GartnerConsulting

CJIS Architecture

Prepared on Behalf of

MECKLENBURG COUNTY

March 31, 2000

GartnerGroup

3190 Fairview Park Drive, Falls Church, VA 22042

Table of Contents

1. CJIS ARCHITECTURE 1
1.1 Application Architecture 2
1.2 Integration Architecture 3
2. APPLICATION SOFTWARE ARCHITECTURE 5
2.1 Main Application Architecture 5
2.2 Software Architecture 6
2.3 Imaging and Workflow 8
2.4 Internet Integration 9
2.5 Integration with Other Applications 9
2.6 Summary of Application Architecture Elements 12
3. TECHNICAL ARCHITECTURE 13
3.1 Platforms and Operating Systems 13
3.2 Network Infrastructure 15
3.3 Internet Architecture 16
3.4 Query, DSS and OLAP Tools 17
3.5 Imaging and Workflow Management Software 17
4. VENDOR / PRODUCT SELECTION 18
4.1 Message-Oriented Middleware 18
4.2 Message Broker 18
APPENDIX A – IT PRINCIPLES 21

Entire contents © 2000 Gartner Group, Inc. All rights reserved.

1. CJIS ARCHITECTURE

This document specifies the desired technical architecture for reengineered applications and application components comprising the new Criminal Justice Information System (CJIS).

The CJIS Integration Project is an initiative that involves State, County and City agencies that represent the Criminal Justice Community (CJC). The new CJIS will include both reengineered applications and integration with existing applications that run in a variety of environments. New applications may (a) be developed on a custom basis and/or (b) be commercial off-the-shelf (COTS) packages procured from independent software vendors (ISVs).

This section is a summary that describes the main technology areas from an architecture and infrastructure perspective. The expected benefits associated with this technical architecture include interoperability of platforms and systems, faster applications deployment, lower support costs, a common sense of direction and higher-quality systems.

The CJIS Architecture is aligned with Mecklenburg County's guiding principles regarding application development and the general use of related technologies (see Appendix A).

CJIS will use a distributed architecture built around a central Windows 2000 server cluster running the applications. Overall, the architecture can be characterized as a state-of-the-art client/server architecture that incorporates a three-tier component software architecture. This "thin-client" architecture is used in the latest versions of commercial products such as Peoplesoft and SAP R/3. It is swiftly supplanting the earlier two-tier application architecture as the best practice at major corporations. The architecture is built around principles such as:

- A "thin client" (software)
- A component-based, service-oriented architecture
- Application services executing on a Windows 2000 cluster (AppCenter Server)
- Microsoft DNA 2000 with MTS middleware
- A single relational DBMS for centralized application databases
- Special-purpose Web and query/report servers.

The figure below illustrates the architecture from an application perspective.

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

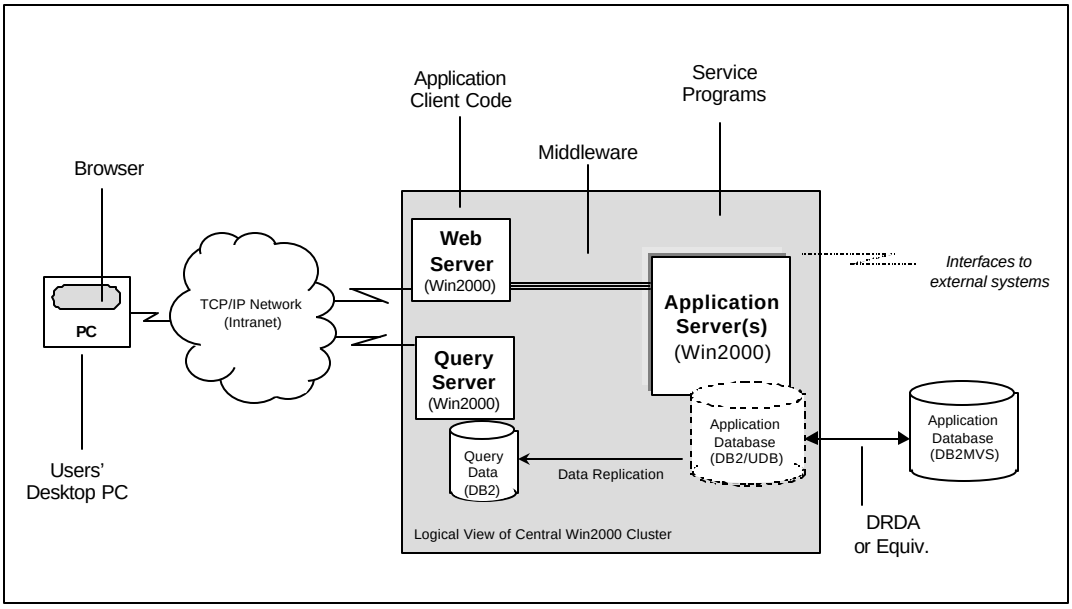


Figure 1-1. Distributed Client/Server Architecture

The above diagram shows the architecture from a logical view. It is independent of the physical configuration of the Intel hardware cluster.

1.1 APPLICATION ARCHITECTURE

Applications will be designed using a "component-based, service-oriented architecture," as illustrated below.

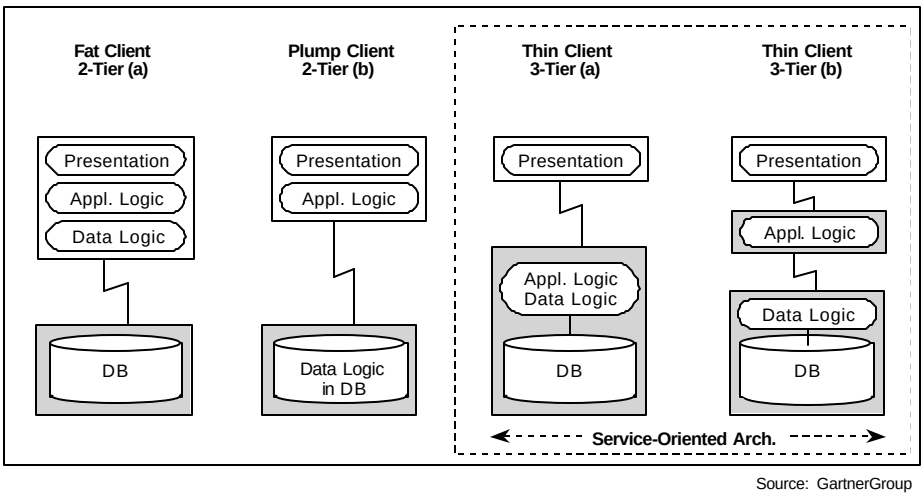


Figure 1-2. Component Architecture

The above diagram shows the partitioned program categories: presentation, application logic and data logic, and how they relate to each other.

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

1.2 INTEGRATION ARCHITECTURE

The new CJIS application must be able to integrate with a variety of existing systems. They are all important, but the Mecklenburg County and AOC applications are especially so.

- **Mecklenburg County** currently runs most mission-critical applications on an IBM OS/390 mainframe. These are traditional "legacy" applications programmed in COBOL with data in IBM DB2/MVS and/or VSAM files.

- The North Carolina **Administrative Office of the Courts** (the AOC) also runs most applications on an IBM mainframe. They are legacy COBOL applications with data in IBM DB2, VSAM or IMS databases. One application, the Magistrates' System, is programmed in Delphi with a MS SQL Server database.

- **Other agencies** – City, County and State – run on a variety of platforms.

"Middleware" is enabling software used to make applications work across multiple computers or let applications talk to one another. For instance, you need middleware to construct client/server applications or to access a single database from multiple computers.

DB2 database middleware will be used to enable the CJIS database to be hosted on the IBM OS/390 yet accessed by data logic components running on the Windows 2000 application server. This approach also permits future migration of the database off the mainframe with no program changes if this is ever required.

CJIS, and the Mecklenburg County legacy applications, will communicate via IBM MQ-Series, which is program-to-program middleware. Many state agencies, including the AOC, will also implement MQ-Series.

> Note: The AOC has been using low-level middleware and a non-standard software module ("ALI") that combines both middleware and application functionality. While the AOC's direction is MQ-Series, it may be necessary to integrate with ALI during an interim period.

A message broker will be used to unite different applications built with differing technology. Message broker technology has not yet been selected; the use of MQ-Series for the underlying message transport is a mandatory requirement.

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

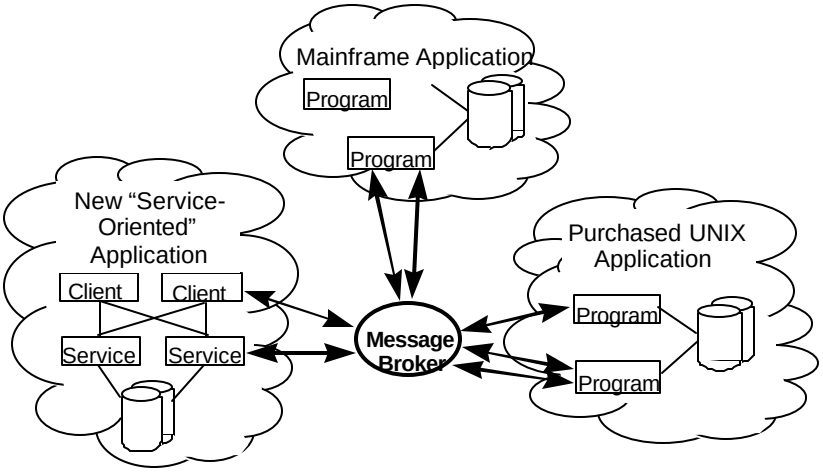


Figure 1-3. Message Broker

Note that the architecture calls for *both* a service-oriented architecture with synchronous middleware *and* a messaging approach for inter-application integration. This is appropriate; different architectural approaches should be taken based on the scope and nature of the task at hand. The two are complementary.

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

2. APPLICATION SOFTWARE ARCHITECTURE

This chapter specifies the CJIS application architecture. The primary focus is on the new application software and the "enabling" software used to construct it.

2.1 MAIN APPLICATION ARCHITECTURE

CJIS will be built to an advanced client/server architecture built around principles such as:

- A "thin client" software architecture
- A service-oriented architecture that emphasizes reusable components
- Application services executing on a centrally managed server complex
- The Microsoft MTS TP Monitor for transaction control, reliability and through-put
- A single central application database, which is relational and normalized
- Special-purpose Web and report servers.

The figure below illustrates the architecture from an application perspective. Client code is shown residing on the Web server; it is downloaded from there for execution on the desktop PCs. Internet access is not shown on this diagram.

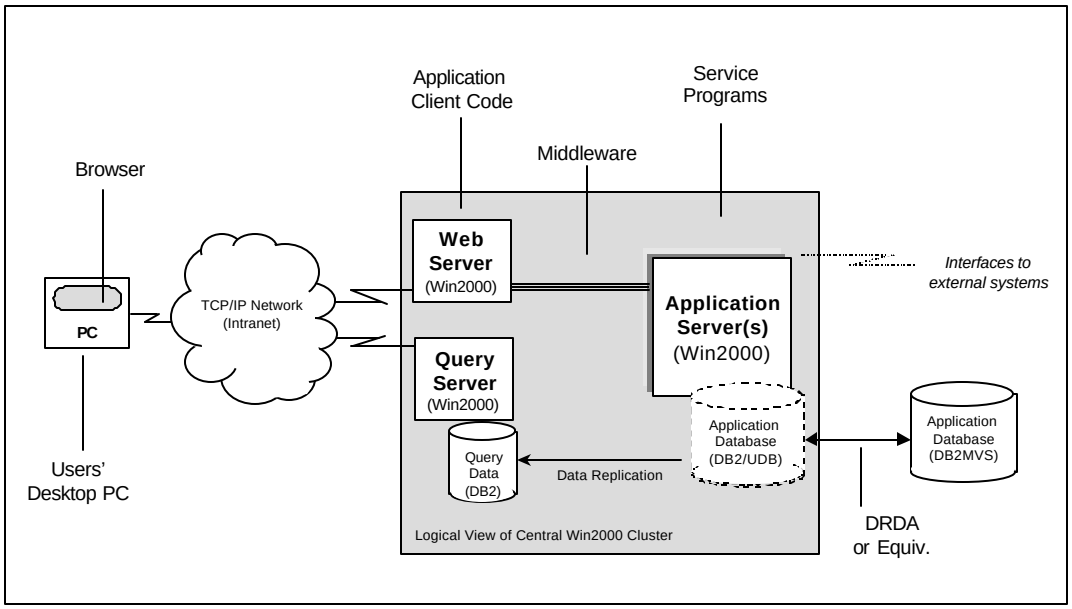


Figure 2-1. Application Architecture

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

2.2 SOFTWARE ARCHITECTURE

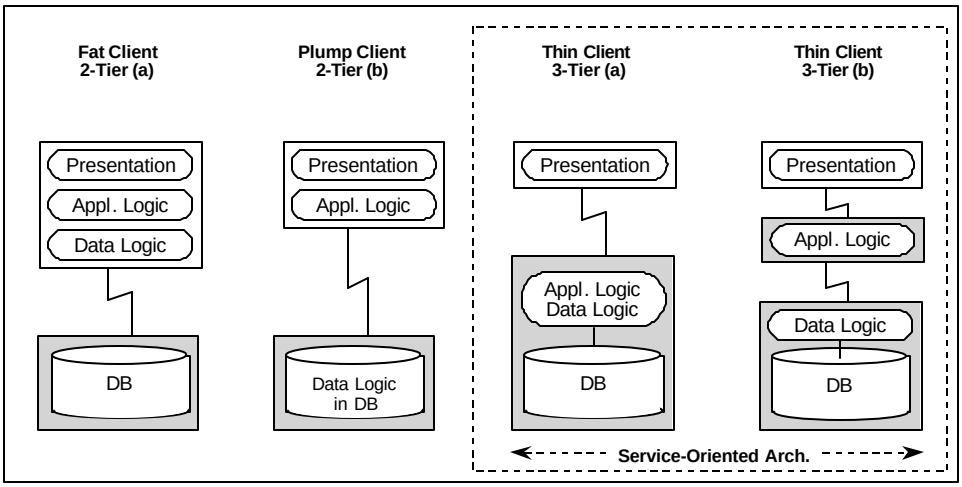
2.2.1 Fundamental Design Principles

Fundamental design principles include the following:

- Modularity
- Encapsulation
- Reuse/sharing of component services
- Separation of presentation logic from application (business) and data access logic
- Server-centric processing.

2.2.2 Service-Oriented Architecture

CJIS will use a "component-based, service-oriented architecture" as illustrated on the right side of the figure below.



Source: GartnerGroup

Figure 2-2. Application Partitioning

The above diagram shows the partitioned program categories: presentation, application logic and data logic, and how they relate to each other. But of course, an application is constructed with many implemented modules in each category. We can visualize an application constructed with a component-based architecture with the help of the next diagram:

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

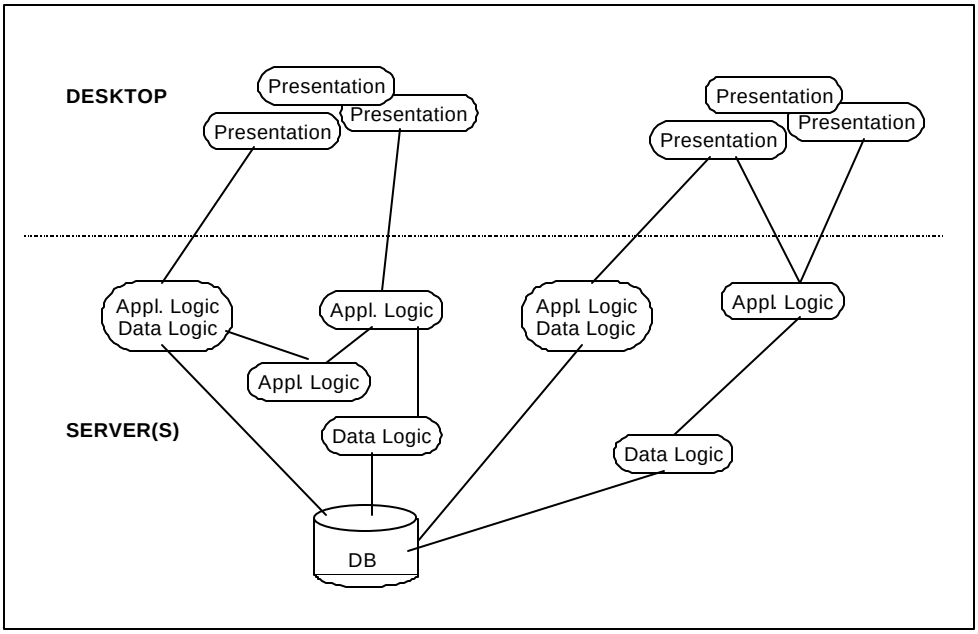
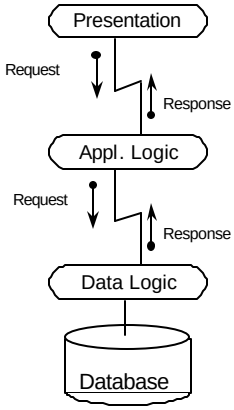


Figure 2-3. Application Components

Applications designed with an service-oriented architecture benefit from improved design scalability, modular development, testing, etc. The cost to develop and maintain is significantly lower.

The number of component services for a medium-large application might be, say, 400-800 application services and 200-300 data services.

2.2.3 Message-Based Model



The architecture also calls for a message-based design model. Message-based communication is conceptually simple. There is a perfect analogy to the real world where we send data "over the wire." Another reason the message-based model was chosen because it fits well with (a) Web clients, (b) Visual Basic components and (c) IBM MQ Series middleware.

Messages are usually paired: Request Message and Response Message. Messages can and should be re-usable. Messages will be defined as Visual Basic classes.

2.2.4 Component-Based

The architecture also calls for a component-based approach. IT design practices are being reshaped by the increasing use of components and a "black box metaphor." Related design principles are:

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

- Modularization—organizing applications into smaller modules
- Encapsulation—hiding data and logic from uncontrolled external access.

Modularization divides big problems into smaller "bite-sized" problems that are easier to solve.

Encapsulation implies that all communication between modules is done only through controlled, documented interfaces ("contracts"). Encapsulation has the following two benefits:

- It helps protect internal data and code from careless or malicious misuse (i.e., integrity protection)
- It helps shield external developers from the complexity and dynamism of the contents of a module.

External people who want to use an **encapsulated** module do not have to see or understand the logic or data model that is inside the module. The module is a "black box" that is accessed only through the defined interfaces. The reuse or sharing of a module implies reuse or sharing of both logic and data. By contrast, the direct reuse or sharing of data (as in data integration) means that one is not reusing any program logic.

The principles of modularization and encapsulation are applied on several levels:

- They are applied within a single application program via techniques such as component software.
- The same principles are applied between related application programs in the same application system or group of related application systems using a service-oriented architecture. Services are modules that contain a shareable set of business rules and data access logic that may be invoked by multiple client application modules.
- The principles of modularization and encapsulation can also apply on a macrocosmic level, using program-to-program messages (function transfer) that cross the boundaries of independently designed application systems. In this manner, production databases are encapsulated on a coarser level. They are accessed only by their respective native application programs so that edits, integrity checks and business rules can be reused.

2.3 IMAGING AND WORKFLOW

While these are not major elements, CJIS will need to be able to display images and use automated workflow management for some processes. The following guidelines apply:

- Images need to be compatible both with the web and HTML (implies GIF, JPEG or possibly TIFF) and Visual Basic clients.
- The enabling software for workflow needs to provide a tool for creating the “flows”, in addition to the run-time software.
- Only selected processes should be workflow-based. It is neither necessary nor desirable to drive every process this way. Simple processes should be kept simple.

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

2.4 INTERNET INTEGRATION

The Internet and/or a *Criminal Justice Community Extranet* will be used for:

1. File transfer: browser-enabled uploading & downloading (from other agencies, the legal community, etc.)
2. Information access, status inquiry from users and the general public
3. Browser-based (HTML) access to the CJIS application
4. E-mail, including attachments. (The CJIS Project may choose to block incoming attachments to prevent viruses. Authenticated users would use browser-based file uploads in lieu of sending attachments.)

Security is key, so the Internet Web server will be separated from the internal servers by a firewall. The following diagram illustrates the architecture:

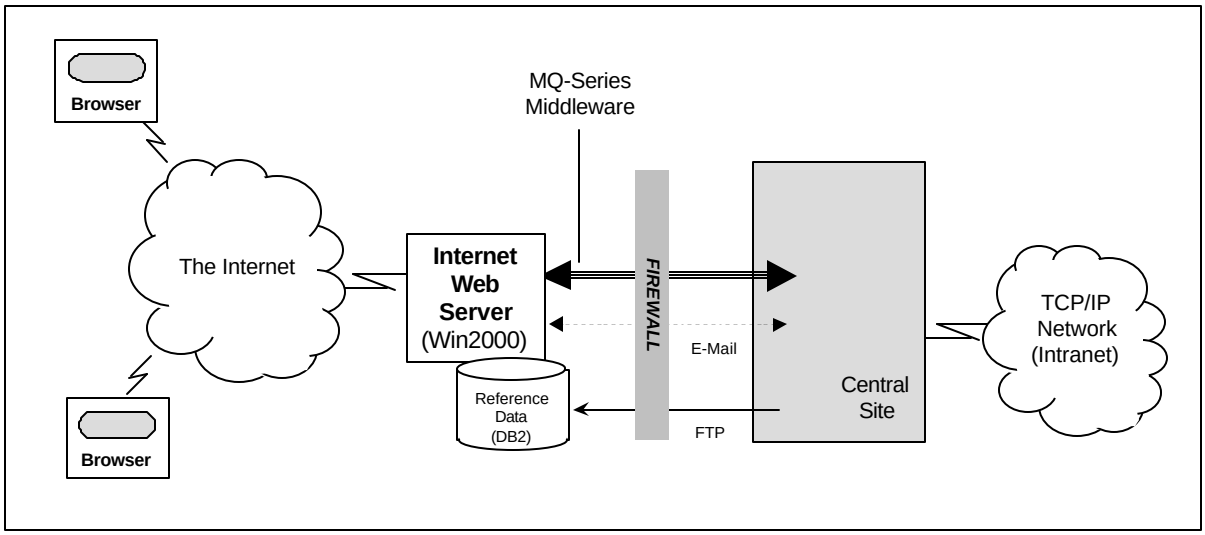


Figure 2-4. Internet Architecture

Notes:

(a) CJC Extranet access is not shown
(b) User file upload & download is via FTP from Browser to Web Server (not shown in above diagram).
(c) The elements shown are discussed in a table in the Technical Architecture.

2.5 INTEGRATION WITH OTHER APPLICATIONS

Message-oriented middleware and message brokers will be used. These two technologies are related. Message brokers are built on top of message-oriented middleware and offer extended capabilities.

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

2.5.1 Message-Oriented Middleware

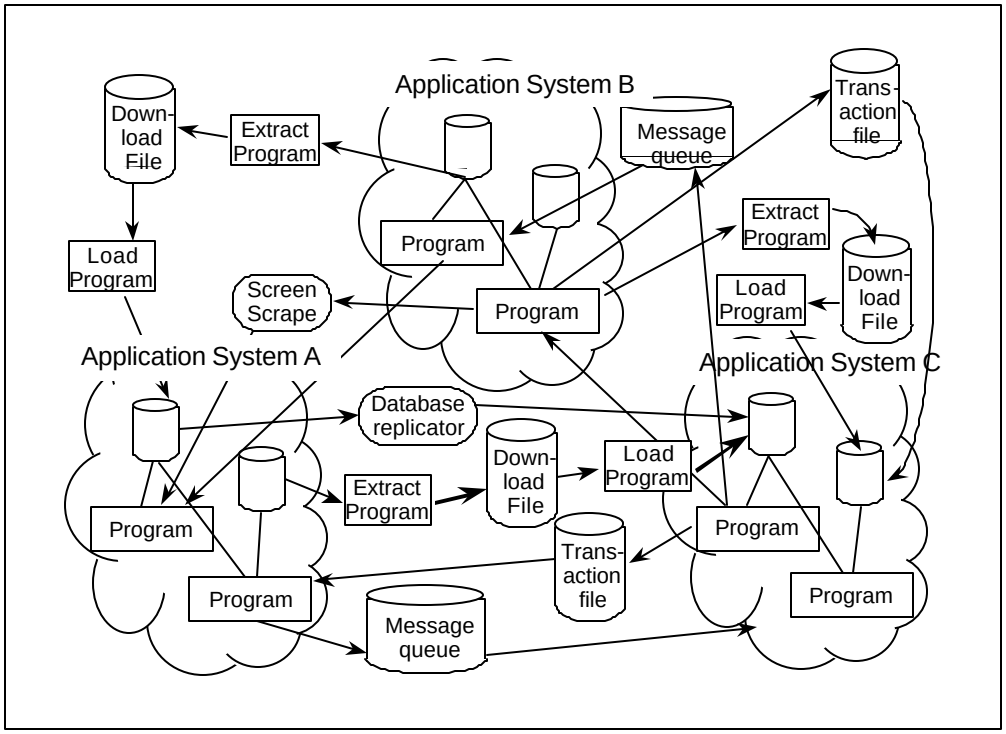
Asynchronous, message-oriented middleware will be used for near-time communication with other Mecklenburg County applications running on the IBM mainframe. It can also be used to communicate with other agencies' applications on various platforms (e.g., AOC, DHS or DMV applications), though a message broker is preferred (the architecture allows for design discretion).

IBM's MQSeries is the specified message-oriented middleware because (a) it is a leading product, (b) it has been selected by the AOC and (c) it is recommended for state agencies in general by the North Carolina IRMC.

Usage of MQSeries middleware for communication between the external (outside the firewall) Internet Web server and the main application server is also specified.

2.5.2 Message Broker

CJIS will use a message broker for integration with other agencies' applications. Message brokers are "application integration de-tanglers." A typical application integration diagram looks like the following:



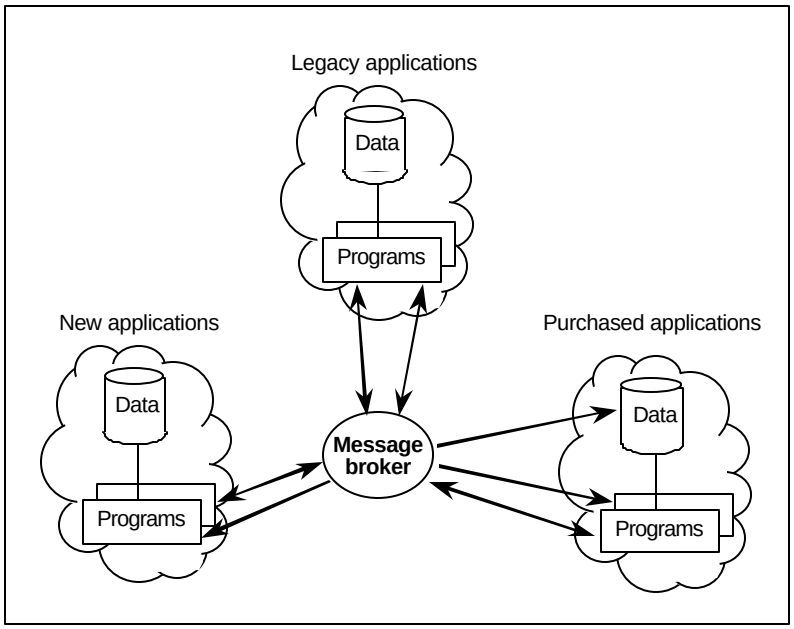
Source: GartnerGroup

Figure 2-5. Typical Application Integration Style

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

Why is this the typical pattern? Because choices made in developing one application (style, technology, etc.) cannot be imposed on another one. Applications are usually built at different times, by different teams, under different constraints.

However, CJIS *is an integration project* and must do better than the norm. A message broker is a better way to unite different applications built with differing technology.



Source: GartnerGroup

Figure 2-6. Recommended Application Integration Style

For CJIS purposes the key facilities that are provided by the message broker *externally from the application logic code* are:

1. Ability to route a message to another application or applications
2. Data transformation
3. Ability to store messages in a file for later transfer (e.g., via FTP)
4. Uses selected message-oriented middleware (MQ-Series) for message transport.

Note that the architecture calls for *both* a service-oriented architecture with synchronous middleware *and* a messaging approach for inter-application integration. This is appropriate; different architectural approaches should be taken based on the scope and nature of the task at hand. The two are complementary.

2.5.3 FTP, Tapes, etc.

For some integration needs, complete files may need to be transmitted via file-transfer protocol (FTP) or tape. Designers can choose between (a) periodic database extracts to build the transfer file or (b) logging of messages from the broker to a transfer file.

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

2.6 SUMMARY OF APPLICATION ARCHITECTURE ELEMENTS

The software architecture is summarized in the following table.

Topic	Approach
Thin Client	• Predominantly HTML forms, connected to the server using Visual Basic (VB) applets and/or JavaScript (MS JScript) and DNA 2000 middleware. • Extended clients that need a complex dynamic flow can be programmed in Visual Basic applets invoked from HTML menus. • External clients cannot use Visual Basic.
Client Software Distribution	• HTML and Visual Basic code automatically downloaded from Web server.
Component-Based, Service-Oriented Architecture	• Component services reside on a central server. Changes to business and data logic are made in one place only. Complex data logic routines and all routines that update the database are packaged as tier-three components. Simple data access may be combined with business logic into a single tier-two/three service. • The system should be primarily online. Services should not just queue up transactions for later update, but actually do the updates in most cases.
Development Language(s) and Tools	Languages: • Clients programmed in HTML (~90%) and Visual Basic (~10%). • Services programmed in Visual Basic. • Batch programs in Visual Basic (Windows 2000 server) or COBOL (on OS/390 server). Tools: • MS Visual Interdev (includes MS Front Page) used for HTML authoring. • MS Site Server (includes IIS -- Internet Information Server) • MS Visual Basic used to construct VB applets.
Reporting	Meets query/report needs as follows: • High-volume production reports on main application/database servers (at night) or query server. • PC ad-hoc queries against query server with suitable end-user tool. • OLAP and Business Intelligence tools available for DB2/UDB database.
Integration	• IBM MQ-Series middleware used to integrate with other Mecklenburg County applications on the IBM mainframe. • Message-broker used for integration with other agencies' applications • Additional integration work, including special-purpose COBOL programs, may be needed to integrate with other agencies.
Database	IBM DB2/UDB. (Programmers will see DB2/UDB but data will actually reside in DB2/MVS. Database middleware will connect the two.)
Flexibility, Reuse	Service-oriented architecture encourages the development of an application that is flexible and will be able to reuse component services in multiple places. Only directly affected components change in maintenance mode.
Imaging	The CJIS application will be able to acquire and display images.
Workflow	Selected processes will use automated workflow management.

Table 2-1. Application Architecture Elements

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

3. TECHNICAL ARCHITECTURE

This chapter specifies the Technical Architecture. Section 3.1 shows the hardware, operating systems and enabling software. Section 3.2 discusses the network infrastructure. Section 3.3 focuses on the application development infrastructure. Section 3.4 provides a comprehensive specification for the Application Development Environment (ADE), including tools for client construction and service construction.

3.1 PLATFORMS AND OPERATING SYSTEMS

3.1.1 Desktop Workstations (Clients)

Element	Specification	Comment
Standard Workstation Hardware	Intel PC with built-in network adapter to support 10/100 Mbps Ethernet. Minimum 17" display.	Determine disk and memory configuration just before procurement.
Portable Workstation	Same as above	Same as above
CBT Workstation	As above, configured for multi-media, minimum 19" display.	
DSS/OLAP Workstation	As above, but with faster processor and additional memory.	
Operating System	• Windows 2000 Professional (CJIS) • Web Browser (External Users)	CJIS workstations will run Windows 2000. For external users only a web browser (Netscape or Microsoft) is required.
Browser	Internet Explorer	IE supports dynamic HTML including strong integration with Visual Basic.

Table 3-1. Workstation Elements

The strategy is a "thin client" one, to minimize TCO. "Zero Administration" capabilities will also reduce TCO. Windows 2000 include many new Zero Administration for Windows (ZAW) functions.

County workstations need to be capable of running browser-based CJIS client software, other applications, e-mail and office automation software. Workstation price/performance ratios continue to improve rapidly, therefore exact specifications and the procurement should be done just-in-time prior to CJIS roll-out.

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

3.1.2 Application Server

Reliability and availability are achieved through the use of an Intel cluster with (a) multiple machines, (b) multiple CPUs per machine, (c) the MTS TP Monitor and (d) operating system software – Windows 2000 AppCenter – that provides fail-over capabilities.

Element	Specification	Comment
Application Server	• Windows 2000 AppCenter Server • Intel Hardware	Cluster of Intel SMP machines, to be configured.
Web Server	• Windows 2000 Server • MS Internet Information Server • Intel Hardware	
Database Server	• Windows 2000 Server • IBM DB2/UDB and DB2/MVS • Intel Hardware	Database middleware (DRDA) will connect the two. Programmers will see DB2/UDB but data is in DB2/MVS.
Middleware	• DNA 2000 • MTS • HTML and XML over HTTP	Microsoft Transaction Server (MTS) is an underlying component of Microsoft's Distributed Network Architecture (DNA 2000)
Network	• 100% IP	Local Area Network and Wide Area Network will be 100% TCP/IP.
Query/Report Server	Same as Database Server	
Message Server	Windows 2000, may be separate or combined with another server.	For MQ-Series and the message broker. The MQ-Series hub could also be hosted on the mainframe.
Message-Oriented Middleware	IBM MQSeries	
Message Broker	Add a message broker to MQSeries	The State of North Carolina is recommending HEI's broker and has negotiated favorable terms.
Remote Access	Dial-up access	Secure access for mobile Mecklenburg County users, inside the firewall.

Table 3-2. Application Elements

Notes:

(1) The Web server shown above is for the CJIS application and internal Web pages. Public access to the Mecklenburg County applications, home page (etc.) will be hosted on another server with a similar configuration or possibly out-sourced.

(2) Images will be served from the Application Server and/or the Web Server shown above.

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

3.2 NETWORK INFRASTRUCTURE

From an applications standpoint, it is enough to know that the network is 100 percent native TCP/IP with adequate capacity. Current GartnerGroup recommended best practices are shown below for information purposes. Keep in mind that this area is changing rapidly, and most products available today will be replaced by newer versions in the next 12-18 months.

Element	Best Practice	Comment
Wide Area Network		
Physical	Fiber singlemode UTP CAT-5	
Network and Transport Layers	One type, IP	
Wide Area Remote Access	Variable bandwidth sensitive; frame preferred, minimum 5-10 Kbps/user/site	Frame relay is current recommendation
Hubs, Routers and Switches	Single vendor Switched Devices SNMP-compliant	
Firewall	Dedicated hardware device (not a general-purpose server)	Guardian, Cisco, etc.
Local Area Network		
Physical	UTP CAT-5 or better, consistent use of structured wiring closets	
Connectivity	TCP/IP Switched Ethernet	100 Mbps Ethernet for medium and large offices, 10 Mbps for smaller ones. All new or upgraded LANs 100 Mbps.
Network System Management		
Network Systems Management	Event management console accepting events from multiple sources, with auto correction	HP Openview, CA Unicenter, Cisco Net Management Tools, Tivoli, etc.

Table 3-4. Network Infrastructure Elements

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

3.3 INTERNET ARCHITECTURE

The diagram from Chapter 2 is reproduced here for convenience.

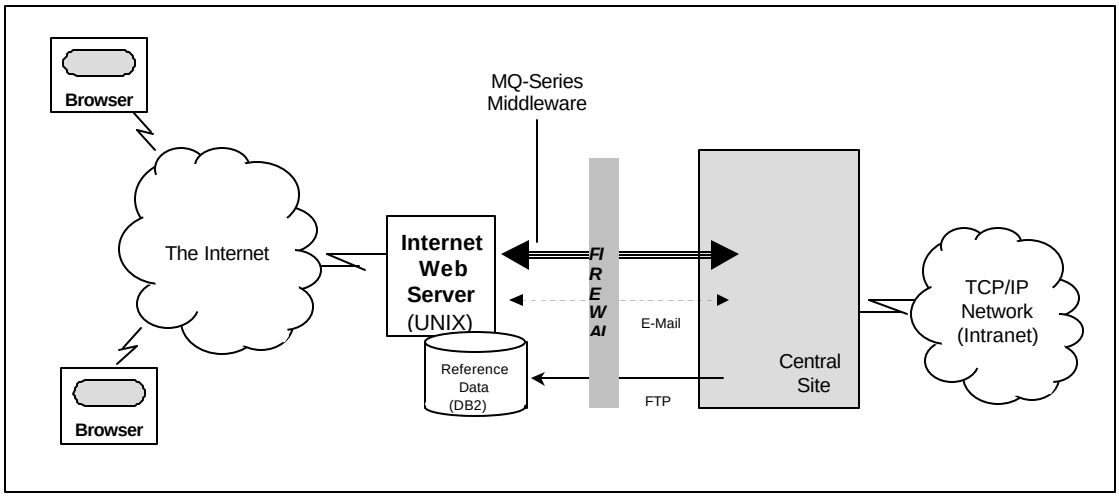


Figure 5-1. Internet Architecture

Elements of the Internet architecture are as follows:

Element	Specification	Comment
Internet Web Server	Same as application Web server	Contains HTML menus, text, forms, applets, etc.
Database for Reference Data	DB2/UDB	A subset of information from the main application database. Data is to (a) support queries and (b) edits of HTML input forms. Transferred nightly via FTP. *Direct access of the main database is prohibited for security reasons.* A permitted alternate method would be to send queries to the application database via MQSeries.
Middleware	IBM MQSeries	Transfers data between HTML forms and the central system. This method was chosen so that transactions can be sent to the application server in near real-time in a secure fashion.
Browser	• Netscape • Internet Explorer	The browser runs standard HTML forms and perhaps the MQ client (a Java applet). Forms and applet automatically download from the Web server. Both Netscape and Internet Explorer browsers must be supported, including older versions. This means that Microsoft proprietary extensions (e.g., dynamic HTML) cannot be used here.

Table 5-5. Internet Architecture Elements

There are several different ways in which the MQSeries middleware connection can be made:

1. An MQSeries client is linked into the Web server. The message queue manager (MQM) is hosted on a central site server, behind the firewall. When a transaction from an HTML form arrives at the Web server a linked-in subroutine passes it to the MQ client.

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

2. IBM's *MQSeries Client for Java*, which is itself a Java applet, is downloaded from the Web server to a browser when needed. HTML forms use the applet to send messages to a MQM residing on the Web server, which in turn routes them to the central MQM. Note that this involves using an IBM-supplied Java applet, not programming in Java.
3. Use of the *MQSeries Internet Gateway* on the Web server, to provide access to MQSeries via CGI.

Any of these methods is permitted. Note that the last (CGI) is likely to have slower performance than the other methods.

3.4 QUERY, DSS AND OLAP TOOLS

Selection of reporting tools is beyond the scope of this architecture. Products that work using Internet technology, through a browser, are an emerging area. These should be favored because they reinforce the thin client approach.

3.5 IMAGING AND WORKFLOW MANAGEMENT SOFTWARE

Selection of this category of embedded software is beyond the scope of this architecture and would in any case be premature. Selection should be made:

- Only after the County has selected a document management system, which will almost certainly include image storage capabilities and perhaps workflow.
- After vendor responses are in and platform decisions are final.
- In conjunction with the systems integration contractor that is selected. Some integration work may be needed.

Web-enabled workflow is still evolving. In this case a just-in-time decision is recommended.

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

4. VENDOR / PRODUCT SELECTION

4.1 MESSAGE-ORIENTED MIDDLEWARE

The CJIS Architecture includes IBM MQSeries both because it has been selected by numerous State agencies and also because it is the leading product in this category.

IBM's MQSeries dominates the MOM market, accounting for about two-thirds of the total 1998 revenue for all separate (unembedded) MOM worldwide. Announced in 1993, MQSeries has been continuously enhanced. It is now up to v.5.1 and it runs in about 3,000 mostly large enterprises. The main strengths of MQseries are its broad platform support, reliability, integrity, good integration with OS/390 and, of course, the low business risk of its vendor. MQSeries runs on 30 platforms, including HP-UX and all major versions of Microsoft Windows.

IBM has fostered an aftermarket for products and services that deal with MQSeries. There are numerous external services providers and consultants (e.g., Candle, MessageQuest, MQSoftware and MQ Tech) with specific expertise in the product. It also is the target for development tool vendors (e.g., Netsoft, for a VB interface to MQSeries) and system management tools vendors (e.g., BMC, including tools obtained in its purchase of Boole and Babbage), Candle, IBM/Tivoli, Landmark and NasTel). MQSeries has become something of a de facto standard. Most integration tools have built an interface or gateway to MQSeries. However, MQSeries is a complex product that can be overkill for some situations. It has many choices for communication semantics and queue handling, which cause it to have a fairly complex programming model despite the fact that it uses only two key verbs (MQPUT and MQGET). It can be a management burden, particularly if there are many participating applications, many queues and many queue managers, although management tools are available to reduce this issue.

MQSeries has limited support for publish-and-subscribe, available either through add-on broker products (see next section) or through a software extension available as a free Web download.

4.2 MESSAGE BROKER

Message brokers (AKA "integration brokers") complements message-oriented middleware. These products either leverage external communication middleware product(s) and/or they embed their own communication facilities. Many include gateways, abstraction layers and application-specific adapters.

These products all support one-way (asynchronous) communication, whether individual messages, large files or databases, or both. This lets them address a broad range of integration scenarios such as 1) data reconciliation among two or more application systems and/or 2) multi-step business processes that involve a sequence of activities among multiple applications. Some of these also support synchronous, request/reply integration, including composite applications.

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

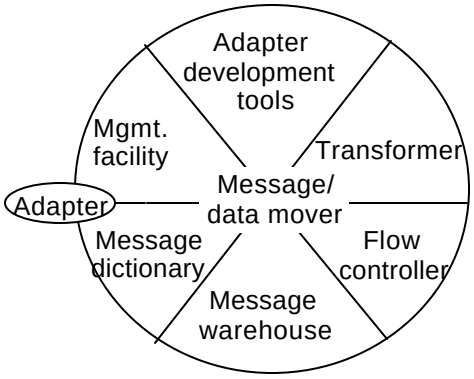


Figure 5-1. Desirable Message Broker Functionality

Mecklenburg County's short-list for message broker vendors should include:

- IBM
- New Era of Networks (NEON)
- HIE

Gartner's "magic quadrant" ranking of message brokers is shown below:

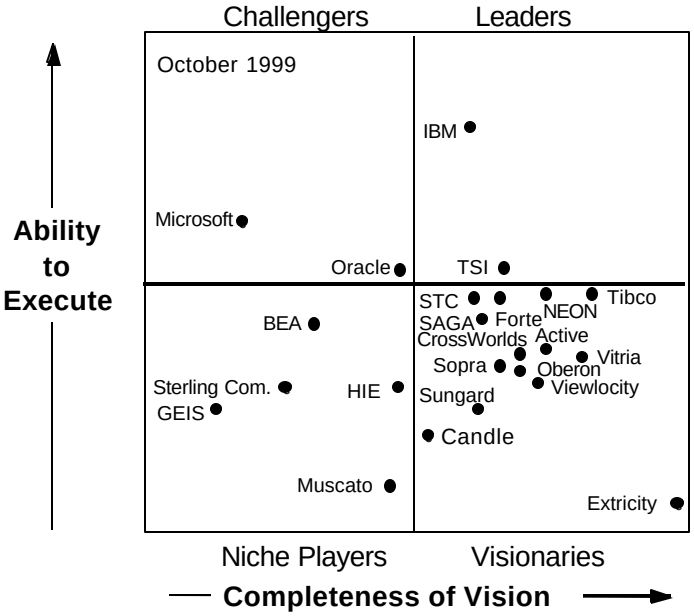


Figure 5-2: Message Broker Vendors

HIE is included primarily because it has been chosen as a recommended product by the State of North Carolina's IRMC and will be used by many agencies. Favorable terms have been negotiated and are available to Mecklenburg County. While not top-ranked, it is a proven

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

product with a good track record. Note that message brokers are advanced technology with a growing but still small market, hence prices can be very high.

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

APPENDIX A – IT PRINCIPLES

The following is a summary of Mecklenburg County's guiding principles regarding application development and the general use of related technologies. Changes related to the Internet are given emphasis and were the primary motivation for this study.

GOVERNING FACTORS

Dynamic Nature of Internet Technologies

Internet technologies emerge and change quickly. The general consensus among industry analysts is that the technologies of today will continue to evolve at a fairly rapid pace and new technologies will continue to emerge. This inevitable change is a product of both the dynamic nature of the Internet and the fact that current technologies do not fully meet industry requirements. Therefore, an architecture will provide the most benefit when it focuses as much on higher level goals and principles as it does on technology choices. Rigid technology standards often have a short life span. The desired end product of a Web architecture is a flexible, manageable technology infrastructure that can accommodate change.

Existing Technologies and Staff Skills

A key success factor in implementing an architecture is minimizing the time and the risks associated with transitioning to new technologies and methods. Another key factor is reducing the number of disparate technologies that must be integrated within the organization. Therefore, it is reasonable to assume that additional weight should be given to those technologies that most closely fit within our organization's current infrastructure and employee skill set. This is particularly true when multiple completing vendors and technologies offer competitive functionality.

The Nature of Government

When selecting technologies, we must keep in mind the budgetary limitations that we typically face as a governmental organization. Therefore, cost must be a consideration when evaluating competing technologies. We must also be sure that the selected technology infrastructure can be managed by our current staff, or by those whom we can reasonably expect to attract and retain. A related issue is the need to minimize the risks associated with adopting newer, more advanced technologies. Leading edge technologies inherently carry more risk and are therefore not necessarily good choices for a local governmental organization. It is more important for us to be leaders in the application of technologies than to be just technology leaders.

Time Frame for Internet Applications

Our current Web applications are mostly small, read only systems that do not employ sophisticated transaction processing. According to our survey, it will probably be in excess of

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

two years before we could implement an enterprise level Web application. This time line makes it possible for us to seriously consider technologies that are not considered enterprise level at this time, but have the likelihood of maturing into that category within a two year window.

GENERAL TECHNOLOGY APPROACH

The following are some basic principles to which Mecklenburg County IST should adhere regarding application development:

- Be Flexible – As previously stated, a good architecture is one that provides a flexible, manageable technology infrastructure for development. There must be a degree of flexibility built into the architecture to accommodate the inevitable changes that will occur in the market place. There must also be the flexibility to accommodate the different technology needs between types of applications. For example, for Web applications, we must allow for the differences between online transaction processing applications and read only applications, and between Internet and Intranet applications. Each situation may require a different technology approach.

- Use Mainstream Technologies- With Web technologies currently in a state of flux, and with no single dominant vendor or technology at this time, it is important to choose technologies and methods that are projected to have good longevity and mainstream acceptance. Proprietary solutions that offer only short-term benefits should be avoided.

- Take a Conservative Approach - Given the budgetary constraints and the dynamic state of the market, it is important for Mecklenburg County to be conservative in adopting new technologies. We are not a leading edge technology organization and must stay on a proven, well-traveled path.

- Ensure Bridges to other Technologies – Despite the need for standardization, there will always be vendor segmentation of technologies. Vendors feel compelled to differentiate their products from their competitors in order to enhance their position in the market. Technologies chosen as part of this architecture should not be so isolated as to preclude some form of bridge to other competing products. The justification for the technologies chosen for this study should include the identification of plausible alternatives in the event that the vendor or product does not survive.

- Promote Integration and Reuse - The architecture that we put in place should enable integration of disparate technologies. Where possible, the Web enabling of older applications should involve the reuse and integration of working code rather than a complete rewrite of the application. Just as we prefer to purchase before building, we should also favor integration before rewriting when there are no other compelling reasons to replace the existing code. The mainframe and its associated technologies will continue to play an important role in automated solutions at Mecklenburg County and it is important that we choose technologies that can connect to and integrate with the data and applications in this

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

environment. Windows NT (soon to be Windows 2000) and OS/390 are platforms of choice for our existing applications architecture and there is little chance of that changing in the near future.

- Control and Limit Application Interfaces – According to industry analysts, most organizations do a poor job of addressing application integration issues and end up with a patchwork of information sharing methods such as data extracts, screen scraping, file transfers, etc. With the Web, integration issues go beyond internal applications and technology. It is anticipated that there will be an increasing need to communicate and exchange data with outside organizations and vendors as well as the public. Therefore, it is important that we employ structured, mainstream technologies for both internal and external program to program communication.

- Limit proliferation of technologies - Although flexibility is a high priority, we still need to make the effort to reduce the number of technologies and vendors that we introduce into our environment. This is one of the primary benefits of an architecture. Technology decisions should be based on the organization's architecture and the goals and principles upon which that architecture is based, as opposed to aiming for short- term gain. A single vendor solution would be ideal in that it would reduce the risks associated with transitioning to new technologies, but care must be taken to ensure that all of the County's requirements can be met under that scenario.

- Use a Layered Approach - We should strive for logical layering of architecture components to promote flexibility and scalability. Logical layering implies that the various components of an application are clearly delineated and can be individually managed regardless of the physical implementation or distribution. This type of layered approach provides better control and some insulation from changing business requirements or technologies. Logical layer boundaries must be clearly defined within our architecture The following is a list of logical layers:

  Presentation
  Validation and transformation
  Business logic
  Data access
  External resource access

  As long as the logical layers and their boundaries are defined, then the physical layering, or resource placement, becomes part of the individual application design. The position of each layer relative to the firewall is also a physical consideration that should not overly effect the application. How these layers are segmented and how we exchange information between these layers is the core of our architecture.

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

Successful layering of applications requires that application logic be designed and written in small, interconnected, logically distinct segments. According to industry analysts, this type of development is emerging as the dominant development methodology for server based applications because it promotes flexibility, scalability and application integration. For example, when there is a clear separation between the client interface logic and the business logic, the business logic can be accessed by multiple client interfaces without change. This type of benefit has led both Gartner and Giga to pronounce the two-tier client server approach near its end as a viable architecture for all but smaller, departmental applications. This type of architecture may require that we employ some form of application server to help manage components and provide some functionality and server-side run time services, such as load balancing and thread pooling, that more robust applications require. It is likely that some type of application server will be a part of our new architecture. This piece of the architecture must be well integrated with both our application development tool (Visual Basic) and with the selected object request broker.

- Favor Thin Clients- According to Gartner, over 70% of the total cost of ownership for an application is related to the ongoing management of that application and its necessary infrastructure. The deployment of a thin client strategy can help keep the total cost of ownership for an application down by reducing or eliminating the need to build, install and support client software and heterogeneous clients. The technology is available to dynamically propagate applications and new functions to user devices in a relatively instantaneous manner. Ideally, IST can "write once, run anywhere" without having to dictate the technology used by customers, the public or other organizations. There may be some instances, particularly for in-house applications, where some code will need to be placed on the client's machine due to user requirements or for reasons of efficiency.

- Continue RDBMS Course – We are currently using Microsoft SQL Server as the relational database for the Windows NT platform and IBM's DB2 for OS/390. The introduction of Web technologies should not significantly impact this strategy and we should continue to resist the introduction of any new RDBMS vendors into our technology mix.

ONGOING CONCERNS

Systems Development Life Cycle

Once the Web Architecture is implemented, it is recommended that we have the necessary processes in place to ensure compliance. Our Systems Development Life Cycle can accommodate the appropriate reviews through the architectural review task in both the project initiation and requirements phases. Smaller projects that do not use Synergy would need to schedule some type of architectural review as part of their project initiation.

It is anticipated that the existing client server methodology in Synergy can meet the development requirements for a component-based application. However, it would be helpful if the first application developed under the new architecture follow the client server model and document

Entire contents © 1998 Gartner Group, Inc. All rights reserved.

any changes or departures that are specific to component/Web development. At that point, we could either modify the client server model to include tasks specific to component/Web development or develop a new methodology within Synergy.

Role of Architecture Team

It may be helpful to keep some form of the Web QIT in place beyond the scope of this project. Tasks for this team could include architecture reviews for specific applications through Synergy, and periodic reviews of the appropriateness of the architecture, particularly as technologies evolve and our experiences change. Despite the Web focus of this study, we are actually updating our current architecture to include Web technologies. In the future, Web and client server development will be distinguished only by the type of client interface employed.

A periodic review of our architecture will keep our organization moving in the right direction.

Entire contents © 1998 Gartner Group, Inc. All rights reserved.