire file transfers. To many messaging systems, the format of the content of the message doesn’t matter as long as it has the understood envelope/wrapper or an operating system recognizable format. However, the format of the content is very important to the receiving operating system, application, or user. Format translations may be performed by integration products. Also included in this section are messages that are object-oriented. These messages are requests or replies that are issued or received by applications or databases.

When data or even entire databases are transferred between like systems, the entire tables can be copied in their native format. If the systems are dissimilar, data must be converted to a common format understood in both systems. In the past, this format was stated in a standard such as EDI and the encoding was often ASCII, or human readable text that was either fixed-width or delimited with a special character that could be understood by both systems. Sometimes both systems support a common method of formatting or delimiting the export/import file; but, in other cases an intermediary program or integration application is needed to do some transformation.

The ASCII encoding has been used for both file and transaction-based message systems. ASCII is compact, efficient, and compressible. ASCII continues to be used today in newer data access messaging methods such as the Extensible Markup Language (XML). With XML, the standards provide message format and document type definitions (DTD) much like Electronic Data Interchange (EDI) standards provide file formatting for sharing common documents such as Purchase Orders between different systems. Existing EDI methods are still in wide use for financial transactions when modifications or extensions are not needed. To achieve the same goal of standardization today for a wider variety of applications, the trend is to use the XML tagging standards along with contractual arrangements between the sending and receiving parties. With XML, the method is standard, and the content or meaning is flexible.

XML methods can be used to provide structured data formatting for either transaction-based messages or entire files. XML is a subset of Standard Generalized Markup Language (SGML) as is Internet Hypertext Markup Language (HTML). It provides start and end tags in a hierarchical structure to define data for example:

```
<XML>
  <EMPLOYEE>
      <NAME>John C. Smith</NAME>
      <DIVISION>Fiscal</DIVISION>
  </EMPLOYEE>
<XML>
```

Many databases now read XML input, have XML tools and provide XML output (e.g., the requested data from an XML or SQL query may be output in the form of XML tagged data). XML messages can transmit DTDs or XML Schemas in the same message with the data or in a linked file. The DTDs and Schemas define the rules for what may be in the file and what it means. They provide a means for defining the structure, content, and semantics of XML documents in more detail.

One of the benefits of using XML files is that the source system can add a new tag to the message without breaking the message communication.

The human readable ASCII encoding, and format tags are helpful to programmers. Programmers need to tell the application what to look for in each message type and so need to understand what tags to expect in what messages. For this reason, it is important for agencies to develop consistent approaches to tag definitions across applications. Any standardization effort needs to take place between communicating Commonwealth government entities or other communicating entities including other states, industry parties, the federal government or even other countries. However, it is also important to keep in mind that one of the benefits of XML is its flexibility. Standardizations should not get in the way of timely and useful solutions. To aid in standardization that would promote XML Schema reuse and tagging standards, the Commonwealth may wish to create a central metadata repository accessible to all Commonwealth agencies.

Efforts in the area of geographic information systems (GIS) provide an example of the reuse that is possible. GIS standards have affected GIS data transmissions in ways similar to the effect of EDI on financial communications. Standards for GIS metadata and messages have been instrumental in the development of GIS mapping servers that can search for data stored on distributed servers and overlay it on a map in the client browser.

One potential area of weakness for XML is its high overhead (e.g., from tagging). It is verbose and access to the data is slow due to parsing and text conversion. For moderate sized messages the automatic compression of HTTP 1.1 (the core protocol of the Internet) or standalone compression tools can improve the transmission efficiency.

There are no Message Formats requirements or recommended practices at this time.

Message Formats Standards
Technology Component Standard INT-S-04 provides technology ratings. In general, those technologies listed as strategic are based on open standards.

Table INT-S-04: Message Formats Technology Component Standard
Strategic:
• XML and CSS (presentation style configurable by administrator for device types) • 7 bit ASCII; 8 bit ASCII; EBCDIC (translation)
Emerging:
• None
Transitional/Contained:
• None
Obsolescent/Rejected:
• None
Exception History:

Message Transfers
Message transfers refer to software applications that are designed to provide for correct and reliable end to end data transport between communication partners.

1) File Transfers

The most common form of message sending and receiving is a request for the transfer of a file. File transfer requests are generally accomplished through the use of operating system “file copy” commands. Integration compression programs are sometimes used to shrink the size of the message copied.

2) Terminal Emulation

Mainframes have difficulty communicating to the Web natively because their communications protocols were developed and fleshed out before LANs and WANs became ubiquitous. Visual user interfaces other than terminals typically do not exist for mainframes.

Screen scrapers are one integration method of converting terminal output to browser-viewable output. Thin-client integration technology is similar but involves running the terminal application on a remote server and transferring the pixels and pixel changes to the end-user’s browser. XML conversion of the output is a third approach. Hostbridge, for example, uses a middle-tier application to invoke a CICS transaction using Internet protocols and provides output as an XML document, instead of a mainframe terminal screen. With mainframes, integration products provide work-arounds because integration technology is a distributed system concept and mainframe communications methods do not blend easily with distributed network communications.

3) Translation Services

Some integration products provide platform-related translation services to ensure that the message is delivered in a language or form that can be understood by the receiving application. Common examples include 7-bit to 8-bit ASCII (American Standard Code for Information Interchange) or ASCII to EBCDIC (binary coded data). Translation service may also include translation from one proprietary implementation of XML to another.

4) File Transfer Protocols

File Transfer Protocols (FTPs) are used to transport whole files over the Internet. This protocol allows files to be transferred between dissimilar systems, but it is not a secure protocol. Some integration tools may enable the scheduling of file transfers for afterhours processing, add automated archiving, error recovery, and summary logging. FTP (IETF RFC 959[4]) should not be used if data security is an issue as passwords are transmitted in clear text. A security extension to FTP is provided in IETF RFC 2228[5].

5) HyperText Transfer Protocol

HTTP is short for HyperText Transfer Protocol, the underlying protocol used by the World Wide Web. HTTP defines how messages are formatted and transmitted, and what actions Web servers and browsers should take in response to various commands. For example, when you enter a URL in your browser, this sends an HTTP command to the Web server directing it to fetch and transmit the requested Web page.

[4] http://ietf.org/rfc/rfc959.txt?number=959 provides the original FTP specifications from the Internet Engineering Task Force (1985).
[5] http://ietf.org/rfc/rfc2228.txt?number=2228 provides the security extension RTF from 1997.

XML is transmitted using HTTP. With XML, the presentation of the data can be separated from the screen format. A programmer may use XML-aware application tools including parsers, extensible style language (XSL) and cascading style sheets (CSS) to create more than one presentation of the data. For example, PDAs and cell phones require presentation styles that are quite different from what would be appropriate for a computer monitor. Yet, because of CSS, the same XML data could be sent to PDAs and computers and a different interface would be shown to each equipment user. Style sheet aware browsers can enable multiple viewing options for the Internet client without requiring the server to resend the data. Browser support for XML style sheets is fairly recent.
HTTP is called a stateless protocol because each command is executed independently, without any knowledge of the commands that came before it. This is the main reason that it is difficult to implement Web sites that react intelligently to user input. This shortcoming of HTTP is addressed in a number of technologies, including ActiveX, Java, JavaScript and Cookies.

There are no Message Transfers requirements or recommended practices at this time.

Message Transfers Standards
Technology Component Standard INT-S-05 provides technology ratings. In general, those technologies listed as strategic are based on open standards.

Table INT-S-05: Message Transfers **Technology Component Standard**
Strategic:
• File and Data Requests/Replies   o FTP   o XML file transfer • Presentation and Translation Services for Security   o Encryption/Decryption Services (A wide variety of encryption algorithms are strategic depending on security needs) e.g., Symmetric Encryption, DES, Triple DES, RC2, RC4 • Terminal Emulation   o APPC LU6.2
Emerging:
• None
Transitional/Contained:
• Presentation and Translation Services for Security   o Proprietary style layout separates from application • Terminal Emulation   o SNA/SDLC (OSI level 2)
Obsolescent/Rejected:
• FTP whenever security required
Exception History:

Message-Oriented Middleware
Message Oriented Middleware (MOM) refers to a special set of software applications that are used to manage the message distribution, receipt confirmation, and error handling processes. The messages are distributed network communications between applications. Message tracking on a distributed network is like international package delivery tracking. For example, package shippers today are able to know exactly where their packages are at each step of a physical delivery sequence, which packages are lost, and which are damaged and require resending. To have this level of information detail about application communications, the programmer relies on integration tools. To address such complex message sending, tracking and receipt recording requirements, MOM vendors provide several different message services.

1) Store and Forward Integration Tools and Services

Store and forward services allow the producer of a message (e.g., one application) to identify the recipient location, but if the recipient is not available, the message is held in a central queue or store. The transmission of the message to the recipient may be delayed until multiple messages have accumulated for the recipient, until a time interval has elapsed, or until an event has occurred (e.g., the recipient has become available). This service can enable a busy system to delay processing messages that are not time sensitive to off hours.

Email applications employ a store and forward method of messaging. Email illustrates both the benefits to be gained from this form of messaging and the potential problems that are inherent in messaging systems. Before the advent of Internet mail standards, email systems in the Commonwealth were often incompatible. Integration tools were sometimes used to translate between the systems or to provide additional functionality on top of the email. Now state and local agencies are well on their way towards standardization in this area. The email applications used support receiving standard messages from any system. One of the weaknesses of email messages is that they are one way. A response is procedurally required when appropriate but not mandatory. Complex systems have been built based on email messaging, but they require both sides to trust that the other will respond as appropriate and in a timely manner. Extensions to email systems that are not uniformly implemented include options for notification of message receipt, notification of message being opened, or recall of the message (optional applications that can be invoked by the sending application).

2) Publish/Subscribe Integration Services

Publish and Subscribe Integration services allow the producers of messages to publish the messages to a central location. This central location then uses distribution lists formed by subscription of the recipients.

3) Event Registry Services

For any messaging system, a variety of events may occur between receipt and delivery for one-way messages (asynchronous) or between request and reply for two-way messages (synchronous). To manage and use these events to control messaging, the messaging system needs a way to identify the events and exceptions. Examples of events are “server is not available”, “time limit has passed”, or “now is 8:00 AM”. Example responses are “check for server availability” or “notify sending application”. Once a message leaves its application and before it gets to its destination, the use of events and actions falls under the auspices of integration technology. Messaging integration products enable the establishment of an event registry and event monitoring services. The event registry can be used to identify thresholds and corresponding actions (e.g., recovery steps). In the case of transaction processing monitor software, the events and responses may be more complex, involving transaction statistics and invocation of special functions. When the transaction process runs cleanly performance statistics would be gathered but nothing more. But if the process were to fail, the failure event may invoke a paging function to notify the operator on call or if no answer, the backup or supervisor.

4) Intelligent Routing Services

Intelligent Routing Integration Services ensure the message gets delivered to the appropriate recipients in the correct sequence. The routing service can be configured to handle the exceptions with instructions to forward the message on to another service or to return it if the intended recipient is not available. It could also be configured to transfer a message in sequence from one recipient to another.

There are no Message-Oriented Middleware requirements, recommended practices, or technology component standards at this time.

Messaging Integration Standards
The recommended protocols may apply to mail messaging and/or other application-to-application messaging. Mail programs should support use of MIME (Multipurpose Internet Mail Extensions), be SMTP/ESMTP enabled (Simple Mail Transfer Protocol/Extended Simple Mail Transfer Protocol) and provide proxy through IMAP4/POP3 servers (Internet Message Access Protocol 4/Point of Presence 3). Mail programs that interface with Windows clients use Microsoft's MAPI (Messaging Application Programming Interface) interface. Integration protocols used by mail applications and/or other applications include LDAP, DNS (Domain Name System), SSL (Secure Sockets Layer), and additional security protocols. Mail uses security protocols for digital signatures and related encryption. Encryption may also be used in the transmission of sensitive data over LANs including transmissions of passwords, social security numbers, credit card numbers, etc.

MIME stands for Multipurpose Internet Mail Extensions. MIME extensions are important to the recipient mail application because they are used to identify the helper application that enables viewing of the attached file. Several examples of common MIME types are provided below:

Table 2: Multipurpose Internet Mail Extensions (MIME)

MIME Examples[6]		
MIME Type	**Extension**	**Explanation**
text/html	html, htm	Hyper-Text Tag Markup Language
text/plain	txt, text	Plain ASCII text
image/gif	gif	Graphic interchange format
image/jpeg	jpeg, jpg	Joint Photographic Experts Group
image/tiff	tif, tiff	Tagged Image File Format
application/pdf	pdf	Portable Document Format
application/msword	doc	Microsoft Word format
application/zip	zip	compression format

Following are the two requirements for Message Integration Standards:

INT-R-09: Agency email messaging must be SMTP and MIME compatible. Local governments are encouraged to follow this standard as well.

INT-R-10: The Message Transfer Agent (MTA) in email applications should be LDAP enabled.

Rationale:

- Industry standards such as SMTP and MIME email protocols replace the need for proprietary “gateways” and as a result Networks became faster, operating systems became more reliable, and email systems became “enterprise ready.”
- LDAP compliant directories facilitate email authentication, access control, directory lookups, and distribution lists.

[6] The assignment of MIME types is controlled by the Internet Assigned Numbers Authority (IANA). MIME sub-types that begin with 'x-' are not registered with IANA.

Message Integration Standards
Technology Component Standard INT-S-06 provides technology ratings. In general, those technologies listed as strategic are based on open standards.

Table INT-S-06: Message Integration Technology Component Standard
Strategic:
• IMAP • MAPI • SMTP/MIME • XSL (presentation style and content configurable by user)
Emerging:
Transitional/Contained:
• X.400 • POP3 • VIM • CMC
Obsolescent/Rejected:
• Non-Internet compatible email
Exception History:

Message Integration Guidelines
1) Design Interfaces to be Message based

Interfacing a new application to an existing one is easier when applications are designed for messaging. In an application designed for messaging, the interface mechanism is already defined. The interface process can be automated without modifying the target application.

2) Move towards Asynchronous Messaging

Messages are essentially requests or responses. From the viewpoint of a programmer interactions between applications and/or databases are of four types:

1. Synchronous messages (requester suspends processing);
2. Deferred synchronous messages (e.g., requester uses polling but continues processing);
3. One-way messages (no reply expected); and
4. Asynchronous messages (requestor continues processing and replier interrupts).

Asynchronous messaging provides greater efficiencies. The systems do not need to have synchronized startups or shutdown procedures. Messages can be stored and processed once the system becomes available. There may be procedures that cannot be completed without a response from the recipient, but the source system can gracefully give a reason and a resolution to the client.

3) Messages may have to be delivered to a variety of platforms.

Although integration tools can provide some translation services to assist with application-to-application communications, there are also several protocols and methods that are designed to facilitate communications across platforms. ASCII, EDI, XML, and to a lesser extent MIME types, define open message format standards that can be supported/or translated on most platforms.

Following is the recommended practice for Message Integration Guidelines:

INT-RP-06: Agencies should use asynchronous messaging to provide opportunities for making efficient use of parallel processing capabilities in the network environment.

Rationale:

- Asynchronous communication offers more flexibility than synchronous communication. Asynchronous messaging allows the downstream process to decide on an appropriate processing strategy to optimize its throughput and/or respond to different priority requests. This promotes increased performance, flexibility and scalability of the application.

4) Use security where applicable including appropriate authentication, authorization, message encryption/decryption, secure transport protocols, and secure storage.

Security services employed by an application are part of a larger set of agency-determined security policies, strategies, and tool. The security architecture document addresses this big picture. Integration technology plays a role in deploying security as messages traverse the network and find their destination; however, only a small part of security is provided through integration technology. Integration technology provides only selected access, communication and authorization tools and protocols for networked applications. Actual security attained by using these tools is dependent on many factors beyond the scope of this paper. As part of a larger plan for building a secure application, the planner may be interested in whether the integration technology provides a particular level of encryption determined by key size, for example.

No Message Integration Guidelines Technology Component Standards have been developed at this time.

Message Integration Limitations
MOM is typically implemented as a proprietary product, which means MOM implementations are nominally incompatible with other MOM implementations. Using a single implementation of a MOM in a system will most likely result in a dependence on the MOM vendor for maintenance support and future enhancements.

Transaction Processing Monitor Integration and Services

Distributed transaction processing ensures transaction integrity for transactions that involve databases. Transaction processing is the independent execution of a set of operations on data in a relational database, which treats that set of actions as a single event. If any part of the transaction process fails, the entire transaction fails and all participating resources are rolled back to their previous state. Transaction processing monitors and some web services software are critical to n-tier computing, because they facilitate writing of the programs necessary to track transactions across multiple platforms. In the N-tier world, the application layer functions between the presentation layer on the PC and the data layer on the mainframe, UNIX, or NT system. Historically some of the following services have been included in transaction processing monitor integration: two-phase commits, failure/recovery, synchronization, scheduling, repeat attempts, business-rule-based transaction workflow services, message queuing resource managers, and load balancing. These services are described briefly below. Perhaps the most significant feature of the TP monitor is its ability to funnel database requests. General guidance information follows the descriptions.

Two-Phase Commits
A transaction is a unit of work that either commits (executes to completion and its results persist) or aborts (fails and its results are undone). Commit processing means that a transaction that involves multiple tables or systems is managed so either all of the data is modified or none.

Failure/Recovery
Transaction monitors maintain a sequenced log of all transactions that happened across an integrated set of tables. Transactions logs are used to provide the option of rolling back changes made to the tables to a predefined state by reversing the actions. The logs can also be used to redo the actions after backup files have been restored from a set point in time, or to redo the actions if an incorrect calculation was being made.

In the event of a failure, the failed process is either restarted or switched over to a process on another node.

Synchronization
Transaction monitors can be used to synchronize two different systems. A log is kept of all transactions, which can then be applied to another system so that an identical set of inputs may be provided to both systems.

Scheduling
The log of transactions can be stored in the message queue until a predetermined event, time, or request occurs to initiate the transfer of those transactions to the other system.
Low priority transactions or non-time sensitive transactions can be delayed for afterhours processing.

Repeat Attempts
Transaction monitors can manage multiple repeat attempts to update another system or table, thus, offloading that monitoring responsibility from the requesting system so it can go on to other requests. The transaction-processing monitor may ensure that required processes happen or that the appropriate recovery is initiated.

Message Queue Management
Message queues are an important feature of transaction processing monitors. Queues can be established to focus on either availability or reliability. When queues are in memory, they have greater availability and speedier response times. When disk storage is used, this provides greater reliability at the expense of availability. Some integration message queues provide both disk and memory queues.

Business-Rule-Based Transaction Workflow Services
Transaction processing monitors can monitor transaction flows from multiple distributed systems so that business rules can be applied at a central point and so that intelligent routing of certain transactions to other systems or persons may take place. To address customer service and other priorities, business owners may assign priority processing or special handling to selected transactions. Transaction processing monitors allow the business owner to change such rules centrally in the monitor without changing the core application.

Load Balancing Services
Load balancing and thread management services are important because transaction processing monitors need to process many transactions on many different systems in a very short time period. The monitor can change traffic patterns and processing parameters or increase the pool of processors. This enables the monitor to dynamically adjust to the workload.

Transaction Processing Integration Guidelines
Transaction processing monitors on a mainframe may not provide distributed transaction processing in the same sense as does an integration transaction processing monitor in a distributed network environment. However, because today’s integration transaction processing monitors were modeled on mainframe monitors, and because mainframe monitors are still widely used, mainframe monitors are listed here as strategic.

Transaction processing monitors should be used only when transactional integrity is required. Examples of systems that require transactional integrity include integrated financial systems and other core, distributed business systems. Some of the other management capabilities such as scheduling, load balancing, repeat attempts, business rule workflow or logging with recovery can be found in message-oriented middleware and database management integration tools.

Functions that typically were included for support of distributed transaction processing (including distributed database storage/management), are now included in global architectures for support of database, database access or application tiers. Some of the structures are service-oriented architecture and the variety of 'web services' specifications and standards.

Accordingly, more detail about transaction distribution can be found in the appropriate areas of those topics in this document.

There are no Transaction Processing Monitor Integration and Services requirements at this time. Following is the recommended practice for Transaction Processing Monitor Integration and Services:

INT-RP-07: The Commonwealth and its agencies should carefully investigate the success other agencies and states have had in the deployment of Transaction Processing Monitor Integration and Services products before considering a separate product acquisition.

Rationale:

- While transaction processing monitors offer significant functionality in addition to transaction management, using transaction processing monitors when other integration services suffice may cause unnecessary overhead and may result in performance problems.
- When distributed transactional integrity is needed, use a TP monitor rather than the transaction management capability of a database management system.

Transaction Processing Monitor Integration and Services Standards
Technology Component Standard INT-S-07 provides strategic open protocols and examples of mainframe programs used to define the typical work performed by transaction processing monitors. In genera general, those technologies listed as strategic are based on open standards.

Table INT-S-07: Transaction Process Monitor Integration and Services Technology Component Standard
Strategic:
• SOAP • WSDL • HTTP M-POST
Emerging:
• None
Transitional/Contained:
• X/Open: XA interface (X/Open is the standard, XA is the interface) • STDL (structured transaction definition language) • DTP (distributed transaction processing) • CPI-C (common program interface for communications) • CORBA • DCOM
Obsolescent/Rejected:
• None
Exception History:
Historical Note: Two TP monitors were widely used in the mainframe world and then later transitioned to the client-server world. These were CICS (customer information control system) and ACMS (automated code management system).

Application Integration Servers and Services

Application integration provides interfaces to a wide variety of applications. Application integration might be a service that enables running a legacy system through a thin-client browser or a service that enables the execution of multiple application functions from an …

Methods to Integrate Applications

Methods to integrate applications tend to be aligned with particular application development languages (e.g., Visual Basic, C, C++, Java etc.). The methods are often sets of standards developed by particular industry groups. However, even when two vendors deploy implementations using the same set of industry standards (e.g., .NET), their implementations may have differences and may not interoperate. This result may be due to application tool designers extending the standards or to the standards not being sufficiently specific.

The different industry protocol sets are not designed to interoperate with one another. This is not an issue as long as applications are designed to interoperate only within their intended sphere. When developers try to extend beyond the intended sphere to other applications that implemented a different industry standard, either an integration gateway must be used to provide protocol translation, or the developer must employ both sets of standards in the applications that need to communicate. Remote procedure call differences provide an example. An ONC+ RPC server application cannot interoperate with a DCE RPC client application unless the server and the client have both interfaces or an integration gateway is employed. In general, it would be advisable for Virginia agencies to use only one RPC method for within-agency distributed functions.

1) Application Programming Interfaces (API)

Many common business applications have a defined interface language to allow programmers to customize or extend the tool. In application-to-application communications, the programmer may use an application provided API or use integration technology provided API to which the programmer adds the required arguments. To the extent that application or database interfaces are open rather than proprietary, future application maintenance will be lessened. APIs often require customizations to the calling program, use of an appropriate application generation tool and supplied library, or use of a target application software development kit (SDK).

2) Remote Procedure Calls (RPC)

An RPC is a function call issued by the requesting application to run a procedure in a different address space. RPC code is compiled into programs at both ends of a communication. RPCs may require the suspension of processing by the sender until a response is obtained. Implementations today support distributed object components interoperating as a unified whole. The distributed objects may be on different computers across a network, and yet to the application, they all appear as if they were local.

There are competing RPC protocols embraced by major industry providers that do not interoperate. They are Sun's ONC+ RPC and the Distributed Computing Architecture's RPC (DCE RPC). Initially, Remote Procedure Calls were limited to the network protocols for which they were developed which restricted cross platform compatibility. SOAP over HTTP is a widely adopted specification for interoperability in heterogeneous environments.

3) Object Interfaces

Object interfaces simplify addressing a remote call. Object interfaces are part of end-to-end object architecture models. Component Object Model (COM, DCOM), .NET Remoting, and Remote Method Invocation (RMI) are examples of object models from Microsoft, and Sun Microsystems, respectively. Object architectures provide the interface definition language (IDL) and blueprints used to define object interfaces (interface stubs for objects). A blueprint provides guidance for the development of the interface with a particular language binding (dynamic binding).

4) Object Request Brokers (ORB)

Object Request Brokers act as a software bus for objects to intercommunicate. They provide an object location and access service. Microsoft and Sun Microsystems provide this service as part of their object architecture. Applications can request or dynamically invoke objects regardless of location and through metadata services defined as part of the end-to-end architecture.

5) Web Services

Web Services are another means of communicating and passing data between different applications. Web Services are not tied to any one software platform, but allow methods to be called across different platforms, operating systems and firewalls. The key point here is that the separate systems can be coded in any language which supports web services. This list of languages is increasing at a rapid pace, and includes industry leaders such as .NET, Java, PHP, Perl, C, C++, as well as a variety of other languages.

Web Services use a communication protocol named SOAP to pass data between the client and the server. Simple object access protocol (or SOAP) is a remote procedure call protocol that works over the Internet. A Web Service message is an HTTP M-POST request, in which the body of the request is an XML document. The procedure executes on the server, which then replies with message content formatted in XML. The WSDL (or Web Services Description Language) describes the format and content of the message and publishes exactly what is needed to communicate to any given web service method.

Web Services eliminate the interoperability issues of existing solutions, such as RMI and DCOM, by leveraging open internet standards. Also, since Web Services use HTTP through port 80, it gets around firewall security in ways that DCOM and RMI cannot.

Application Integration Management Services

In addition to utilizing the Object interfaces to applications, many Application Integration server vendors include other management functions. These may include conversion services, legacy wrappers, and data integration services (access to a set of data such as employee data from several remote sites).

1) Legacy Wrapper Services

Legacy wrappers allow connection to common business systems to allow interfacing with minimal intrusion to the external system. Some package existing legacy code as an object that can be called from another program.

2) Conversion Services

Conversion services provide a wizard or engineering support for modifying a target system to incorporate new object functionality so that the calling program can integrate with it.

[1] Security concerns are from http://www.vnunet.com/News/1103805.

3) Data Mapping and Transformation Services

Data mapping and transformation services can handle the data mapping or transformations on the fly that are necessary when the format of the data is not compatible or complete. Other types of integration technologies also provide these services.

4) Event Posting Services

Event posting services allow monitoring the status of the interfaced functions and calls to external applications, logging errors, or tracking milestone events. Some ability to correct and reprocess a call may be provided separately or through an integrated Transaction Processing Monitor.

5) Process Trigger Services

Process triggers may be linked to calls to objects to cause variable processing based on the response from the called object. To the extent that the objects and calls have been defined as steps in business functions, business users may be able to rearrange the steps without the assistance of IT staff.

6) Automated Workflow Services

- A workflow engine is the component in automated workflow services that knows all the procedures, steps in a procedure, and rules for each step. The workflow engine determines whether the process is ready to move to the next step. In general, workflow management focuses on processes rather than documents. A number of vendors make workflow automation products that allow an organization to create a workflow model and components such as online forms and then to use this product to manage and enforce the consistent handling of work. Examples include Microsoft BizTalk 2006, Handysoft BizFlow, NMS Imaging E-Flow, Open Text LiveLink, and Oracle Workflow/9i+.

Application Integration Examples

1) Common Address Change Example:

Application Integration Servers that function as a software hub can fuel the imagination of any citizen or manager. This method can be used to write a function such as an address change function one time, package it as a service and call it from every application that needs to change an address. Or better yet, a developer could have this one service automatically call every application that stores a particular user's address. It would also be possible to call systems that are not connected physically by using the Internet.

Is the above example achievable? In today's technical world, it is very possible using Web Services. These calls can be made over the Internet and can support a host of runtime environments used to write Web Services. Security can be designed to ensure address changes are being requested by a valid caller, and that requests to the end data stores are valid as well. Many older applications, however, were not written for distributed networks and have no exposed Web Services interface.

Given the hundreds of programs that store a name and address in the Commonwealth, enabling such Commonwealth-wide integration for making address changes would be a monumental task. To integrate to these applications across the Commonwealth, a variety of changes might be required. For example, the developer might have to modify applications to add appropriate interfaces; replicate application logic in Web servers; or modify database integration used to manage updates to databases.

Other applications may not be available around the clock; they may be designed for operation only during office hours or may require batch processing of address changes. Such applications may need message queue services so information could be stored and processed when the system is available. The application may not require the services of a transaction processing monitor. Updating the address in some systems and not in others would be an option preferable to not providing any central updates. Updating at different times may be acceptable for many systems. The queue and error logs on the application integration server would allow address changes that could not be made initially to be updated accurately in a locally determined and timely manner. While we have the technical ability to tackle a problem such as this, the widespread nature of the request and technical diversity of the end applications pose a larger task than anticipated.

2) New Enterprise System Example

Some managers have approached the problem of the many different existing programs by replacing the many systems with one comprehensive system. This approach does not eliminate the need for system integration. Even a comprehensive application may be distributed across the physical architecture or may have client and server functionality, thus requiring services provided by integration technology. Applications that access objects need object location and invocation services provided by integration technology. Application integration services may be needed to integrate parts of an application. Database integration may be required to do the initial data transformation and loading for the new application.

Often, when implementing a turn-key system, the business may not replace some specialized functions available previously or an external entity may require specialized or ad hoc reports. To meet these needs, some kind of database, messaging or transaction processing integration will be required; however, some of the older major systems do not support thin client browser access, or do not support all functions over the Web. The extensive scope and timeline for the comprehensive approach can make them costly and time-consuming projects to implement, increasing the risk of failure. On the other hand, well designed replacements with new object-oriented interfaces could be strategic for building and extending the future architecture.

3) Digital Signature Example

Security is a real concern when extending the application outside of the firewall. The COTS Digital Signatures workgroup implemented several pilot projects. Most applications do not natively support the public key infrastructure (PKI) for encryption or authentication, but most integration products provide the needed tools. As part of a larger plan, integration products can help in implementing such methods as using digital signatures. The tools enable the current applications to use security protocols for authentication or encryption. Also unavailable at present are multi-vendor Certificate Authorities systems that manage trust levels across vendors. Authentication levels may be implemented to the same standard (X.509) but given different names.

4) Common Portal Example

Integration technology can be instrumental in implementing portals. Application integration services may be an important part of the overall architecture that enables the Commonwealth to steadily progress towards a new citizen-centric, Internet-based government service portal with high-priority online applications for citizens. To simplify portal use by citizens, common functions needed across agencies such as credit card payment processing should be designed once and shared. Methods to verify and authenticate users may also be developed once and shared. Integration tools are extremely useful in enabling the sharing process. A centrally available integration tool set can be shared across new development efforts.

The above examples illustrate the need for the last type of integration technology, the Enterprise Service Bus, which combines all of the above integration tools and services into a managed suite.

Application Integration Guidelines

Networks that employ TCP/IP can take advantage of IIOP compliant distributed objects.

New object-oriented business applications that need to interoperate should be SOAP compliant and strong consideration should be given to an enterprise-level Services Oriented Architecture (SOA).

There are no Application Integration Servers and Services requirements at this time.

Following is the recommended practice for Application Integration Servers and Services.

INT-RP-08:	Agencies should buy interfaces/adapters wherever possible, not build them. New object-oriented business applications should be SOAP compliant.

Rationale:

- Buying commercial off the shelf (COTS) interfaces/adaptors that use mainstream technologies based on industry standard protocols and interfaces minimizes the dependence between application and platform infrastructure and simplifies the support of the distributed environment.
- Using mainstream standards-based solutions helps to assure the availability of adequate numbers of external implementers, training sources, and technical support options.

Application Integration Services Standards

Protocols and services related to application integration are noted in Technology Component Standard INT-S-08. In general, those technologies listed as strategic are based on open standards.

Table INT-S-08: Application Integration Services Technology Component Standard
Strategic:
• Object Request   ○ .NET Remoting   ○ SOAP over HTTP   ○ J2EE/RMI, Java 2 Enterprise Edition (the distributed version) and Remote Method Invocation • Enterprise Application Integration Services (EAI)   ○ Use of Integration Servers/Services   ○ SOA • Remote Procedure Calls   ○ Web Services • Object and Application Interfaces   ○ IDL (interface definition language) stubs; MIDL (Microsoft); OMG IDL; DCE IDL
Emerging:
• None
Transitional/Contained:
• Remote Procedure Calls   ○ Suns' ONC+ RPC   ○ MS DCOM + (distributed common object model)   ○ OMG CORBA (common object request broker)   ○ DCE RPC   ○ DCE secure RPC (integrated with DCE security protocols for authentication, protection level and authorization)   ○ ebXML
Obsolescent/Rejected:
• None
Exception History:
• None
Historical Note: Fully utilizing Web Services is the recommended strategic direction when combined with an overall Service-Oriented Architecture (For a description of SOA please see Appendix A of the ETA Application Domain Report [Example SOA Centralized Implementation and Governance Model]). Other methods, such as DCOM and CORBA are still used and recommended for specific scenarios.

Enterprise Service Bus (ESB):

An emerging integration category is an Enterprise Service Bus (ESB.) An ESB is a Web-services-capable integration infrastructure that supports communication and mediates application interactions. To be an ESB, an integration subsystem must:

1) implement program-to-program communication (always supporting Simple Object Access protocol/Hypertext Transfer Protocol [SOAP/HTTP], and almost always supporting SOAP on message-oriented middleware [MOM] and plain MOM);

2) support other Web services standards (including Extensible Markup Language [XML] and Web Services Description Language [WSDL]).

3) be capable of service discovery, binding and virtualization (transparently substituting alternative service providers) and intelligent message routing.

4) have an extensible, intermediary-based architecture so that additional features can be plugged in; and

5) have an awareness of message schemas through the use of metadata. [7]

Value Added Services:

One of the key advantages of an ESB is that it supports all of the communication patterns commonly used in business applications, including request/reply, reliable delivery of one-way messages, publish-and-subscribe and more complex dialogues that combine multiple messages. Earlier forms of communication integration systems were each optimized for only one of these patterns, so companies had to use many different products, each with its own programming model. Prior to ESBs, developers used sockets, remote procedure calls (RPCs), the Component Object Model (COM), Common Object Request Broker Architecture (CORBA) or MOM for program-to-program communication.

Table 3: ESB Supported Communication Integration Systems

	TCP/IP	RPC, COM, CORBA	MOM	Web Services 2005	ESB
Documented Interfaces and Events		Y		Y	Y
Service and event registration and discovery		Y		Y	Y
Industry Standards	Y	P	P	Y	Y
Qualities of Service		P	Y	P	Y
Management		P	P		P
SOA Interactions		Y	P	Y	Y
Event notification and messaging			Y		Y
Y= Yes, feature is supported P = Feature is partially supported					

Gartner Research [8]

[8] Integration Suites and ESBs: Integration Technology for the Mainstream. Jess Thompson & Roy Schulte. Gartner Research.

Generally, a common set of characteristics apply to many of the products in this category.

- Brokered Communication. The basic function of an ESB is to send data between processes on the same or different computers. Like message-oriented middleware, the ESB makes use of a software intermediary between the sender and the receiver, providing a brokered communication between them.
- Address indirection and intelligent routing. ESBs typically include some type of repository used to resolve service addresses at runtime. Also, they are typically capable of routing messages based on a predefined set of criteria.
- Basic Web services support. A growing number of ESBs support basic Web services standards including SOAP and WSDL as well as foundational standards such as TCP/IP and XML.
- Endpoint metadata. ESBs typically maintain metadata that documents service interfaces and message schemas.

In addition, some ESB vendors offer additional features including message transformation, validation, logging, and auditing.[9]

ESB Recommended Best Practices:

Although no standards presently exist, as indicated in the table above, second-generation ESBs merge the best features of SOAP/HTTP and other protocols. The "seams" between the various protocols are reduced, and new features, such as BPEL-based flow management, are being added. The next generation of ESBs will support the growing list of Web services standards. Three open-source ESBs were announced in 2H05.[10] Agencies should monitor emerging standards and take into consideration those vendors that are able to adapt their ESB tools to those emerging standards.

VITA has implemented an Integration Competency Center (ICC) and intends to implement an ESB and an SOA for in-scope agencies. Other Organizations implementing an ESB should also consider the formation of an Integration Competency Center. The ICC is a full-time, permanent group that operates as a center of excellence for other parts of the enterprise. A typical ICC might include 5 to 10 people. Realize that this is not new headcount; it is a relocation of resources that would be needed in any case for organizations implementing SOA. Typically, an Integration Competency Center is “grown” in stages depending upon the defined scope, which can be from defining and establishing standards and toolset and providing architecture and integration consultation.

to actual centralization of all integration activities. The scope for most ICCs is to serve as a center of excellence in defining best practices, selecting and implementing standard toolsets, processes and architectures, and providing consultation in order to ensure consistency and maximize value, while decentralizing actual applications integration activities. In addition, the ICC should design and develop the technical components of the service-oriented architecture.

There are no ESB requirements at this time.

[9] Microsoft on the Enterprise Service Bus. July 2005

[10] Integration Suites and ESBs: Integration Technology for the Mainstream. Jess Thompson & Roy Schulte. Gartner Research.

Following are the five recommended practices for ESB:

INT-RP-09:	Agencies should monitor emerging standards and take into consideration those vendors that are able to adapt their ESB tools to those standards.
INT-RP-10:	Organizations implementing an ESB should also consider formation of an Integration Competency Center (ICC).
NT-RP-11:	Integration Competency Centers (ICCs) should serve as a center of excellence in defining best practices, selecting and implementing standard toolsets, processes, and architectures. The ICCs should also provide consultation in order to ensure consistency and maximize value, while decentralizing actual applications integration activities. In addition, ICCs should design and develop the technical components of the service-oriented architecture.
INT-RP-12:	Integration architects and Integration Competency Center (ICC) professionals should coordinate their ESB, Web services and integration strategies because of the application design interdependencies and the overlap in the enabling integration systems.
INT-RP-13:	Organizations that are implementing ESB on a large scale should also implement SOA.

Rationale:

- Enterprise Services Buses are a method of making application integration simpler and less expensive. As Gartner[11] notes "An ESB is a Web-service capable integration infrastructure that supports communication and mediates application interactions."
- The effective use of an Enterprise Services Bus requires an integration competency center (ICC) that consistently enforces governance procedures and achieves effective use of the products and services offered through a thorough understanding of the technologies employed.
- An ICC can develop the necessary specialized skill set by integrating pre-existing technical and application skills with specific new knowledge. It can define a coherent integration policy as more and more integration projects are carried out. The competency center should provide business units, subsidiaries and external entities (e.g., suppliers and partners) with a comprehensive set of services — such as consulting services or management of an interface repository — aimed at facilitating the implementation of the application integration scenarios they need.

[1] *Integration Suites and ESBs: Integration Technology for the Mainstream*, Gartner Application Integration Web Services Summit, 5-7 December 2005, Orlando, FL

[1] Management Update : Predicts 2006: The Strategic Impact of SOA Broadens, Gartner Publication, ID Number: G00136557, 23 November 2005

Service-Oriented Architecture (SOA) Governance:

Service Oriented Architecture (SOA) is a computer systems architectural style for creating and using business processes, packaged as *services*, throughout their lifecycle. SOA separates functions into distinct units (services), which can be distributed over a network and can be combined and reused to create business applications. These services communicate with each other by passing data from one service to another, or by coordinating an activity between two or more services. Web services can be used to implement a service-oriented architecture. Each SOA building block can play one or more of three roles:

1. The *Service Provider* makes the service available with its *Service Contract* and advertises it on the *Service Broker*.
2. The *Service Consumer* finds the compatible *Service* and its *Service Contract* using the *Service Broker*.
3. The *Service Consumer* and the *Service Provider* interact.

See the *Applications Domain Report* for *Service-Oriented Architecture* description and requirements.

Service-Oriented Architecture Governance
SOA Governance is the ability to ensure that all the independent efforts (whether in the design, development, deployment, or operations of a Service) come together to meet the enterprise SOA requirements.[12] SOA Governance addresses the way reusable services are defined, designed, accessed, executed, and maintained. For most organizations, the major challenge of service-oriented architecture governance will be sharing reusable services across business units. For most organizations, SOA governance is immature or nonexistent. Many organizations are creating Integration Competency Centers who will help to define strategy and governance for SOA. Before embarking on SOA, it is highly recommended that a strategy be developed. An Integration Competency Center is a good start for an SOA project, but it needs to be reinforced with structured processes, policies, and procedures to support an effective SOA. The following guiding principles define the ground rules for development, maintenance, and usage of the SOA:

- Reuse, granularity, modularity, composability, componentization, and interoperability.
- Compliance to standards (both common and industry-specific); and
- Services identification and categorization, provisioning and delivery, and monitoring and tracking

See the Applications Domain Report for Service-Oriented Architecture Governance requirements.

[1] *Ten Golden Rules for Starting Application Integration*, Gartner Research, ID Number: TG-14-2678, 20 November 2001
[1] SOA Governance, WebLayers, Inc. 238 Main Street, 4th Floor, Cambridge, MA 02142

Instant Messaging

Instant Messaging[13] is the exchange of text messages through a software application in real-time. Generally included in the IM software is the ability to easily see whether a chosen friend, co-worker or "buddy" is online and connected through the selected service. Instant messaging differs from ordinary e-mail in the immediacy of the message exchange and also makes a continued exchange simpler than sending e-mail back and forth. Most exchanges are text-only, though popular services, such as AOL, MSN Messenger, Yahoo! Messenger and Apple's iChat now allow voice messaging, file sharing and even video chat when both users have cameras.

Gartner Research finds instant messaging (IM) has developed into a strategic enterprise communication tool. “Although consumer IM use has been predominant in businesses, we expect penetration rates for enterprise IM to be near 100% by the end of the decade[14].”
Research by Gartner published in April 2008[15] highlights the importance of an enterprisewide instant messaging (IM) strategy that includes standardization on an enterprise IM solution. Furthermore it warns that IT managers can no longer ignore the risks of free consumer services. Key findings in Gartner’s research are:

- IM is now used in over 90% of organizations (consumer and enterprise services combined).

- Depending on "lockdown" measures alone to block the use of consumer IM applications at work is ineffective if no corporate alternative is offered.

- New regulations on electronic discovery (e-discovery) in regulated industries challenge the perception that instant messages may be treated as transitory communications.

Recommended Practices for Supporting Instant Messaging[16][17]

Business strategy and he recommended practices for supporting IM and presence are presently being developed according to Gartner research. There are significant operational challenges that affect configuration, implementation and deployment, security and compliance. There are also issues around resolving the organizational alignment of business goals between IT, the communications group and business units.

Strategies for IM and presence also have to be aligned with the overall collaboration strategy and include considerations for directory services, and portal and core infrastructure directions. The Gartner IM and presence research will focus on taking the IM and presence strategy and making it applicable to real business scenarios and use cases.

Protocols and Products Related to Instant Messaging[18]

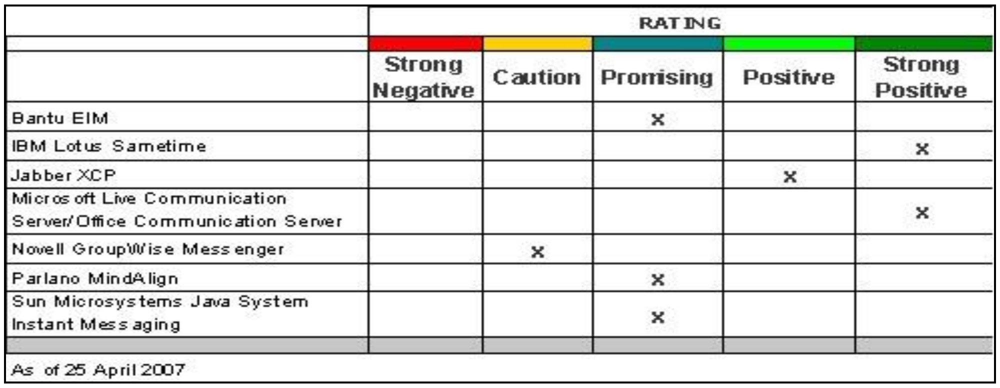
13 Wikipedia, April 2008: http://en.wikipedia.org/wiki/Instant_Messaging
14 David Mario Smith and James Lundy, Gartner, Inc. *MarketScope for Instant Messaging, 2007*, 30 April 2007, ID Number: G00147732
15 David Mario Smith and James Lundy, *Ignoring Instant Messaging at Work Won't Make It Go Away*, 10 April 2008, ID Number: G00157042
16 David Mario Smith, Gartner, Inc., *Key Issues for Instant Messaging and Presence, 2007*, 31 January

18 David Mario Smith, Gartner, Inc., *Key Issues for Instant Messaging and Presence, 2007*, 31 January 2007,

Interoperability through federation was a key trend in 2006, and the underlying issue was the existence of proprietary protocols that were not able to connect natively. SIP/SIMPLE and XMPP are the two leading IM and presence protocols.

SIP/SIMPLE is supported by IBM and Microsoft, while XMPP is supported by Jabber, Google and Sun. Jabber and Google have worked on a new protocol called Jingle which is a signaling protocol similar to SIP for initiating sessions. The emergence and maturity of these protocols, and their impact on real-time communications, will force businesses to make strategic decisions on unified communications platforms that include IM. They will also force vendors to pursue support for multiple protocols through their platforms.

Table 4: Market Scope for Instant Messaging, 2007s



	RATING				
	Strong Negative	Caution	Promising	Positive	Strong Positive
Bantu EIM			x		
IBM Lotus Sametime					x
Jabber XCP				x	
Microsoft Live Communication Server/Office Communication Server					x
Novell GroupWise Messenger		x			
Parlano MindAlign			x		
Sun Microsystems Java System Instant Messaging			x		
As of 25 April 2007					

Source: Gartner (April 2007)

There are no IM requirements at this time.

Following are the three recommended practices for IM[19]:

INT-RP-14a: Develop an IM strategy that decides on an enterprise IM standard and on measures to manage the use of IM services from consumer networks.

INT-RP-14b: *Develop full archiving* capabilities for the retention and deletion of IMs as required by the Virginia Public Records Act § 42.1-76 et seq. of the Code of Virginia and the Library of Virginia's State Agency General Schedules.

INT-RP-14c: Develop an enterprise IM standard that:

1. Includes the creation of a governance process that addresses:

   1.1. IM acceptable-usage in accordance with the Department of Human Resource Management's *Policy Number: 1.75 - Use of the Internet and Electronic Communications Systems*.

   1.2. the requirements of the Commonwealth's IT security and standard; and

1.3. the requirements of the *Code of Virginia* § 2.2-3803 (B), the Internet privacy policy.

2. Expressly prohibit the use at work by state employees of consumer IM services and tools.

Rationale:

- Gartner's research[20] indicates "IM has penetrated over 90% of organizations, with consumer-based tools – doubtless deployed without the sanction of IT staff – predominating." It further determines "The best way to stop staff using consumer IM services is to provide an enterprise alternative."

- New regulations on electronic discovery (e-discovery) in regulated industries challenge the perception that instant messages may be treated as transitory communications.

- Mitigates threats, such as malicious code, entering the enterprise from outside and the leakage of potentially sensitive information to external parties.

[20] David Mario Smith and James Lundy, *Ignoring Instant Messaging at Work Won't Make It Go Away*, 10 April 2008, ID Number: G00157042 Retrieved May 2008

Instant Messaging Standards

Products and services related to instant messaging are noted in Technology Component Standard INT-S-09. In general, those technologies listed as strategic are based on open standards.

Table INT-S-09: Instant Messaging Technology Component Standard
Strategic:
③ IBM Lotus Sametime ③ Jabber XCP ③ Microsoft Live Communications (Server/Office Communication Server)
Emerging:
③ Bantu EIM ③ Parlano MindAlign ③ Sun Microsystems Java System Instant Messaging
Transitional/Contained:
③ Novell GroupWise Messenger
Obsolescent/Rejected:
③ None
Table INT-S-09: Instant Messaging Technology Component Standard
Exception History:
③ None **Historical Note**: None

Mashup

A "mashup" is a lightweight, tactical presentation layer integration of multi-sourced applications or content into a single, browser-compatible offering. Mashups25 currently come in three general types: consumer mashups, data mashups, and business mashups.

Mashups leverage content and logic from other Web sites and Web applications and are built with a minimal amount of code (which can be client-side JavaScript or serverside scripting languages, such as PHP or Python). Mashups aren't intended to be strategic, systematically built, industrial-strength enterprise applications; rather, they're created quickly or opportunistically to meet a focused tactical need. Mashups are generally personalized to fulfill personal productivity needs rather than the requirements of a longstanding corporate role.

[1] The source for much of the information presented in the Mashup sections was obtained through Gartner Research, Gartner, Inc. Stamford, CT. The articles researched are listed in the in Appendix C.
[1] Anthony Bradley, Daniel Sholler, David Gootzit. *Enterprise IT Departments Must Prepare for the Impact of "Mashups"* 7 September 2007 Gartner Research: ID G00151424 Retrieved November 2007. [25] Wikipedia: http://en.wikipedia.org/wiki/Mashup_%28web_application_hybrid%29. Retrieved November 2007.

The mashup phenomenon emerged from the combination of Web-based presentation layer protocols (such as HTML, XML and RSS) and Web-based visualization APIs. This distinguished mashups from other integration technologies, such as SOA, that target application-to-application integration. Mashups are emerging as a "face of SOA" because they provide a visualization capability for SOA implementations. However, the messages must be reformatted into a "mashable" Web Oriented Architecture (WOA) format (for example, plain old XML [POX], RSS or Atom) for SOA-based services to participate in a

Methods to Integrate Mashups
Definitions for this section:

- Event-driven architecture (EDA)
- Service-oriented architecture (SOA)
- Web-oriented architecture (WOA)
- Uniform resource identifiers (URIs)
- Representational state transfer (REST)

A mashup application is architecturally comprised of three different participants that are logically and physically disjoint (they are likely separated by both network and organizational boundaries): API/content providers, the mashup site, and the client's Web browser.

- The API/content providers. These are the (sometimes unwitting) providers of the content being mashed. To facilitate data retrieval, providers often expose their content through Web-protocols such as REST, Web Services, and RSS/Atom (described below). Mashups that extract content from sites like Wikipedia, TV Guide, and virtually all government and public domain Web sites do so by a technique known as *screen scraping*. In this context, screen scraping connotes the process by which a tool attempts to extract information from the content provider by attempting to parse the provider's Web pages, which were originally intended for human consumption.

- The mashup site. This is where the mashup is hosted. Interestingly enough, just because this is where the mashup logic resides, it is not necessarily where it is executed. Alternatively, mashed content can be generated directly within the client's browser through client-side scripting (that is, JavaScript) or applets. This client-side logic is often the combination of code directly embedded in the mashup's Web pages as well as scripting API libraries or applets (furnished by the content providers) referenced by these Web pages. Mashups using this approach can be termed *rich internet applications* (RIAs), meaning that they are very oriented towards the interactive user-experience. (Rich internet applications are one hallmark of what's now being termed "Web 2.0", the next generation of services available on the World Wide Web.) The benefits of client-side mashing include less overhead on behalf of the mashup server (data can be retrieved directly from the content provider) and a more seamless user-experience (pages can request updates for portions of their content without having to refresh the entire page). Often mashups use a combination of both server and client-side logic to achieve their data aggregation. Many mashup applications use data that is supplied directly to them by their user base, making (at least) one of the data sets local. Additionally, performing complex queries on multiple-sourced data (such as "Show me the average purchase price for real estate bought by actors who have co-starred in movies with Kevin Bacon") requires computation that would be infeasible to perform within the client's Web browser.

Wikipedia: http://whatis.techtarget.com/definition/0,,sid9_gci1167147,00.html. Retrieved December 2007.

- The client's Web browser. This is where the application is rendered graphically and where user interaction takes place. As described above, mashups often use client-side logic to assemble and compose the mashed content.

When used together, the goal of these technologies is to create a smooth, cohesive Web experience for the user by exchanging small amounts of data with the content servers rather than reload and re-render the entire page after some user action. Mashups can be constructed using native "plain old XML" (POX), on HTTP Web architecture is an alternative to Web service formats and protocols that leverage SOAP and EDA. Web architecture is based on the concept of resources that are uniquely identified through the use of URIs, generally communicating using standard data formats, such as HTML and XML, and simple HTTP verbs such as "get," "put," "post" and "delete." At the core the focus is on understanding the relative merits of Mashups, decision criteria for the use of REST and POX Web-based protocols such as Atom and Really Simple Syndication.

Thousands of mashups on the Web are often built, shared, modified and used by professional and nonprofessional programmers. For example, HousingMaps.com combines data from Google Maps with apartment rental information from craigslist to create a new application that shows the location of available apartments in a given city.

Mashups often combine data from one or more sources with a visualization[21] capability. The most widespread example of this application is the rapid expansion of geographical mashups, where geo-location data from a variety of sources (such as real estate, retail outlets and airports) is visualized using the Web-based mapping APIs (such as Google Maps, MapQuest and Microsoft MapPoint). Web APIs and mashups are available in an online directory at www.programmableweb.com .

Mashup Integration Services

Like any other data integration domain, mashup development is replete with technical challenges that need to be addressed, especially as mashup applications become more feature- and functionality-rich. This section touches on a handful of these challenges, some of which can be addressed and mitigated, while others are open issues.

Mashups are driven by the Web culture that is, social networking sites tailored to communities of interest. There are thousands of mashups on the Web, often built by nonprofessional programmers.

Mashups are used to quickly integrate content or functions from multiple sources and present easily understandable items of interest. The trade-off is faster time to market and reduced development costs over application robustness and longevity.

The mashup architecture is intended to capture the critical capabilities required for an enterprise mashup environment. Use the term "environment" rather than "system," "solution" or "ecosystem" to show that this is an enterprise-built and supported environment where users can discover high-value mashup components and assemble them into effective mashups. Users also can share, rate, and modify mashups and mashup components within this environment. The purpose of this architecture is to provide a framework for understanding and describing a robust enterprise mashup environment. The architecture consists of eight layers:

[21] Whatis definition: Visualization is the process of representing abstract business or scientific data as images that can aid in understanding the meaning of the data. (Source: http://whatis.techtarget.com/definition/0,,sid9_gci213311,00.html) Retrieved April 2008..

Mashup Sources (Information and Function)

Mashups do not have native data and functionality. They must source capabilities from other systems. Originally, mashups sourced Web-based information and functionality by leveraging open application programming interfaces, such as Google Maps, and information feeds — such as Really Simple Syndication (RSS), Atom and plain old XML over HTTP.

Mashup Assembly

The mashup assembly layer is the core of a mashup environment. The fundamental capabilities of this layer include access to mashup components, the means to assemble the components into a mashup and the ability to preview the result. Mashup components go by different names depending on the vendor (for example, gadgets, widgets, pipes, blocks and cords) but they all serve the same basic purpose. A mashup component is the encapsulation of a source's data or functionality to facilitate mashing. For example, mashup components include a weather widget that displays the weather for a particular zip code, an RSS feed reader gadget that aggregates desired feeds or an investment banking enterprise widget that calculates the net present value of an investment stream.

The mashup assembly layer provides the capability to combine mashup components into mashups. Combining may be as basic as collocating related mashups on a Web page using Web-top products such as iGoogle, Netvibes and Pageflakes, or as advanced as wiring together multiple mashup data, function and visualization components to deliver a more sophisticated mashup result.

Before users can mash up components, they must be able to find them and understand their value. This aspect is often overlooked or undervalued. Mashup component discoverability and usability is critical to achieving widespread user participation in an enterprise mashup environment. This includes basic search functionality of an underlying repository or the external Web. However, more-mature mashup assembly discovery capabilities will include component metadata that will facilitate user understanding of the components' provenance, value and "wiring" capabilities. Future evolution of mashup assembly may involve intelligent selection that, when a user selects a base mashup component, filters the list of other components to remove components that do not mash with the selection. For example, if a military intelligence analyst is seeking intelligence reports on a specific region for visualization with a mapping gadget, then the search results will only retrieve intelligence information report gadgets that have geo-location information as outputs. In this way, the intelligence analyst is only given search result components that can be mashed with the mapping gadget.

Mashup assembly also enables users to preview the results of their mashup prior to "publishing" it for their general use or making it available for use within the mashup community. The preview offers users the chance to see if the mashup has met their objectives or whether it needs adjustment. This capability can range from simple visualization to various levels of testing and assistance. This preview and visualization functionality may overlap with, or leverage, the mashup visualization layer.

Mashup Visualization

This layer covers the visualization destination for a mashup. This destination is a Web browser-based user interface that presents the final mashup result. The user interface is usually a Web page or site, portal or Web-based application. A Web page or a series of Web pages can compose a mashup application with one or more mashups that contribute value to the overall user business process or objective. Also, portals can capitalize on mashups to extend functionality and provide a more dynamic means of customization for specific subsets of the target user base. Finally, mashups can deliver a means of augmenting Web-based applications with information from other systems.

Information Access, Augmentation and Delivery

Mashups applied to the enterprise expand the need for access to a wider variety of structured and unstructured source systems, such as legacy applications, databases, documents, spreadsheets, flat files and HTML pages. These sources are often not Web oriented and, therefore, do not natively support mashups. Access, Augmentation, and Delivery (AAD) layer technologies include adapters, robots and other technologies to gain access to source systems. Although not mandatory, the source information is often augmented through cleansing and processing. Finally, the information is made available for delivery (as outputs) in a WOA "mash able" format.

Mashup Development

The mashup development layer provides professional developers with the capability to build and manage high-value, enterprise-ready mashup components as building blocks for users to rapidly create mashup applications. The mashup environment has a built-in mashup component development capability, or the mashup environment integrates with an existing development environment. This capability is crucial because a mashup environment is only as valuable as the mashup components it supports. Developers create mashup components as chunks of enterprise data and functionality that are desirable for mashing. Enterprise IT developers supporting the mashup environment should spend the bulk of their time ensuring that the core of the mashup component repository is high performance, highly valuable and highly usable.

Mashup Processing

The basic capabilities of mashups often need enhancement as part of an enterprise solution. These enhancements often involve post-mashup processing to incorporate a mashup within a mashup application or within a "bigger picture" system solution. An example is the employment of mashups as the end-user visualization capability of a service-oriented architecture or event driven architecture. Mashup processing can involve the addition of workflow capabilities to mashups and integration into other enterprise systems. Mashup processing layer capabilities are immature, but they hold the potential for significant value in managing mashups as a robust IT capability and in tying mashups into enterprise solutions.

Mashup Infrastructure

A robust enterprise mashup environment will require some level of mashup infrastructure support. Capabilities such as security, governance, administration, repository management, user support and quality of service are considered infrastructure. They provide the enterprise with confidence that the mashup environment is managed as a corporate asset. Enterprise IT must strategically define what infrastructure services to provide in supporting the mashup environment. Enterprise IT must develop a core competency in determining, managing and facilitating those mashup environment aspects that require enterprise support, those that are delegated to the users and those that can be managed by the community of participants.

Mashup Community

Many mashup environments adhere to mashups' Web 2.0 roots and are communityenabled. This means that the mashup environment is also managed as a community where participants can build, share, use, rate, and modify mashups and mashup components. The community approach delivers a significant amount of value to the mashup environment and offloads much of the governance and management burden. The community can police itself by helping to determine which mashups are most effective and easiest to use. Community sharing also enhances the value of mashups and mashup components by reducing duplicate capabilities, decreasing development time and increasing efficiency. Community participation can illuminate the highly valuable mashups and perhaps elevate them from being user-managed tools to enterprise ITmanaged ones. The enterprise IT organization should strive to leverage the benefits of community in delivering and managing an enterprise mashup environment.

Mashup Integration Examples

Will any of these mashup's morph beyond the hobbyist, and gee-ma-look-what-I-can-do realm into actual, revenue-producing businesses? Gartner's Hype Cycle for Emerging Technologies 2007 documents the Maturity Level as Emerging and provides these sample vendors:

- *Sample Vendors:* Applibase (datamashups.com); BEA Systems; craigslist; CommunityWalk; FeedBurner; Google; housingmaps.com; IBM; JackBe; Kapow; Microsoft; Nexaweb Technologies; Oracle; RSSBus; Yahoo

Numerous Websites present hundreds of code samples, libraries, API wrappers and other time-saving resources. The following is a list of top tags used in mashups: mapping, photo, shopping, search, travel, video, news sports, real-estate, and messaging.

Mashups Integration Guidelines

Like any other data integration domain, mashup development is replete with technical challenges that need to be addressed, especially as mashup applications become more feature- and functionality-rich. This section touches on a handful of these challenges, some of which can be addressed and mitigated, while others are open issues.

Data Integration Challenges: Semantic Meaning and Data Quality

Qualitative surveys suggest that the number one enterprise IT concern today is data integration within the enterprise virtual organization. (In this context, the term *virtual organization* means a composition of federated business units, each contained within its own administrative domain.) Like many enterprise IT managers who find themselves up to the task of integrating legacy data sources (for example, to create corporate dashboards that reflect current business conditions), mashup developers are faced with the analogous challenges of deriving shared semantic meaning between heterogeneous data sets. Therefore, to get an idea for what mashup developers have in store, you need look no further than the storied integration challenges faced by enterprise IT. For example, translation systems between data models must be designed. When converting data into common forms, reasonable assumptions often have to be made when the mapping is not a complete one (for example, one data source might have a model in which an address-type contains a country-field, whereas another does not). Already challenging, this is exacerbated by the fact that the mashup developers might not be domain experts on the source data models because the models are third-party to them, and these reasonable assumptions might not be intuitive or clear. In addition to missing data or incomplete mappings, the mashup designer might discover that the data they wish to integrate is not suitable for machine automation; that it needs cleansing.

Another host of integration issues facing mashup developers arise when screen scraping techniques must be used for data acquisition. Deriving, parsing, acquisition tools and data models require significant reverse-engineering effort. Even in the best case where these tools and models can be created, all it takes is a re-factoring of how the source site presents its content (or mothballing and abandonment) to break the integration process, and cause mashup application failure.

Component Challenges

The Ajax model of Web development can provide a much richer and more seamless user experience than the traditional full-page-refresh, but it poses some difficulties as well. At its fundamentals, Ajax entails using the browser's client-side scripting capabilities in conjunction with its Document Object Module

(DOM) to achieve a method of content delivery that was not entirely envisioned by the browser's designers. (Perhaps this hacklike nature of Ajax lends to its appeal.) However, this subjects Ajax-based applications to the same browser compatibility issues that have plagued Web designers ever since Microsoft created Internet Explorer. For example, Ajax engines make use of an "XMLHttpRequst" object to exchange data asynchronously with remote servers. In Internet Explorer 6, this object is implemented with ActiveX rather than native JavaScript, which requires that ActiveX be enabled.

A more fundamental requirement is that Ajax requires that JavaScript be enabled within the user's browser. This might be a reasonable assumption for the majority of the population, but there are certainly users who use browsers or automated tools that either do not support JavaScript or do not have it enabled. One such set of tools are the robots, spiders, and Web crawlers that aggregate information for Internet and intranet search engines. Without graceful degradation, Ajax-based mashup applications might find themselves missing out on both a minority user base as well as search engine visibility.

The use of JavaScript to asynchronously update content within the page can also create user interface issues. Because content is no longer necessarily linked to the URL in the browser's address bar, users might not experience the functionality that they normally expect when they use the browser's BACK button, or the BOOKMARK feature. And, although Ajax can reduce latency by requesting incremental content updates, poor designs can actually hinder the user experience, such as when the granularity of update is small enough that the quantity and overhead of updates saturate the available resources. Also, take care to support the user (for example, with visual feedback such as progress bars) while the interface loads, or content is updated.

As with any distributed, cross-domain application, mashup developers and content providers alike will also need to address security concerns. Sensitive data is also likely to require confidentiality (that is, encryption), and you must take care when you mash it with other sources to not put it at risk. Identity will also be crucial for auditing and regulatory compliance. Additionally, with data integration happening both on the server and client-side, identity and credential delegation from the user to the mashup service might become a requirement.

Risks:
Because mashups combine data and logic from multiple sources, they're vulnerable to failures in any one of those sources. There are also risks and concerns regarding the use of intellectual property and the longevity of provider relationships.

Mashup Standards

Protocols and services related to mashups are noted in Technology Component Standard INT-S-10. In general, those technologies listed as strategic are based on open standards.

Table INT-S-10: Mashup Technology Component Standard
Strategic:
Table INT-S-10: Mashup Technology Component Standard
• Ajax - AJAX (Asynchronous JavaScript and XML), or Ajax, is a group of inter-related web development techniques used for creating interactive web applications. • EDA - Event-driven architecture • SOA - Service-oriented architecture • WOA - Web-oriented architecture • URI - Uniform resource identifiers • Rest - Representational state transfer • ATOM - the *Atom Publishing Protocol* is a simple HTTP-based protocol for creating and updating web resources. • RSS - RSS (Really Simple Syndication) is a family of Web feed formats used to publish frequently updated content such as blog entries, news headlines or podcasts. Use available API's wherever possible **http://www.programmableweb.com/apis/directory/1?sort=mashups**
Emerging:
• None
Transitional/Contained:
Obsolescent/Rejected:
• None
Exception History:
• None **Historical Note**:

Definitions and Terminology

ACMS	A transaction processing monitor from Compaq that runs on the open VMS operating system.
Active X	Microsoft's answer to Java. ActiveX is a stripped-down implementation of OLE designed to run over slow Internet links.
ADSI	Active Directory Service Interfaces (ADSI) abstract the capabilities of different directory services from different network vendors to present a single set of directory service interfaces for managing network resources
Agency	**State agency or agency -** Any agency, institution, board, bureau, commission, council, or instrumentality of state government in the executive branch listed in the appropriation act. ETA requirements/standards identified in this report are applicable to all agencies including the administrative functions (does not include instructional or research functions) of institutions of higher education, unless exempted by language contained in a specific requirement/standard.
API	Application Program Interface or Application Programming Interface.
APPC LU6.2	APPC allows user written programs to perform transactions in a Client-Server IBM network to access a CICS, in MVS "batch" through APPC/MVS, in VM/CMS, in AIX on the RS/6000, and on the AS/400
ASCII	American Standard Code for Information Interchange. "Human readable text." The first 128-character codes of any of the ISO 8859-character sets is always identical to the ASCII character set
ASP	Active Server Page (Microsoft) A scripting environment for Microsoft Internet Information Server in which you can combine HTML, scripts and reusable ActiveX server components to create dynamic web pages.
Asynchronous/ Connectionless Communi-cation	A program-to-program communication model that does not block any of the communicating partners and that allows for time independent interactions.
Authentication	Verification that a user is who they say they are.
B2G	Business to Government. Refers to a business process involving electronic interaction of business partners.

BOA	Basic Object Adapter protocol. Replaced by POA, Portable Object Adapter.
C2G	Customer to government. Refers to a business process involving electronic interaction of citizens with government.
CA	Certificate authority. A system for managing certified digital signatures. Manages the implementation of policies to authenticate, authorize and revoke the assignment of keys to users.
CICS	IBM mainframe application server that provides industrial-strength, online transaction management for mission-critical applications. on MVS/ESA, OS/390, VSE/ESA and z/OS. Thirty years old but repackaged to turn mainframes into Web servers.
COM	Component Object Model (Microsoft); also DCOM and DCOM+ for distributed systems
Cookies	Information stored on a Website visitor's computer regarding a transaction with a Website that may be returned to that Website at each subsequent visit if requested by the Website.
CORBA	Common Object Request Broker Architecture. OMG's open, vendor-independent architecture and infrastructure that computer applications use to work together over networks.
COTS	Virginia's Council on Technology Services. COTS is a stakeholder driven body, representing the interests and needs of the enterprise as a whole, including the Executive, Legislative, and Judicial branches of state government. The purpose of the Council is to advise the Chief Information Officer of the Commonwealth on the services provided by the Virginia Information Technologies Agency (VITA) and the development and use of applications in state agencies and public institutions of higher education (http://www.vita.virginia.gov/cots/).
CPI	Common Program Interface. IBM's Systems Application Architecture API.
CSS	Cascading Style Sheets. An XML protocol used to control formatting of Web pages.
Data	Data is the plural of *datum*. A datum is a *statement accepted at face value* (a "given"). A large class of practically important statements that are measurements or observations of a variable. Such statements may comprise numbers, words, or images.[22]

[22] *Wikipedia, The Free Encyclopedia*. Retrieved 20:38, January 25, 2006 from http://en.wikipedia.org Data.

Data Marshaling	The conversion of data between platform specific representations and the packaging according to the requirements of a particular network protocol in order to perform the data transport between different nodes.
DCE	Distributed Computing Environment from Open Computing Group. Includes Remote Procedure Call (RPC), the Cell and Global Directory Services (CDS and GDS), the Security Service, DCE Threads, Distributed Time Service (DTS), and Distributed File Service (DFS).
DCOM+	The Distributed Component Object Model. A set of Microsoft protocols that enable software components to communicate directly over a network.
Domain	The Enterprise Technical Architecture (ETA) is typically divided into logical groups of related technologies and components, referred to as “domains”. The purpose of a Domain Architecture is to provide a combination of domain principles, best practices, reusable methods, products, and configurations that represent “reusable building blocks”. Thus, the Domain Architecture provides the technical components within the Enterprise Architecture that enable the business strategies and functions. Note, the Conceptual Architecture serves as the foundation for the Domain Architectures and ensures that they are aligned and compatible with one another.[23]
DNS	Domain name system. A general-purpose, distributed, replicated, data query service chiefly used for Internet communications for translating hostnames into IP addresses.
DTD	Document Type Definition. An XML protocol for communicating tagging standards that will be used in an XML communication. The definition of a document type in SGML or XML, consisting of a set of mark-up tags and their interpretation.
E2G	Employee to government. Refers to a business process involving electronic interaction of citizens with government.
EAI	Enterprise Application Integration. The use of technology to integrate the application programs, databases, and legacy systems involved in an organization's critical business processes.

[23] *“Commonwealth of Virginia Enterprise Architecture – Common Requirements* COTS EA Workgroup, *Vision”*, v1.1, December 5, 2000, p 26.

ebXML	ebXML is a set of specifications that together enable a modular electronic business framework. The vision of ebXML is to enable a global electronic marketplace where enterprises of any size and in any geographical location can meet and conduct business with each other through the exchange of XML based messages. ebXML is a joint initiative of the United Nations (UN/CEFACT) and OASIS, developed with global participation for global usage
EBCDIC	Extended Binary Coded Decimal Interchange Code. IBM's 8-bit extension of the 4-bit Binary Coded Decimal encoding of digits 0-9 (0000-1001).
EDI	Electronic Data Interchange. EDI works by providing a collection of standard message formats and element dictionary that can be used by businesses to exchange electronically. EDI is used for electronic commerce. EDI interchanges use some variation of the ANSI X12 standard (USA) or EDIFACT (UN sponsored global standard).
Emerging	Rating category used in this document to rate integration technologies. This technology requires additional evaluation in government and university settings. This technology may be used for evaluative or pilot testing deployments or in a higher education research environment. Any use, deployment or procurement of this technology beyond higher education research environments requires an approved *Commonwealth Enterprise Technical Architecture Exception*. The results of an evaluation or pilot test deployment should be submitted to the **VITA Strategic Management Services: Policy, Practice and Architecture Division** for consideration in the next review.
Enterprise	As used in this document and generally when discussing Enterprise Architecture topics, the *enterprise* consists of all Commonwealth of Virginia agencies as defined above.
ERwin	A database design and optimization tool from Computer Associates.
ESMTP	Extended SMTP. Initially defined in RFC 1869 and extended thereafter
EWTA	Enterprise-wide technical architecture.
Extensible	Quality of a system that allows new features and functions to be added to it.

Firewall	A dedicated gateway machine with special security precautions on it, used to service outside network, especially Internet, connections and dial-in lines. The idea is to protect a cluster of more loosely administered machines hidden behind it from crackers. The typical firewall is an inexpensive microprocessor-based UNIX machine with no critical data, with modems and public network ports on it, but just one carefully watched connection back to the rest of the cluster. The special precautions may include threat monitoring, callback, and even a complete iron box keyable to particular incoming IDs or activity patterns. Firewalls often run proxy gateways.
FTP	File Transfer Protocol. A protocol used to transmit whole files over the Internet. Security with FTP.
G2C	Government to Customer. Refers to a business process involving electronic interaction of government with citizens.
GDS	Global Directory Services, such as DNS and GDS (X.500), grew out of the computer industry's need to reference objects in distributed networks across an entire enterprise and worldwide.
GIS	Geographic Information System.
HTML	HyperText Markup Language – A subset of SGML. A W3C standard for formatting Web pages.
HTTP	HyperText Transfer Protocol. The protocol used on the World-Wide Web for the exchange of HTML documents. It conventionally uses port 80.
HTTP MPOST and HTTP POST	“A SOAP request can use HTTP's POST verb. In fact, however, the protocol requires that the first request to a server is made using MPOST. M-POST is a new HTTP verb defined using the HTTP Extension Framework (http://www.w3.org/Protocols/HTTP/ietfhttp-ext). If a request made using M-POST fails, the client can try again using a standard POST request. (In this case, future requests can also use POST because the server obviously doesn't support MPOST.) M-POST allows sending HTTP headers that can't be sent via the standard POST verb, providing more flexibility for SOAP users. Firewalls can even force the use of M-POST if desired, by simply refusing all HTTP POSTs with a content type of "text/xmlSOAP".
Hypertext	Hypertext is text that contains links to other text
IANA	The central registry for various "assigned numbers": Internet Protocol parameters, such as port, protocol, and enterprise numbers; and options, codes, and types. The currently assigned values are listed in the "Assigned Numbers" document STD 2. To request a number assignment, e-mail <iana@isi.edu>.

IDL	Interface Definition Language defined by OMG is a language for describing the interfaces of software objects. Various Vendors have their own version of IDL (e.g., MIDL by Microsoft).
IETF	Internet Engineering Taskforce. A standards group that works on Internet architectural issues.
IIOP	Internet Inter-ORB Protocol. A protocol that defines a way for Remote Procedure vendor to map messages to the TCP network communication protocol.
IMAP	Internet Message Access Protocol. It permits a "client" email program to access remote message stores as if they were local.
Information	Too often the words information and data are used interchangeably which leads to confusion. Data is unstructured, lacks context and may not be relevant to the recipient. When data is correctly organized, filtered and presented with context it can become information because it then has "value" to the recipient.[24]
Interface Repository	Interface Repository. The interface repository is part of object-oriented integration. It contains the definitions of all the services that objects can provide. The definitions form the contract by which a client can invoke requests upon a server object.
IP	Internet Protocol. A network addressing protocol. Two versions are defined: IPv4 and IPv6.
IP address	An identifier for a computer or device on a TCP/IP network. Networks using the TCP/IP protocol to route messages based on the IP address of the destination. The format of an IP address is a 32-bit numeric address written as four numbers separated by periods. Each number can be zero to 255. For example, 1.160.10.240 could be an IP address. Within an isolated network, you can assign IP addresses at random as long as each one is unique. However, connecting a private network to the Internet requires using registered IP addresses (called Internet addresses) to avoid duplicates.
ISO	International Standards Organization.
IT	Information Technology
LDAP	Lightweight Directory Access Protocol. A protocol for accessing on-line directory services. LDAP was defined by the IETF to encourage adoption of X.500 directories. The Directory Access Protocol (DAP) was seen as too complex for simple Internet clients to use. LDAP defines a relatively simple protocol for updating and searching directories running over TCP/IP.

[24] *Wikipedia, The Free Encyclopedia*. Retrieved 21:33, January 25, 2006 from Information is not data. http://en.wikipedia.org.

J2EE	Java 2 Enterprise Edition. The distributed version of Sun's Java platform. with Enterprise JavaBeans^{TM} (EJB^{TM}), JavaServer Pages^{TM} (JSP^{TM}) and Java Servlet API component technologies.
Java	Portable language from Sun designed to run on any machine with a Java Virtual Machine interpreter.
JDAP	Java Directory Access Protocol --an implementation of the Lightweight Directory Access Protocol.
JDOM	Java document object model. A way to represent an XML document for easy and efficient reading, manipulation, and writing.
JDBC	Java Database Connectivity is a standard SQL database access interface. It comes with an ODBC bridge.
Load Balancing	Load balancing means that requests from clients are distributed across available servers to achieve better utilization of computing resources. In general, load balancing can be based on network traffic, CPU load, relative power of the server, size of the server's request queue, a simple round robin method, or other mechanisms.
Loosely Coupled	Architectures based on publish/subscribe communications can provide a lightweight and resilient foundation for applications that do not require tight coordination.
MAPI	Messaging Application Programming Interface. A protocol used to write components that connect to different mail servers, provide access to custom address books and provide rich storage facilities.
MDC	Meta Data Coalition
Metadata (also Meta data)	Data about data that makes the process of finding and using data easier.
MIME	Multipurpose Internet Mail Extensions. An official Internet standard that specifies how messages must be formatted so that they can be exchanged between different email systems.
MOM	Message Oriented Middleware delivers messages from one software module to another. Modules do not have to execute on the same machine. Analogous to the US Mail. The mail is typically delivered when you're at work; you pick it up at your convenience.
Monolithic Application	An application that is entirely installed on one machine.
MTA	Message Transfer Agent. The internal component of an e-mail delivery system, responsible for mail collection from and distribution to MUAs, and relay of mail between e-mail post offices. Also called e-mail server.

MUA	Mail User Agent. Primary entry and exit point for an e-mail system. Also called an e-mail client.
Multithreaded	Sharing a single CPU between multiple tasks (or "threads") in a way designed to minimize the time required to switch threads.
Naming Service	Naming service refers to the ability of application programs to locate application components offered by other applications in a distributed environment. Typical naming service should support registration of services in the naming service and their subsequent location through the naming service.
NDS	Netware Directory Services. A hierarchical, class-based directory structure for accessing network resources.
NIST	National Institute of Standards and Technology. Formerly, the National Bureau of Standards. A United States governmental body that helps to develop standards.
N-tier	Describes a method of dividing an application into three or more physical or logical tiers to provide for ease of maintenance and flexibility. Any architecture that utilizes a 3-tier architecture, which componentizes one or more of the logical tiers is said to be n-tier. Typically, this componentization occurs in the business rule tier, however this is not a requirement. An n-tiered application is designed to integrate a diverse collection of reusables, component-based services into a unified system. The layers may operate in multiple configurations, using any number of physical systems. This architecture provides a flexible and scalable solution for meeting the State's current and future requirements.
Obsolescent	Rating category used in this document to rate integration technologies. This technology may be waning in use and support, and/or has been evaluated and found not to meet current Commonwealth Technical Architecture needs. Agencies shall not make any procurements or additional deployments of this technology. Agencies currently using this technology should plan for its immediate replacement with “strategic” technology to avoid substantial risk. The migration or replacement plan should be included as part of the Agency’s IT Strategic Plan.
ODBC	Open Data Base Connectivity. ODBC is based on Call-Level Interface and was defined by the SQL Access Group. Microsoft was one member of the group and was the first company to release a commercial product based on its work (under Microsoft Windows), but ODBC is not a Microsoft standard.

OLE	Object Linking and Embedding. The software capability that enables the creation of a compound document that contains one or more objects from one or more applications. Objects can be linked or embedded in the compound document. Changes to linked objects are reflected in the source and vice versa. Embedding objects breaks all links.
OLE-DB	Microsoft's interface to data. OLE-DB is an open specification designed to build on the success of ODBC by providing an open standard for accessing all kinds of data.
OMG	Object Management Group. A consortium aimed at setting standards in object-oriented programming.
ONC+ RPC	Open Network Computing (Sun) Remote Procedure Call. A remote procedure call or function call protocol developed by Sun.
Open Group	The Open Group is a standards development and product approval consortium. “The Open Group's Mission is to offer all organizations concerned with open information infrastructures a forum where we can share knowledge, integrate open initiatives, and certify approved products and processes in a manner in which they continue to trust our impartiality.”
Open Standards	Standards that are available for all vendors to use in product development.
Open System	A desirable but unachievable computing architecture state. A system built using open rather than proprietary standards. A product based on widely implemented vendor-neutral standards.
ORB	Object Request Broker. A software tool that enables the location of and access to objects in a distributed system.
ORCA	Online Review and Comment Application is a web-based application managed by VITA to allow public comment and review of proposed policies, standards, and guidelines. ORCA may be accessed through the Commonwealth Project Management Web page or by pointing your Web browser to the URL http://apps.vita.virginia.gov/publicORCA.

OSI Reference Model	Open Systems Interconnect seven-layer model. A model of network architecture and a suite of protocols (a protocol stack) to implement it, developed by ISO in 1978 as a framework for international standards in heterogeneous computer network architecture. The OSI architecture is split between seven layers, from lowest to highest: 1 physical layer, 2 data link layer, 3 network layer, 4 transport layer, 5 session layer, 6 presentation layer, 7 application layer. Each layer uses the layer immediately below it and provides a service to the layer above. In some implementations, a layer may itself be composed of sub-layers.
POA	Portable Object Adapter standard. An adapter written using IDL.
PDA	Personal Digital Assistants
Persistence Service	Defines a service when an object state can be preserved in a persistent media such as an object database.
PKI	Public Key Infrastructure. A way to distribute security and encryption keys.
POP3	Post Office Protocol version 3. The most common protocol used by MUAs to retrieve mail from a central message store (messaging server). Most commercial Internet Mail post office products include a POP3 server. IMAP is typically a better choice than POP3 for unified messaging.
Portability	1) The ability to pick-up, store and delivery messages everywhere. 2) The ease with which a piece of software (or file format) can be "ported", i.e., made to run on a new platform and/or compile with a new compiler.
Principles	High-level fundamental truths, ideas or concepts that frame and contribute to the understanding of the Enterprise Architecture. They are derived from best practices that have been assessed for appropriateness to the Commonwealth Enterprise Architecture.[25]
Protocol	A set of rules. For example, network protocols are rules that enable connectivity and communication.
Protocol Stack	A software subsystem that manages the flow of data on a communications channel according to the rules of a particular protocol, for example the TCP/IP protocol. Called a “stack” because it is typically designed as a hierarchy of layers, each supporting the one above and using the one below.

[25] *“Commonwealth of Virginia Enterprise Architecture – Conceptual* COTS EA Workgroup, *Architecture”,* v1.0, February 15, 2001, p 5

Publish & Subscribe	1) Providers of information can publish it for consumption by information consumers, without any logical connection between the participating applications. 2) Software or protocols that enable publishing and subscribing.
Quality of Service	1) Reliable message delivery (no messages are lost in case of system failure). 2) Guaranteed message delivery (messages are delivered within a defined time limit, even in the case of network or system unavailability). 3) Assured message delivery (messages are delivered at most once).
Recommended Practices	Are activities which are normally considered leading edge or exceptional models for others to follow. They have been proven to be successful and sustainable and can be readily adopted by agencies. They may or may not be considered the ultimate “best practice” by all readers but for this place and time they are recommended practices and should be used and implemented wherever possible.
Repository	A repository is a collection of resources that can be accessed to retrieve information. Repositories often consist of several databases tied together by a common search engine.
Requirements	Are activities that are considered strategic components of the Commonwealth’s Enterprise Technical Architecture. They are acceptable activities for current deployments and must be implemented and used for all future deployments.
Reusable Component	A sub-object derived from an object or a class of objects by taking advantage of inheritance properties. The derived object inherits the instance variables and methods of the super class but may add new instance variables and methods.
RMI	Remote Method Invocation. A J2EE RPC.
RPC	Remote Procedure Call. An external form of communication that allows a client to invoke a procedure in a server.
Scalability	The ability to expand as higher and higher volumes occur due to high volume operations with a parallel engine.
SDK	Software Developer’s Kit; Software Development Kit
SDLC	Synchronous Data Link Control. An IBM/SNA communications protocol. HDLC, high level data link control was derived using SDLC. SDLC manages synchronous (i.e., uses timing bit), codetransparent, bit-serial communication which can be duplex or halfduplex; switched or non-switched; point-to-point, multipoint, or loop.

Security Service	Compared to monolithic environments, distributed systems create new challenges for the implementation of security. Integrated systems must provide authentication, auditing, authorization, and encryption services that allow a client to conduct a secure communication with a server.
SGML	Standard Generalized markup Language. HTML and XML are subsets of SGML.
SMTP	Simple Mail Transfer Protocol. Documented in RFC 821, SMTP is Internet's standard host-to-host mail transport protocol.
SNA	IBM's Systems Networking Architecture provides a structure for transferring data between a variety of computing platforms.
SNMP	Simple Network Management Protocol. The Internet standard protocol, defined in STD 15, RFC 1157, developed to manage nodes on an IP network.
Service- Oriented Architecture	SOA is an architectural approach that presents a set of reusable software components that align with the agency's business goals and the Commonwealth's strategic objectives. The services are highly cohesive, loosely coupled, discoverable software components that are decoupled from hardware and network dependencies and that encapsulate the complexities of the underlying implementation.
SOAP	Object. A minimal set of conventions for invoking code using XML over HTTP
Sockets	Virtual connections between processes. They can be of two types, stream (bi-directional) or datagram (fixed length destination addressed messages). The socket library function socket () creates a communications end-point or socket and returns a file descriptor with which to access that socket. The socket has associated with it a socket address, consisting of a port number and the local host's network address.
SQL	Structured Query language. An industry-standard language for creating, updating, and querying relational database management systems.
STDL	Structured Transaction Definition Language. A high-level language for developing portable and modular transaction processing applications in a multi-vendor environment.
Store and Forward	A term used in message processing where a message is saved and then delivered.

Support for Standard Management Platforms	Management of large-scale distributed application environments requires appropriate tools. These tools should be based on standards (e.g., SNMP), so that the management of applications can be integrated with popular management platforms like OpenView in order to provide a consolidated picture of the state of network, operating system and application components.
Strategic	Rating category used in this document to rate integration technologies. This technology is considered a strategic component of the Commonwealth's Enterprise Technical Architecture. It is acceptable for current deployments and must be used for all future deployments.
Synchronous/ Connection Oriented Communi-cation	The implementation of a request/reply model for communication, i.e., the client program transfers data and control to the server with each call and it is blocked until a reply is returned.
TCP/IP	Transmission Control Protocol over Internet Protocol. 2) The TCP/IP Suite of protocols.
Technical Architecture	In enterprise architecture, business and technical computing specifications are considered. The technical architecture includes specification for only technical dimensions or components. In Virginia's enterprise architecture, the technical domains include: integration, security, platform, networking and telecommunications, application, database, enterprise systems management, and information architecture.
TP	Transaction Processing Monitor
Transaction Service	Guaranteed "all-or-nothing" execution of update requests against multiple (heterogeneous) resources.
Transitional	Rating category used in this document to rate integration technologies. This technology is not consistent with the Commonwealth's Enterprise Technical Architecture strategic direction. Agencies may use this technology only as a transitional strategy for moving to a strategic technology. Agencies currently using this technology should migrate to a strategic technology as soon as practical. A migration or replacement plan should be included as part of the Agency's IT Strategic Plan. New deployments or procurements of this technology require an

Triggering	Application components are dispatched automatically based on a predefined event condition. The definition of the event concept varies between the different types of integration technologies, e.g., request for certain elements in a database, arrival of a message in a queue, or method invocation request for an object that is managed by an ORB.
URL	Uniform Resource Locator. An address, usually for locating Web pages. (e.g., FTP//: abc.org). The part before the first colon specifies the access scheme or protocol. Commonly implemented schemes include ftp, http (World-Wide Web), gopher or WAIS. The "file" scheme should only be used to refer to a file on the same host. Other less commonly used schemes include news, telnet or mailto (email). The part after the colon is interpreted according to the access scheme. In general, two slashes after the colon introduce a hostname (host:port is also valid, or for FTP user:passwd@host or user@host). The port number is usually omitted and defaults to the standard port for the scheme, e.g. port 80 for HTTP.
VIM	Lotus/IBM CC:Mail.
VITA	The Virginia Information Technologies Agency. An agency of Virginia state government that is the Commonwealth's new consolidated, centralized information technology organization. VITA's responsibilities fall into three primary categories: Operation of the IT infrastructure, Governance of IT investments, and Procurement of technology.
Web services	A standardized way of integrating Web-based applications using open standard interfaces over an Internet protocol backbone. Used for businesses to communicate with each other and with clients, Web services allow organizations to communicate data without intimate knowledge of each other's IT systems behind the firewall.
X.400	International Telegraph and Telephone Consultative Committee (CCITT), now known as the ITU Telecommunication Standardization Sector completed the first release of the X.400 message handling system standard. The standard provided for the exchange of messages in a store-and-forward manner without regard to the user's location or computer system.

X.500	An ISO OSI Directory Service with an information model, a namespace, a functional model, an authentication framework, and a distributed operation model. X.500 directory protocol is used for communication between a Directory User Agent and a Directory System Agent. To allow heterogeneous networks to share directory information, the ITU proposed a common structure called X.500. However, its complexity and lack of seamless Internet support led to the development of Lightweight Directory Access Protocol (LDAP), which has continued to evolve under the aegis of the IETF. Despite its name, LDAP is too closely linked to X.500 to be "lightweight".
X.509	Standards for PKI or Public Key Infrastructure (e.g., Digital Signatures)
X/A	An application program interface (API) specification between a global Transaction Manager and Database.
XSL	Extensible Stylesheet Language
XML	Extensible Markup Language
XML Schema	XML Schemas express shared vocabularies and allow machines to carry out rules made by people. They provide a means for defining the structure, content and semantics of XML documents.

The domain team would like to thank their counterparts in the many states and federal government agencies whose excellent work preceded this. We couldn't have completed this report as quickly as it was done without the tireless energies obviously expended to complete their ETA documents. We also hope that other states will find this document useful in the design and updating of their own Enterprise Architecture. Significant contributions, references, and insights were derived from the following documents and web sites.

FOLDOC, the Free On-line Dictionary of Computing for references, links, and definitions: http://foldoc.doc.ic.ac.uk/foldoc/

What Is? For a verity of highly recommended resources, links, and definitions: http://whatis.techtarget.com/

North Carolina Statewide Technical Architecture Lexicon: http://www.ncsta.gov/docs/Lexicon/Lexicon%20-%20Terms%20and%20Phrases.pdf

The World Wide Web Consortium: http://www.w3.org/

Network Computing, CMP Media LLC: http://www.networkcomputing.com/

The Internet Engineering Taskforce: http://www.ietf.org/home.html

The Open Group: http://www.opengroup.org/

Appendices

Appendix A: History of the Evolution of Separate Integration Services

The term integration is a term in transition in the definition of computing architectures; in the 1990s similar technology was known as middleware. As a number of middleware services became incorporated into products or separate technologies, the term integration appears to more accurately describe various services utilized in today's diverse network environment. In the more homogeneous mainframe computing environments of the 60's, 70's and 80's, programmers did not generally have to deal with the complexities of local and wide area networking, logical partitioning of applications, or applications running on multiple physical platforms. Less complex environments made communications simpler. Programmers knew where data resided, used reports or terminal screens as the primary user interface, issued simple function calls for remote processing, and employed simple operating system commands when necessary. They conducted most of their computing work inside the controlled world of a unified mainframe or mainframe environment.

When local area networks (LANs) began to proliferate in the late 80's and 90's, two-tiered client server applications became commonplace. The computing environments were often characterized by fat client workstations interacting with multifunctional database servers. The servers had internal operating architectures that were much like the mainframe—architectures that provided simple integrated solutions for most of the communication needs between the client workstation and the database server over the LAN. The database systems also provided the application languages, metadata stores, internal integration solutions, and other tools. Similar self-contained, stovepipe applications for massive transaction processing remained on the mainframes. During this period, mainframe applications were generally viewed as legacy systems that should be converted to the new database system or replaced by shelf ware systems. Major enterprise off-the-shelf systems (e.g., PeopleSoft and SAP R/3) sometimes used integration functions defined within a database environment or created their own built-in network communications functionality. Programmers using the databases or shelf ware had to learn new methods, but generally were still somewhat shielded from the network communications and environmental complexities.

The two-tiered applications and client-to-database server communications quickly evolved to three or more tiers when applications became Web-enabled and server farms dotted the landscape. Internet connectivity, e-business, corporate mergers and buyouts, and internationally distributed businesses began to define the heterogeneous mix of application types, databases, application languages, and shared services in even more complex, distributed networked environment.

Databases, mainframes, and complex applications all provided redundant pieces of the communications between and among applications and databases. The computing environments had no central controller of all high-level communications. IT managers were faced with finding solutions that would integrate the high-level communications in their increasingly heterogeneous environments. Of extreme importance was simplifying the environment from the viewpoint of programmers who had different integration solutions with every application.

The growth of Internet and e-business played a very significant role in changing attitudes. One big change was that mainframe and legacy applications became more accepted as a permanent part of the computing landscape. Another, perhaps more important change was that a large proportion of existing systems potentially needed to provide data and services to customers over the Internet. Many new interfaces were needed.

Having to separately program so many new database and application interfaces in these multi-tiered, distributed environments would have been a very wasteful approach. The marketplace responded to the apparent coordination void by creating bundles of integration services. While integration in a number of instances was not separately tracked or monitored, the impact of integration's transition into the Web Service specifications (WS-) and the incorporation into Enterprise Application Integration processes has impacted the role and place of integration technologies as a separate and distinct service or product.

Historically, transaction processing integration technology had been around for a long time. It migrated from mainframes to the client server world and often formed the base layer of the "messaging plus" integration solutions. Some bundled services were designed to help specific legacy systems to interface with non-mainframe, networked applications. Some products specialized in communications among disparate databases. Others focused on providing a comprehensive solution for object-oriented services. Each provided some tools for building and accessing connectivity services that could be shared.

The Enterprise Services Bus (ESB) is a more recent phenomena in the marketplace. The ESBs bundle a variety of integration services from two or more vendors (e.g., transaction processing from one vendor and object brokering from another vendor) along with additional environment management functionality into one product. Essentially, some integration providers offer a suite of services that envelop the high-level communication needs of an entire distributed, multi-platform environment. Super services try to return central control to interoperability and location transparency tasks by providing all of the brokering, monitoring, metrics, protocols and communication services needed in a particular computing environment. In offering these services, integration providers assume that legacy systems would continue, and multiple standards would be in use (e.g., in a government setting, the mixed environment might be within an agency or across several agencies with a common system). Other integration providers focus on the enterprise integration aspects (e.g., enterprise application integration services [EAI], transaction processing, or Web enablement needs, e.g., comprehensive e-business solutions).

One important issue regarding acquiring integration technologies is that agencies must understand that they may be making some serious architectural decisions in choosing product. Agencies must understand all of the underlying components and how they work together (or do not), in order to make architecturally sound decisions. The advent of the Web Service specifications (http://www.w3.org/2002/ws/) has incorporated a number of functions and features separately defined as integration services. So, the integration decisions made related to one application may seriously limit future options for providing a comprehensive, coordinated approach to integration services and functions. Agencies need to know what all the pieces are and what they do. It is the role of enterprise architecture to make accessible and public the use of scalable and sustainable architecture for implementing application integration as befits the Commonwealth of Virginia IT goals. Agencies also need to know how well coordinated the setup is, from component to component and with the existing architecture. Understanding integration technology from only a user perspective (a programmer's viewpoint) or from only a "problems addressed" perspective is not enough. The decision makers must dig into the details to see the big picture, hence the enterprise view of integration business components.

Appendix B: What Communication Services Are Integration Domain Service: The OSI and TCP/IP Models

Virginia's EWTA includes a Networking and Telecommunications Domain and an Integration Domain. The Open Systems Interconnect (OSI) reference model provides a vehicle for explaining what network communications services are covered in Virginia's networking and telecommunications domain and what communication services are covered in its integration or enterprise systems management domains. A particular integration product may or may not cover all of the services that are to be discussed here.

The OSI Model is a seven-layer model used to describe what may take place in a particular network communication. (Note: not all communications require all seven layers of functionality.) The networking and telecommunications domain team and the integration domain team have mutually decided that functions in OSI reference layers 1-4 will be addressed primarily by the networking and telecommunications domain team and functions in layers 5-7[26], primarily by the integration domain team. A brief explanation will help.

Table 4 provides a picture of client server communications using the OSI reference model. Within a client server communication such as "the client mail application requests 'get new mail' from mail server application," the client performs steps 7 through 1, as needed, to *send the request* and the server performs the same steps in reverse order to *receive the request*. The original point of this division of communication tasks into layers was so that one vendor could provide the functions of one layer and another vendor could know how to interface with the neighboring layer and what tasks to perform at their layer. The model will never be fully implemented, especially for layers 5-7. Nevertheless, the model does provide a useful vehicle for drawing functional lines in an enterprise architecture.

A couple of points are important for understanding the role of integration technology in Virginia's architecture:

First, Virginia's networking and telecommunications domain architecture recommends a standard interface between network layers and integration layers. The interface protocols are TCP/IP.

Second, OSI layers 5-7 tend to be implemented as functional stacks rather than as layers. The TCP/IP stack contains only one layer above the transport layer. This is a more accurate picture of how high-level communication functions are actually implemented. Examples of integrated high-level functions are security functions for digital signatures and email functions.

Third, there are exceptions to the division between Virginia's network and integration layer. These exceptions are most likely to surface when vendors are providing a workaround for a proprietary protocol. The integration solution may use "tunneling" or hiding of one protocol inside another (layers 2 and 3) or may employ a different transport protocol (layer 4).

Fourth, some services performed by integration products are not communication functions, but are instead environment management and integration functions such as providing a GUI interface for building the workflow components, directory services, etc.

[26] In actuality, Virginia is recommending that the TCP/IP (5 layer) protocol stack be implemented as a standard with directory services and security services following open protocols. Where middleware is concerned, packetizing belongs to the networking and telecommunications domain and higher functionality to the integration domain and the enterprise systems management domain.

Table 5: OSI Seven Layer Model Showing a Communications Flow

Client Application **(e.g., Sender)**		**Server Application** **(e.g., Receiver)**
OSI 7 Application .... ▼		OSI 7 Application .... ▲
OSI 6 Presentation .... ▼		OSI 6 Presentation .... ▲
OSI 5 Session .... ▼		OSI 5 Session .... ▲
standard interface		*Standard interface*
OSI 4 Transport .... ▼		OSI 4 Transport .... ▲
OSI 3 Network .... ▼		OSI 3 Network .... ▲
OSI 2 Data Link .... ▼		OSI 2 Data Link .... ▲
OSI 1 Physical.............. ▼	Signal over medium .... ▶	OSI 1 Physical .... ▲

The Network Layers in Brief

The four OSI layers that define networking are as follows. Layer 1 is the specifications for the physical layer (e.g., network wiring or other media). Note that the physical layer could be copper or fiber or air (wireless) and still communicate with the data link layer, which is layer 2. Ethernet methods of making sure that only one communication is taking place at a time on the physical medium is an example of this layer. The 3rd layer is called the network layer. One thing it deals with is protocols like IP addressing to locate a client or a server when more than one network is involved. The 4th layer is the transport layer, and it uses protocols like TCP to packetize a communication and to ensure the packets are transported properly. The important thing to note here is that network layers are rarely used to provide business application related services. They move communications from one place to another.

The Integration Layers in Brief

The integration layers in the OSI model are called session (layer 5), presentation (layer 6), and application (layer 7). An example of a session service is providing a one-way communication or tracking a two-way transmission. Presentation services include encryption for security, language translations (e.g., from ASCII to EBCDIC), and compression/decompression so that the application programs and operating systems will understand the communication when it is received. The application layer is not the business application program (e.g., email read and write program), but rather, the service that knows when an application wants to communicate with the network (e.g., that the email program is issuing a "file send" command).

What is important to note is that the integration layers deal with *business functionality* related to transmissions between network clients and servers (e.g., delivering email or accepting an encrypted password to authorize access). The application program has to ask for the integration layer functionality, but the programmer should not have to know how it is provided. Good integration products should shield the programmer from these complex details.

The Arizona Enterprise Architecture team has identified target and emerging technologies for OSI layers five through seven for the integration layers. They are shown in Table 5 below. [27]

Table 6: Arizona Enterprise Architecture Target Technology Table

Arizona Enterprise Architecture Target Technology Table March 1, 2008	
Target (Strategic)	**Emerging**
OSI Layer 1 – Physical	
Network	
Category 6 UTP Infrastructure less, Jini technology-50/125-micron multimode fiber, 10/125 micron single mode fiber Ultra-wideband (UWB) Structured cabling systems, based on TIA/EIA 568, 569, 606, 607 standards and applicable electrical codes Intra-building Wireless: IEEE 802.11 WLAN Logical star or mesh topology. Logical meshed topology	mobile “ad-hoc” service-based networking transmission
Security	
Keys, locks, badges, cameras, access logs, Biometrics controlled access systems. IP-based control systems	access
Platform	

[27] Arizona Data Information Architecture. http://www.azgita.gov/enterprise_architecture/AZ_EA_Target_Technology_Table.htm

Arizona Enterprise Architecture Target Technology Table March 1, 2008	
Target (Strategic)	**Emerging**
SCSI, iSCSI	Trusted Platform
Single application smart cards	Multi-function smart cards
OSI Layer 2 – Data Link	
Network	
Open, standards-based, multi-service networks 100 Mbps/1 Gbps/10 Gbps IEEE 802.3 Ethernet Wireless: IEEE 802.11 WLAN, IEEE 802.16 WMAN, IEEE 802.15 WPAN 40 Gbps IEEE 802.3 Ethernet Resilient Packet Ring (RPR), SONET, Switched LAN technology IEEE 802.1p/Q QoS, Diffserv, RSVP, VLAN, IEEE 802.3af PoE	Emerging packet- and cell-based wireless and satellite protocols
Security	
Media Access Control (MAC) Access Control Lists (ACLs) VPN, RADIUS Intrusion detection, vulnerability scanning Wireless: IEEE 802.11i, WAP, PEAP w/ IEEE 802.1x	
OSI Layer 3 - Network	
Network	
IPv4, IPv6, Mobile IP Routing Technologies: BGP, OSPF, IS-IS, MPLS, IGMP, PIM, MBGP DHCP Converged networks with QoS, prioritization, and traffic flow control for all services, switched, multisegment design Multi-layer switching Layer 3, wire-speed, network-level switching and prioritization	
Security	
Integrated firewalls - Packet filtering, ICMP Boundary/perimeter Routers, end-point security, static NAT, IPSec End point security – individual firewalls	
OSI Layer 4 - Transport	
Network	
TCP, UDP Wireless: WDP, Wireless Profiled TCP RTP, RTCP Converged networks with QoS, prioritization, and traffic flow control for all services Layer 4, wire-speed, transport-level switching and prioritization	

Arizona Enterprise Architecture Target Technology Table March 1, 2008	
Target (Strategic)	**Emerging**
Security	
Integrated firewalls - stateful inspection, dynamic NAT SLL VPN[28] SSL, SSH, TLS Wireless: WTLS	
OSI Layer 5 – Session	
Network	
DNS Wire-speed, intelligent, session-level	switching and prioritization
OSI Layers 6 – Presentation, 7 – Application	
Network	
SNMP, RMON Wire-speed, intelligent, content-level prioritization	switching and H.323, SIP with SDP, SAP, RTSP
Security	
Integrated firewalls - Application-proxy gateway, Proxy Servers, Dedicated Proxy Servers FTP, S/MIME for mail servers Encryption Technologies: PKI, OpenPGP, AES, 3DES, DSS Smart cards, Kerberos Role-based administration, permissions, and rights	AVDL[29]
Digital signature, Public Key Certificates, PKI Multi- Enterprise directory services - LDAP meta- with an OID tree Virus/malicious code protection software Mobile Firewalled DNS, with services placed on DMZ	function smart cards directory agents
Standards-based platform sign-on with role-based administration Identity federation across web-based Industry-standard and vendor-neutral APIs for identification Strong password policy FIPS 140-3	Single sign-on across platforms, session applications, sessions
Token-based identification Human Authentication (HA-API)	API
	Web Services: WS-Security including Security Profile, Policy (QoS), Privacy, Secure Conversation, Trust, Authorization, Policy Assertions, Security Policy, Federation, Attachments
Platform	

[28] The most important distinction is what each SSL VPN gateway presents to clients: how it communicates with clients, what applications it enables, and how it secures those applications. Almost all suppliers (with one exception) support reverse-proxy mode for authenticating VPN sessions through Web browsers, but some vendors don't extend reverseproxy support to older browsers or operating systems - which could cause compatibility problems for certain SSL VPN applications, such as those targeting e-commerce.

[29] The Open Group's Distributed Computing Environment (DCE) maintains the LDAP standard. For a guide to additional information on LDAP and related standards work, see http://www.opengroup.org/directory/, the Directory Interoperability Forum.

Arizona Enterprise Architecture Target Technology Table March 1, 2008	
Target (Strategic)	**Emerging**
Security	
Integrated firewalls - stateful inspection, dynamic NAT SLL VPN[30] SSL, SSH, TLS Wireless: WTLS	
OSI Layer 5 – Session	
Network	
DNS Wire-speed, intelligent, session-level	switching and prioritization
OSI Layers 6 – Presentation, 7 – Application	
Network	
SNMP, RMON Wire-speed, intelligent, content-level prioritization	switching and H.323, SIP with SDP, SAP, RTSP
Security	
Integrated firewalls - Application-proxy gateway, Proxy Servers, Dedicated Proxy Servers FTP, S/MIME for mail servers Encryption Technologies: PKI, OpenPGP, AES, 3DES, DSS Smart cards, Kerberos Role-based administration, permissions, and rights	AVDL[31]
Digital signature, Public Key Certificates, PKI Multi- Enterprise directory services - LDAP meta- with an OID tree Virus/malicious code protection software Mobile Firewalled DNS, with services placed on DMZ	function smart cards directory agents
Standards-based platform sign-on with role-based administration Identity federation across web-based Industry-standard and vendor-neutral APIs for identification Strong password policy FIPS 140-3	Single sign-on across platforms, session applications, sessions
Token-based identification Human Authentication (HA-API)	API
	Web Services: WS-Security including Security Profile, Policy (QoS), Privacy, Secure Conversation, Trust, Authorization, Policy Assertions, Security Policy, Federation, Attachments
Platform	

[30] The most important distinction is what each SSL VPN gateway presents to clients: how it communicates with clients, what applications it enables, and how it secures those applications. Almost all suppliers (with one exception) support reverse-proxy mode for authenticating VPN sessions through Web browsers, but some vendors don't extend reverseproxy support to older browsers or operating systems - which could cause compatibility problems for certain SSL VPN applications, such as those targeting e-commerce.

[31] The Open Group's Distributed Computing Environment (DCE) maintains the LDAP standard. For a guide to additional information on LDAP and related standards work, see http://www.opengroup.org/directory/, the Directory Interoperability Forum.

Arizona Enterprise Architecture Target Technology Table March 1, 2008	
Target (Strategic)	**Emerging**
Platforms having open industry-standard operating systems (OS), with imbedded security, and open standard interfaces and drivers	Platforms having open industry-standard OS, with imbedded security, multifactor authentication, intelligent I/O, and open-standard interfaces/drivers
DMI	CIM
Platforms having industry de facto standard OS, with imbedded security, and open-standard interfaces/drivers. For example: ○ Mainframes with TCP/IP, SIP, Open APIs ○ Servers with TCP/IP, SIP, Open APIs ○ IP telephony with TCP/IP, SIP, H.323, ISDN PRI, open APIs, standard MOS codecs ○ Hybrid IP telephony (TDM/IP) systems with TCP/IP, SIP, H.323, ISDN PRI, open APIs, standard MOS codecs ○ Network Attached Storage o Direct Attached Storage ○ Storage Area Networking with multi-use access channels ○ End user (client) devices (PCs, Network Computers, PDAs, etc.) with wired/wireless connectivity, TCP/IP and multi-function applications	
Platforms having niche proprietary OS, with imbedded security, and open-standard interfaces and drivers (requires exceptional business requirements)	
SNMP management of platforms	
Platforms deployed on target networks, with class of service (CoS) and quality of service (QoS)	
Software	
n-tier distributed software applications emphasizing client (State employee, community of interest, public customer) productivity and performance enhancements and enablers (decision-making at the appropriate level) through self-service, self administration, etc., utilizing browser-based (HTTP, HTTPS) client access deployed on Target Platform Architecture server, storage, and client devices	Software applications hosted via ASPs
Traditional, monolithic State software applications with web-enabled, browser-based (HTTP, HTTPS) client access	
Three-tier distributed software applications with access to n-tier architecture services C++, Java™, Visual Basic®, etc.	

Arizona Enterprise Architecture Target Technology Table March 1, 2008	
Target (Strategic)	**Emerging**

Java™ and servlet software, COM™, DCOM™ CORBA, ORB, ISO/IEC 11179 Open API Integration: TPM, RPC, RMI, JMS, MOM	Object-oriented software IIOP ESB
EJB™ server-side deployment, COM+ HTTP, HTTPS Web Services: open, industry-std., XML, DSML, SOAP, SAML, JSP, ASP, JNDI, J2EE™,.NET, BizTalk, HTML, XHTML	Web Services: open, industry-std., WSDL, UDDI initiatives, BPEL, WS-Coordination, WS-Transaction, XQuery, XMLA, WSUI. WSRP, RDF, EbXML secure exchange of information, UML™, XSL, CSS3, XSLT
Wireless: WAP, WAE, WSP, WTP, WML, XHTMLMP, CSSMP, VoiceXML, wireless profiled HTTP	J2ME™ for resource-constrained mobile networking
GUI presentation layer access to software as a precursor to browser-based (HTTP, HTTPS) access Browser-based (HTTP, HTTPS) access to software	Portal-based universal browser access to all services
Software applications that are manageable with SNMP-based management tools	SOA
LDAP directory services	Enterprise federated management
Software application security	Enterprise LDAP directory services, WBEM, DEN, CIM
RDBMS	OODBMS, ORDBMS
Open database connectivity: SQL, ODBC, OLE DB, NDMP, NFS, CIFS, JDBC	
Database integration that uses open database connectivity	
Email services: SMTP, S/MIME, IMAP4, POP3	Enterprise email directory services
Productivity software with open APIs	Productivity software conforming to IETF standards such as iCalendar, CAP, IPP, etc.

Appendix C: Refference and Links

State Sites:

Connecticut Middleware Domain Architecture
http://www.ct.gov/doit/lib/doit/Middleware_Architecture_ver_2.0.pdf

North Carolina Systems Integration Architecture:
http://www.ncsta.gov/docs/Principles%20Practices%20Standards/System_Integration.pdf

Massachusetts Enterprise Technical Reference Model - Version 3.5
http://www.mass.gov/?pageID=itdsubtopic&L=5&L0=Home&L1=Policies%2c+Standards+%26+Legal&L2=Documents+by+Type&L3=Enterprise+Technical+Reference+Models&L4=Enterprise+Technical+Reference+Model+-+Version+3.5&sid=Aitd

Arizona Enterprise Architecture: http://www.gita.state.az.us/enterprise_architecture/

Federal Links:
Links: Federal Information Technology Architecture
http://www.gsa.gov/Portal/gsa/ep/channelView.do?pageTypeId=8199&channelPage=%252Fep%252Fchannel%252FgsaOverview.jsp&channelId=-13315

General Integration References *GartnerGroup:*

"Market Trends: Application Integration, Middleware and Portal Software", EMEA, 2004-2009, G00137095, 19 December 2005

"Hype Cycle for Application Integration and Platform Middleware", 2005, G00127756, 18 July 2005

"Integration Suite and ESBs: Integration Technology for the Mainstream", Gartner Application Integration and Web Services Summit, 5 – 7 December 2005, Orlando, FL

"Data Services: The Intersection of Data Integration and SOA", G00130332, 20 September 2005

"Data Integration Is Key to Successful Service-Oriented Architecture Implementations", G00130871, 12 October 2005

"The ICC and SOA Governance", G00137440, 3 February 2006

"Technical Approaches to and Considerations for SOA Governance", G00143592, 26 October 2006

Anthony Bradley, "Reference Architecture for Enterprise 'Mashups", G00143592, 7 September 2007

Anthony Bradley, "Mashups' and Their Relevance to the Enterprise", G00519906 7 September 2007

Anthony Bradley and David Gootzit, "Who's Who in Enterprise Mashup

Technologies", G00519910 7 September 2007

Anthony Bradley, "Key Issues for Enterprise 'Mashup' Practices, Technologies and Products", G00619115 10 March 2008

Anthony Bradley and David Gootzit, "IBM Enterprise Offering Brings Mashups Closer to Mainstream", G00645307 ***IDC International Data Group***

"What is Middleware?" IDC #34362, Volume: 1, November 2005
http://www.idc.com/getdoc.jsp?containerId=34362

NASCIO

The National Association of State Chief Information Officers (NASCIO):
Research Brief "Connecting the Silos: Using Governance Models to Achieve Data Integration", June 2005
https://www.nascio.org/nascioCommittees/interoperability/connectingSilos.pdf

"Government Information Sharing: Calls to Action" March 2005
https://www.nascio.org/publications/index.cfm

Other Sources:

"Middleware Vendor Database: Middleware Spectra, an independent resource on business integration and network computing through middleware and message brokering", http://www.middlewarespectra.com/abstracts/vendordb.htm#6

"Enterprise Wide Information Technology Architecture (EWITA) links and resources", http://www.ewita.com/

"ISG Library prepared by International Systems Group, Inc.", http://www.isg-inc.com/goodies.htm

Special Topics

XML and SOAP

"XML Database Products, by Robert Bourret", http://www.rpbourret.com/xml/XMLDatabaseProds.htm

"XML Key Management Specification (XKMS 2.0), W3C Recommendation 28 June 2005", http://www.w3.org/TR/xkms2/

"SOAP Version 1.2 Part 0: Primer, W3C Recommendation 24 June 2003", http://www.w3.org/TR/2003/REC-soap12-part0-20030624/

Object Models

"Document Object Model (DOM)", http://www.w3.org/DOM/

Web services

"Web Services Overview by W3C", http://www.w3.org/2002/ws/

"Web Services Description Language (WSDL) 1.1, W3C Note", 15 March 2001 http://www.w3.org/TR/wsdl

"Improve Your SOA Project Plans by IBM", http://www-128.ibm.com/developerworks/webservices/library/ws-improvesoa/

Eric Newcomer, Greg Lomow, *Understanding SOA with Web Services*, Addison Wesley (2005)

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