IBM Software Group

WebSphere. software

IBM

University of Toronto

Application Integration

January 26, 2006

ON DEMAND BUSINESS

Kelvin Yung, IBM Canada Ltd
Business Integration Solution Specialist

Module Objectives

After completing this module, the participant should be able to

- Understand what is Application Integration
- Understand different patterns of application Integration
- Understand Quality of Service of application Integration
- Understand how application integration fits into the overall Business Integration Architecture

Agenda

History

Why application Integration

Application Integration pattern

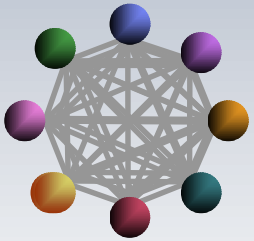
Quality of Service

Application Integration Styles

Exercise

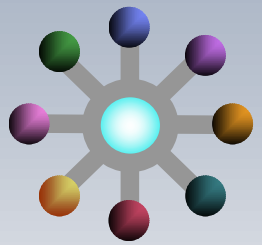
SOA builds flexibility on your current investments . . . The next stage of integration

Messaging Backbone



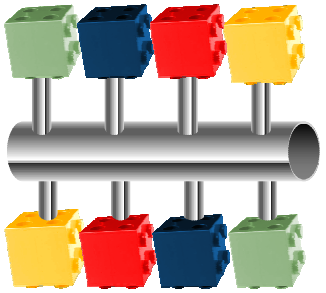
- Point-to-Point connection between applications
- Simple, basic connectivity

Enterprise Application Integration (EAI)



- EAI connects applications via a centralized hub
- Easier to manage larger number of connections

Service Orientated Integration



- Integration and choreography of services through an Enterprise Service Bus
- Flexible connections with well defined, standards-based interfaces

Flexibility

As Patterns Have Evolved, So Has IBM

History

Silos

Historically, applications were written to solve specific, well-delineated problems. There was little vision at the time of an application landscape that would cover the whole range of business requirements, so no need for an integrated architecture was seen. As a result, solutions would evolve on a great variety of platforms.

Batch oriented

If and where integration was needed, it was usually achieved by hosting the applications on the same system and sharing files.
This was no great restriction, since most applications at that time were batch oriented and large central computers (the “mainframes”) were the accepted technology standard.

Data driven

Data were moved between systems to the applications which required them.

- Physical move of data. For example delivery of tapes
- FTP

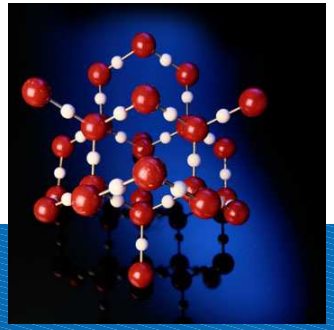
Files remained the favorite entities to share because

- They were well understood and had worked well between applications on the same system.
- Support was available for cross-platform file transfers and file sharing on network servers.
- Most applications were still batch oriented.

Online processing

- collect data during the online day (in files)
- actual processing performed during nightly batch runs.

e-business On Demand



An on demand business is an enterprise whose business processes—integrated end-to-end across the company and with key partners, suppliers and customers—can respond with speed to any customer demand, market opportunity or external threat.

Challenges

Reliable and flexible information flow between diverse applications and organizations

Customer Challenges	Challenges Addressed By WebSphere
▪ Applications are not integrated in a flexible and reliable way across the enterprise … reducing business responsiveness	▪ Reliable, seamless data exchange between multiple applications
▪ Differences between many internal applications and between business partner applications must be managed	▪ Management of differences between multiple internal applications and business partner applications
▪ Maintaining point to point or custom written integration interfaces is cost and time prohibitive	▪ Adoption of an enterprise wide, flexible, service oriented approach to integration

Technology challenges

- Heterogeneous platforms
- Programming complexity to add communication functions
- Technology choice - Network protocols
- Non standard APIs
- Transaction controls across platforms

Why Enterprise Application Integration ?

EAI provides:

- Great cost and efficiency benefits by automating and controlling the interactions of disparate systems
  - Provide new business functions by
    - tying applications together to provide more complex functions
    - combine new applications with existing applications
    - Form new services with mix and match of existing functions and services
  - Shorten development time and enhance re-use

- Supporting technology for Business Process Management (BPM)

- Technology for Business-to-Business (B2B) enablement

Who should do application integration ?

Business and IT drivers

- The business processes need to be integrated with existing business systems and information.

- The business activity needs to aggregate, organize, and present information from various sources within the organization.

Application Integration requirements

• Seamless execution of multiple applications and access to their respective data in order to automate a complex, new business function.

• Reliable integration of applications—be they legacy stovepipe applications, packaged software applications, or custom applications—requires the use of proven, repeatable patterns.

Application Integration Pattern

The Application Integration pattern serves to integrate multiple Business patterns or to integrate applications and data within an individual Business pattern. It is applicable when integrating applications and data within the bounds of an organization.

Two different approaches:

- **Process-focused integration**:
  The integration of the functional flow of processing between the applications.
  For example, the integration of an e-commerce application with an Enterprise Resource Planning (ERP) system for a newly created sales order would most definitely be a Process-focused integration activity.

- **Data-focused integration:**
  The integration of the information used by applications.
  For example, the master data synchronization of the product catalog between the ERP system and the e-commerce system would be a Data-focused integration activity.

Which is the right pattern ?

Understanding your applications

Enterprise Application Integration is a complicated undertaking. It requires, first, a thorough understanding of the individual applications being integrated, and also the possible methods that can be used to interconnect them.

Request for information versus request for processing

- The Process-focused Application Integration patterns are concerned with integration of the functional flow of processing between applications.
- The Data-focused Application Integration patterns are concerned with integration of the information used by applications.

Foreground versus background integration

Is there a user awaiting the outcome of the operation or is this operation running behind the scenes?

Scope of integration

Does the integration project involve only a single Business pattern, multiple Business patterns, or the creation of an entire e-infrastructure for multiple e-business solutions?

Understanding your applications - cont.

Operation latency (applications or data queries)

Operations that can not complete in less than a couple of seconds dictate the need for asynchronous methods of integration. A query on product inventory may be a quick operation, whereas the computation of the production plan for the manufacturing of that inventory could take minutes to hours to complete.

Geographic proximity

How close do the applications being integrated reside to one another?
Integration of applications residing in the same data center has a much smaller integration latency than integration of applications spread around the world.

Application portfolio

What is contained in the mix of applications? The portfolio might include pre-packaged software, legacy applications, or newly developed applications.

Understanding your applications - cont.

Process re-engineering

Is there a need to re-engineer business processes or extend an existing business process?
Is the EAI effort for better integrate functional operations of a disconnected, narrow business process?
Is it for business processes improvements?

There are varying degrees of process extensions for application-based BPM:
- Extending reach of the business process with integration to other applications.
- Joining together two separate application-based business processes into one unified process.
- Separating BPM from application logic by implementing the process in a Process Manager.

Invasive versus non-invasive

The impact of the change. The degree of invasiveness is often described in terms of coupling (loose coupling versus tight coupling) or a black box versus white box approach.
Ideally, the less invasive the integration, the more successful the integration will be long-term.
This is the primary reason for the use of messaging-based integration to isolate as much as possible of the integration processing from any application-specific dependencies. EAI best practices should be employed to ensure that the integration is as non-invasive as possible.

An Enterprise architecture (EA)

The enterprise architecture is an instantiation of the application functions, application data model, application interfaces, and application flow of control.
A good Enterprise Architecture (EA) takes into account new business processing requirements.

The completeness of the EA often will dictate the level of invasiveness in the EAI integration. A well conceived EA enables a more extensible enterprise application integration design.

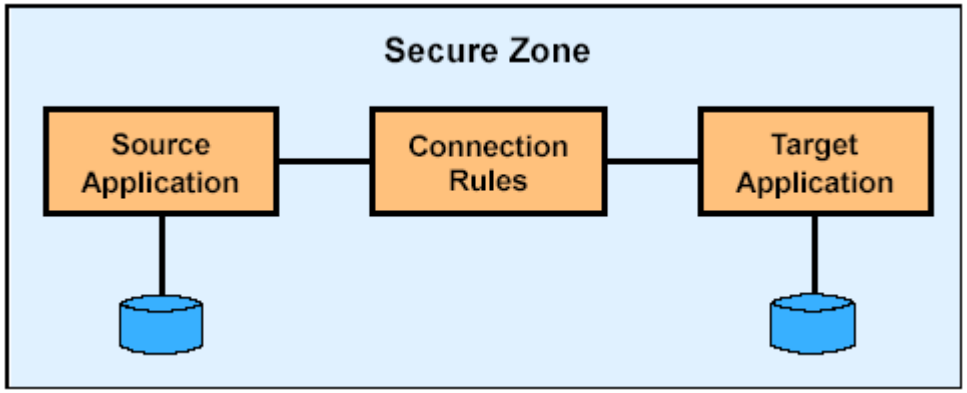
Key characteristics of the EA that affect the EAI approach include the:

- Number of applications
- Degree of centralization of the data repositories
- Completeness of the application interfaces
- Conformity of the participating applications to the EA data and interface model

Application patterns

- Direct Connection application pattern
  - Message/Call Connection variations
- Broker application pattern
  - Router variation
- Serial Process application pattern
- Parallel Process application pattern

Direct Connection



The Direct Connection application pattern has two variations:

- Message Connection variation
- Call Connection variation

All applications of the Direct Connection application pattern will be one variation or the other. The variation required depends on whether the initiating source application needs an immediate response from the target application in order to continue with execution.

Both variations may be used either with synchronous or asynchronous communication protocols. However, there are preferences for a specific protocol type depending on the variation.
For example, the Call Connection variation has a more natural fit with synchronous protocols while the Message Connection variation favors asynchronous protocols.

Direct Connection

The business and IT drivers are to:

- Improve the organizational efficiency
- Reduce the latency of business events
- Support a structured exchange within the organization
- Support real-time one-way message flows
- Support real-time request/reply message flows
- Leverage existing skills
- Leverage the legacy investment
- Enable back-end application integration
- Minimize application complexity

Benefits

The Direct Connection application pattern offers the following benefits:

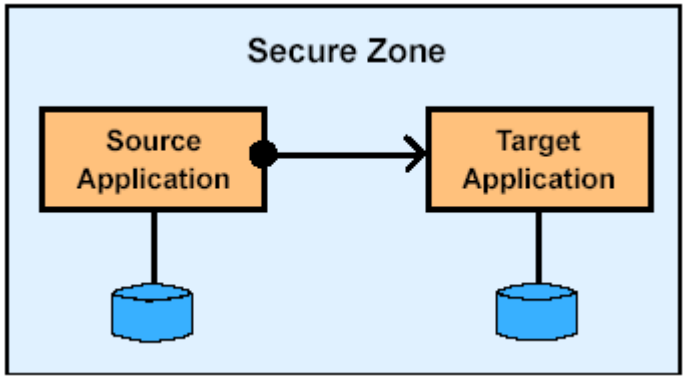
- It works with applications that have simple integration requirements with only a few back-end applications.
- It increases the organizational efficiency and reduces the latency of business events by providing real-time access to business data and business logic, and avoiding manual synchronization of data between applications.
- Direct access to back-end applications reduces the duplication of business logic across multiple tiers. As a result, changes to business logic can be made in one tier rather than in multiple applications.
- It can enable re-use of investments already made with the organization.

Limitations

- This pattern will result in a many to many "spaghetti" configuration with point to point integration mappings for each application pair.
- The expansion of this implementation into a multi-point configuration will require additional application logic to handle the coordination.
- This pattern cannot be used for intelligent routing of requests, decomposition and re-composition of requests, and for invoking complex business process workflow as a result of a request from another application. Under such circumstances, you should consider a more advanced Application pattern, such as Broker or Serial/Parallel Process.

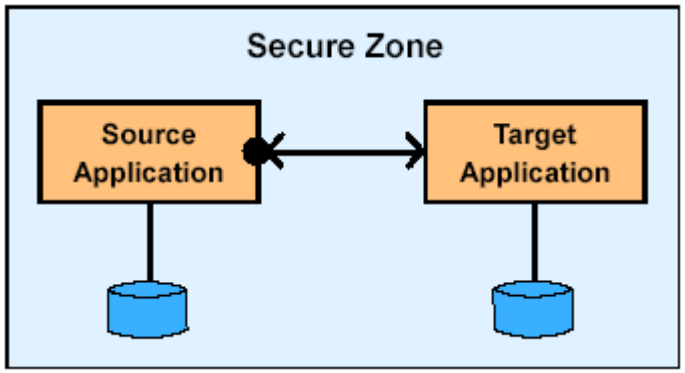
Message Connection variation

Sending messages between applications



Call Connection variation

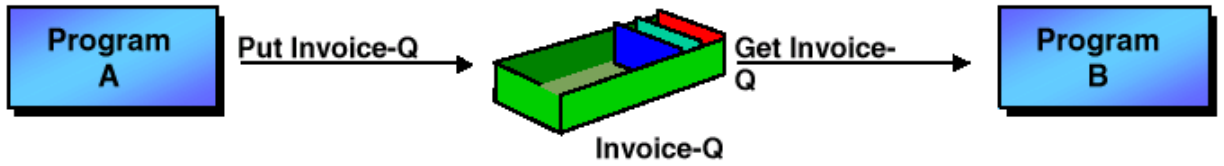
The Call Connection variation, applies to solutions where the business process depends on the target application to process a request and return a response within the scope of the interaction.



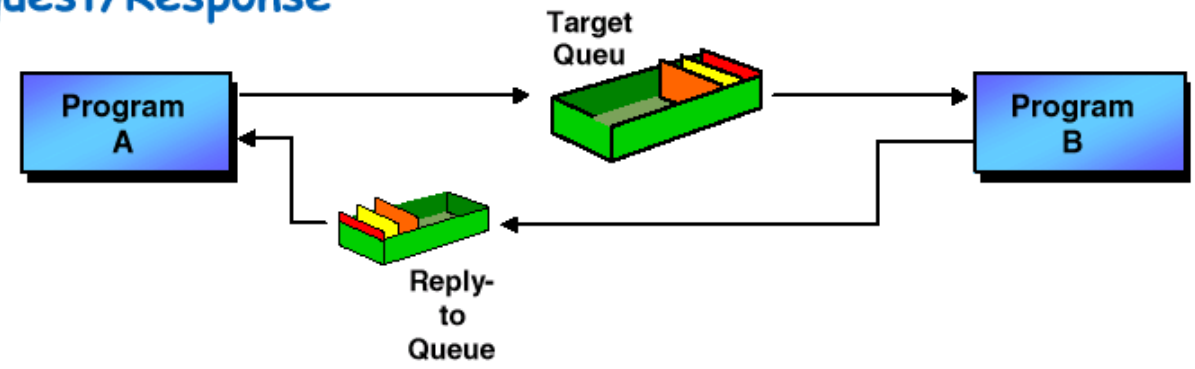
Message Connection variation

Examples

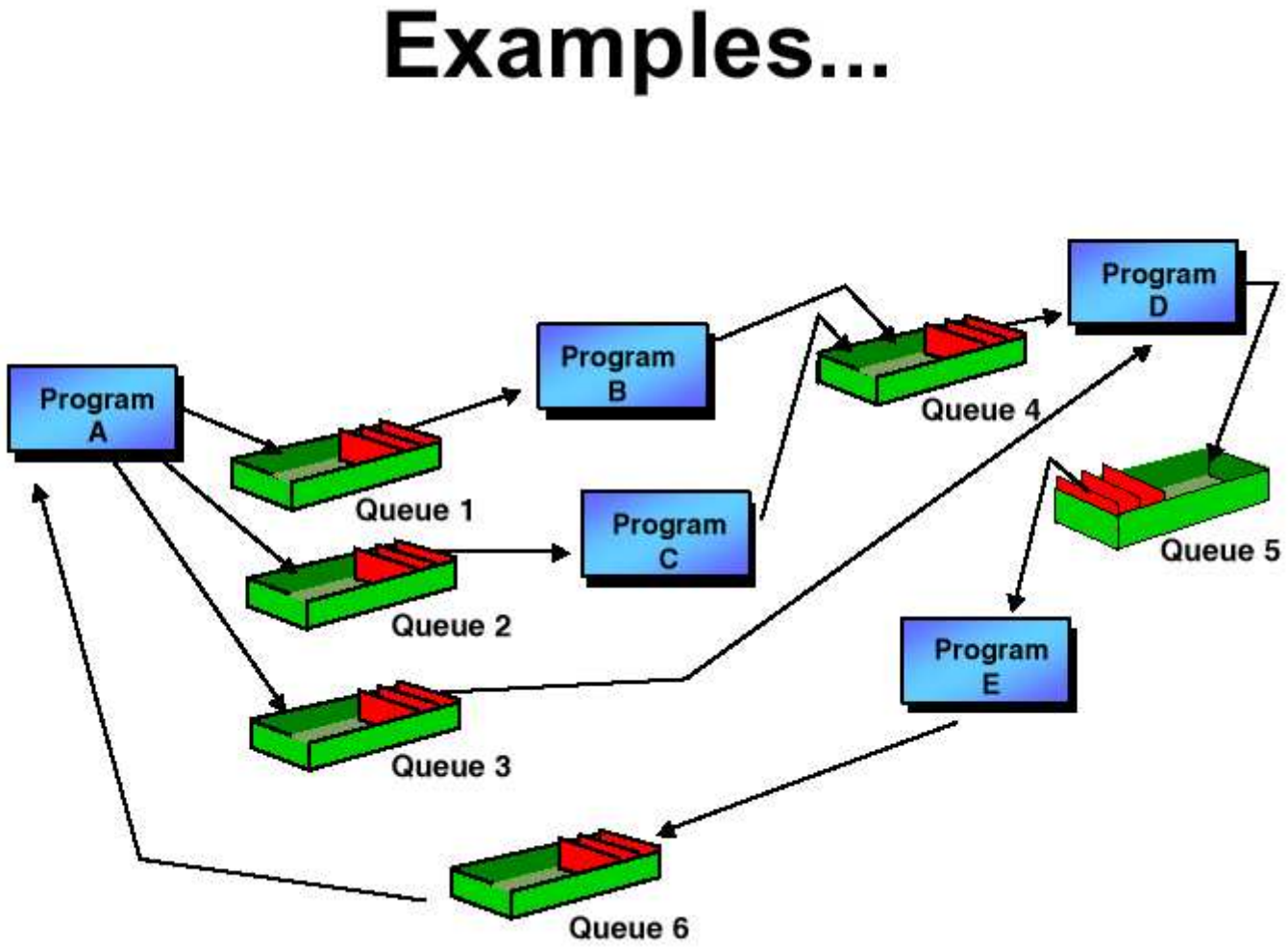
'Send & Depend'



Request/Response

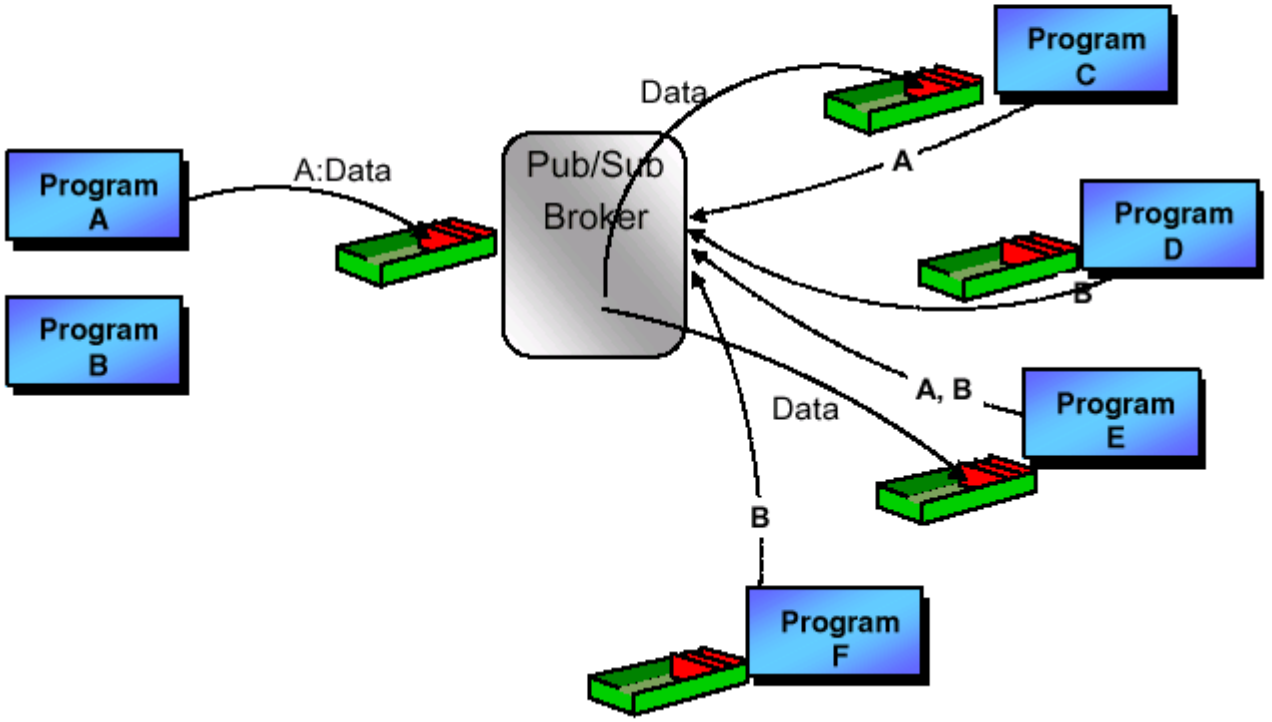


Message Connection variation

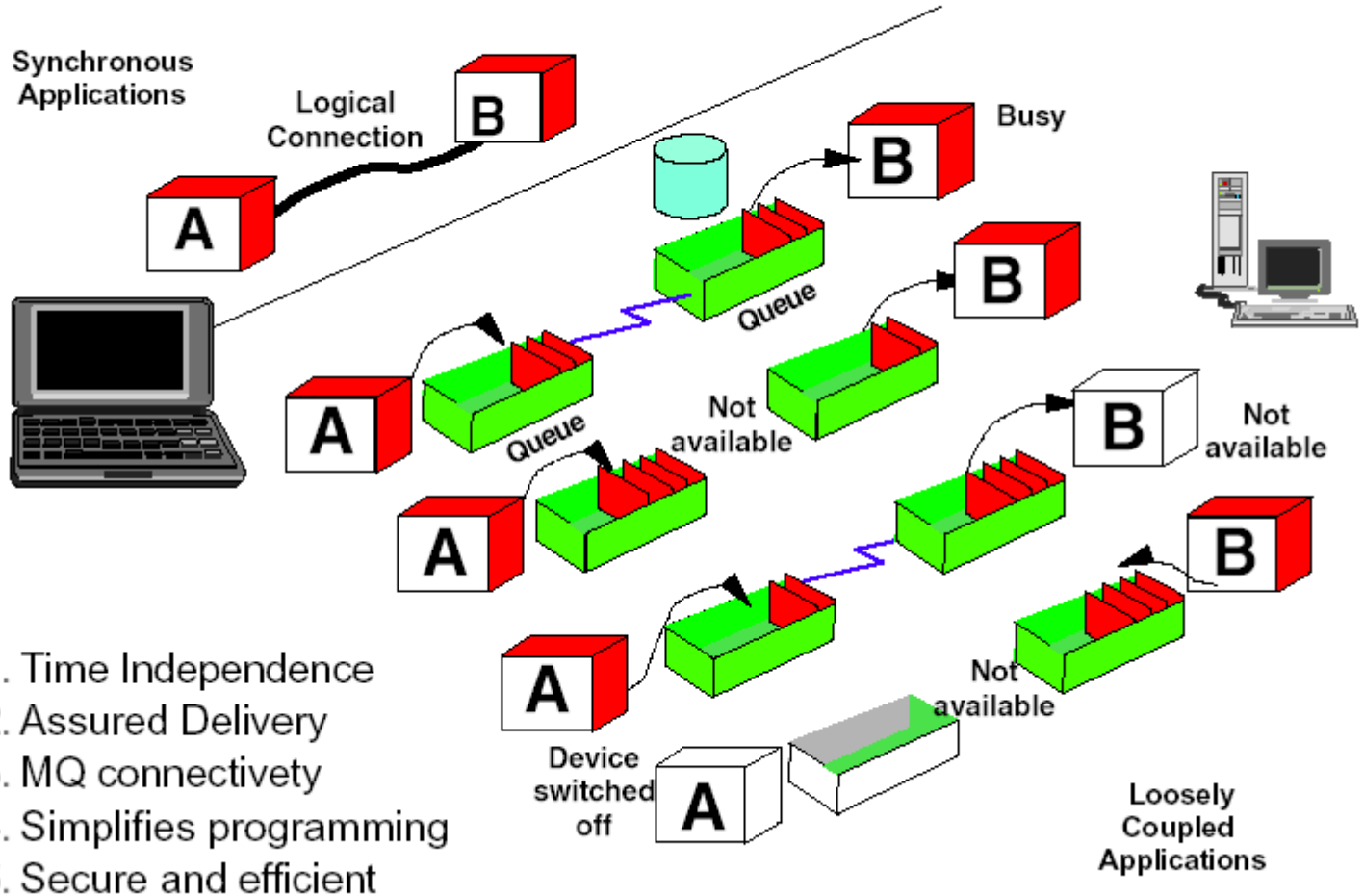


Message Connection variation

Examples...Publish/Subscribe



Message Connection Benefits



1. Time Independence
2. Assured Delivery
3. MQ connectivety
4. Simplifies programming
5. Secure and efficient

Call Connection variation

Examples

Remote Procedure Call (RPC) is a protocol that one program can use to request a service from a program located in another computer in a network without having to understand network details. (A procedure call is also sometimes known as a function call or a subroutine call.)

RPC uses the client/server model. The requesting program is a client and the service-providing program is the server.

Like a regular or local procedure call, an RPC is a synchronous operation requiring the requesting program to be suspended until the results of the remote procedure are returned.

When program statements that use RPC are compiled into an executable program, a stub is included in the compiled code that acts as the representative of the remote procedure code.

Call Connection variation

Examples

CPI Communications (CPI-C) provides a cross-system-consistent programming interface for applications that require program-to-program communication.

The model is described in terms of two applications--*speaking* and *listening*--hence, the term *conversation*. A conversation is simply a logical connection between two programs that allows the programs to communicate with each other.

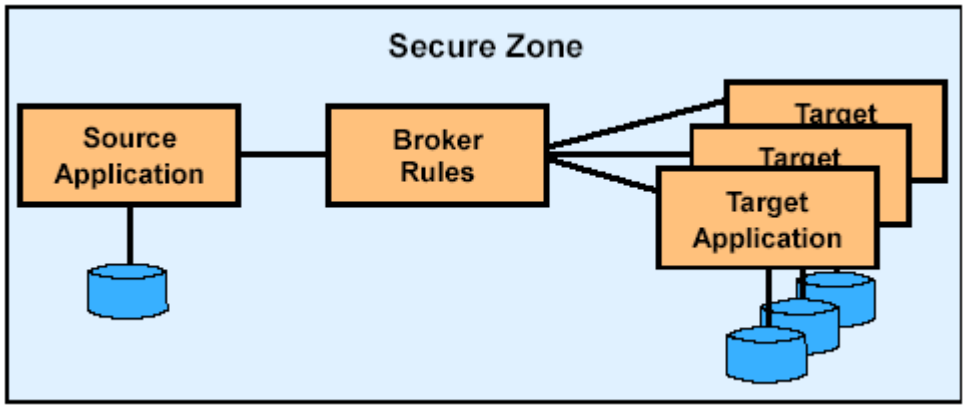
From an application's perspective, CPI-C provides the function necessary to enable this communication.

The conversational model is implemented in two major communications protocols, Advanced Program-to-Program Communication (APPC) and Open Systems Interconnection Distributed Transaction Processing (OSI TP). The APPC protocol is also referred to as logical unit type 6.2 (LU 6.2).

CPI-C provides access to both APPC and OSI-TP.

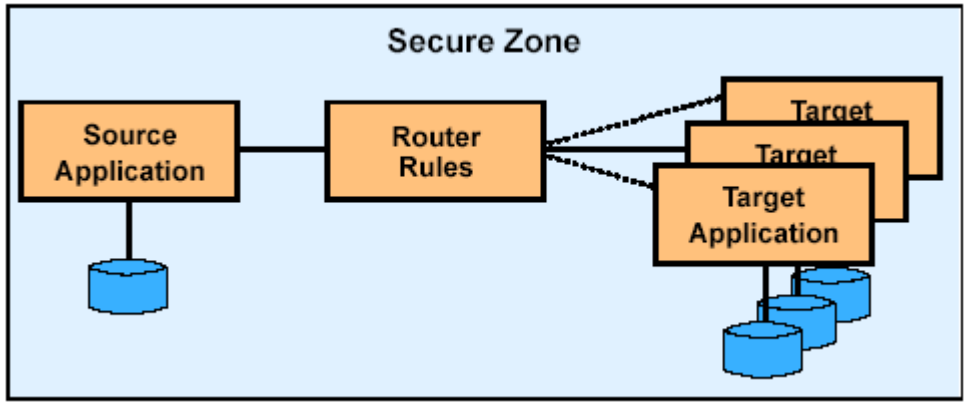
Broker application pattern

The Broker application pattern, is based on a 1-to-N topology that separates distribution rules from the applications. It allows a single interaction from the source application to be distributed to multiple target applications concurrently.



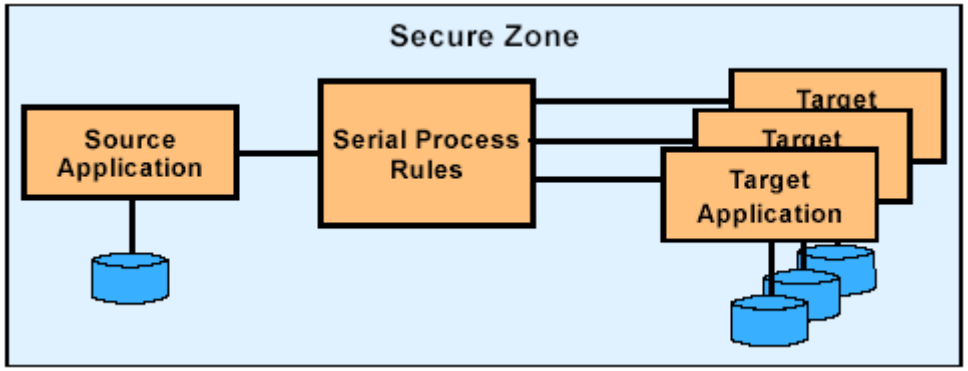
Router variation

The Router variation of the Broker application pattern, applies to solutions where the source application initiates an interaction that is forwarded to only one of multiple target applications. The selection of the target application is controlled by the distribution rules that govern the functioning of the connector component.



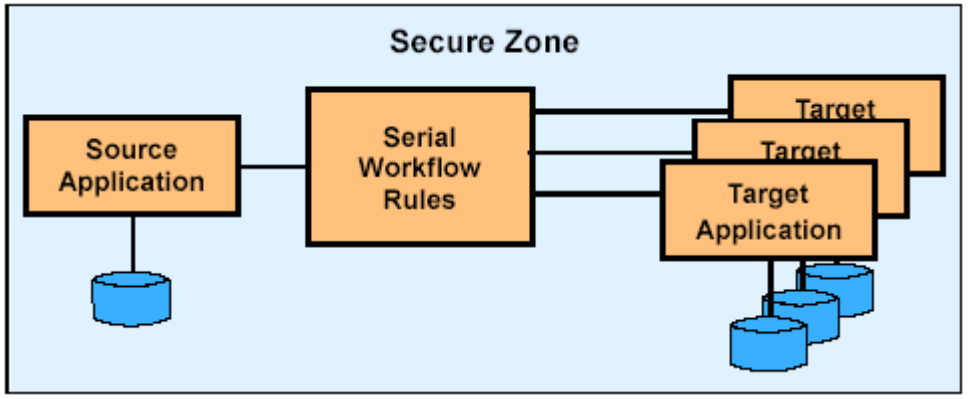
Serial Process application pattern

The Serial Process application pattern, is based on a 1-to-N topology where serial process rules are separated from the applications. It allows a single interaction from the source application to execute a sequence of target applications. The Serial Process application pattern separates the process logic from the application logic. The process logic is governed by serial process rules that define execution rules for each target application, together with control flow and data flow rules. It may also include any necessary adapter rules.



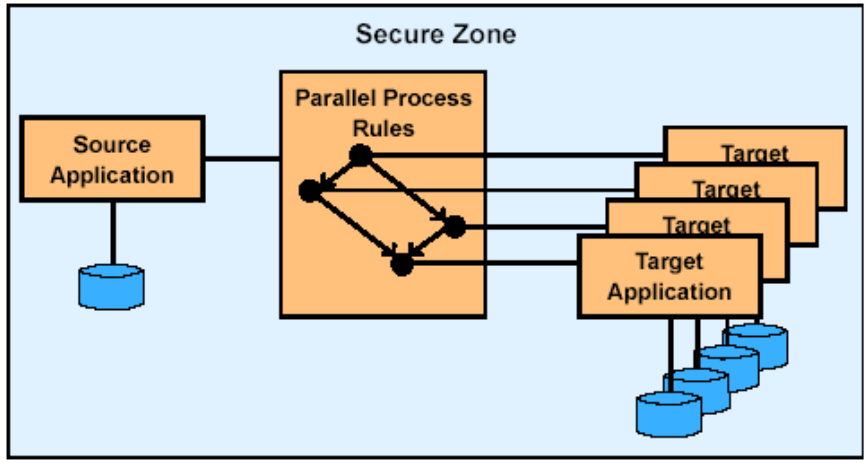
Serial Workflow variation

The Serial Workflow variation of the Serial Process application pattern, allows for routable activities (operations requiring human interaction, for example) to be routed to a suitable resource. In addition to the serial process rules, the serial workflow flow rules are supplemented with resource definitions and task-resource relationships.



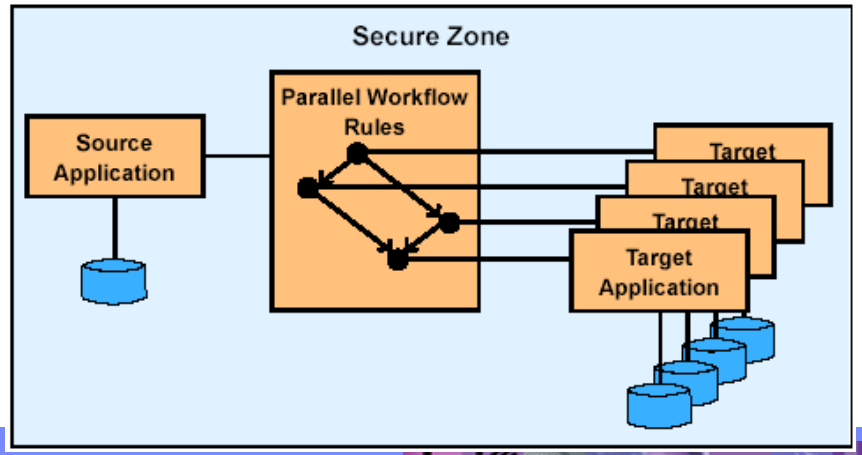
Parallel Process application pattern

The Parallel Process application pattern, is a combination of the Serial Process application pattern and the Broker application pattern. The interaction initiated by the source application may control concurrent (parallel) activities on multiple target applications. Each activity may consist of a sequence
of operations executed in succession on a target application.



Parallel Work Flow variation

An extension of the Parallel Process application pattern to account for routable activities.



Application connectivity middleware capabilities :

Data definition - The ability to describe the format of the messages or business objects so that the data elements contained in them can be accessed and manipulated.

Data transformation - The manipulation of data elements used to build different outgoing data structures from the incoming ones. This capability is typically used to map data to adjust messages and business objects between the native formats of the sending and the receiving applications.

Data routing - The action of delivering messages or business objects to a number of alternative destinations based on business rules and information derived from the incoming data.

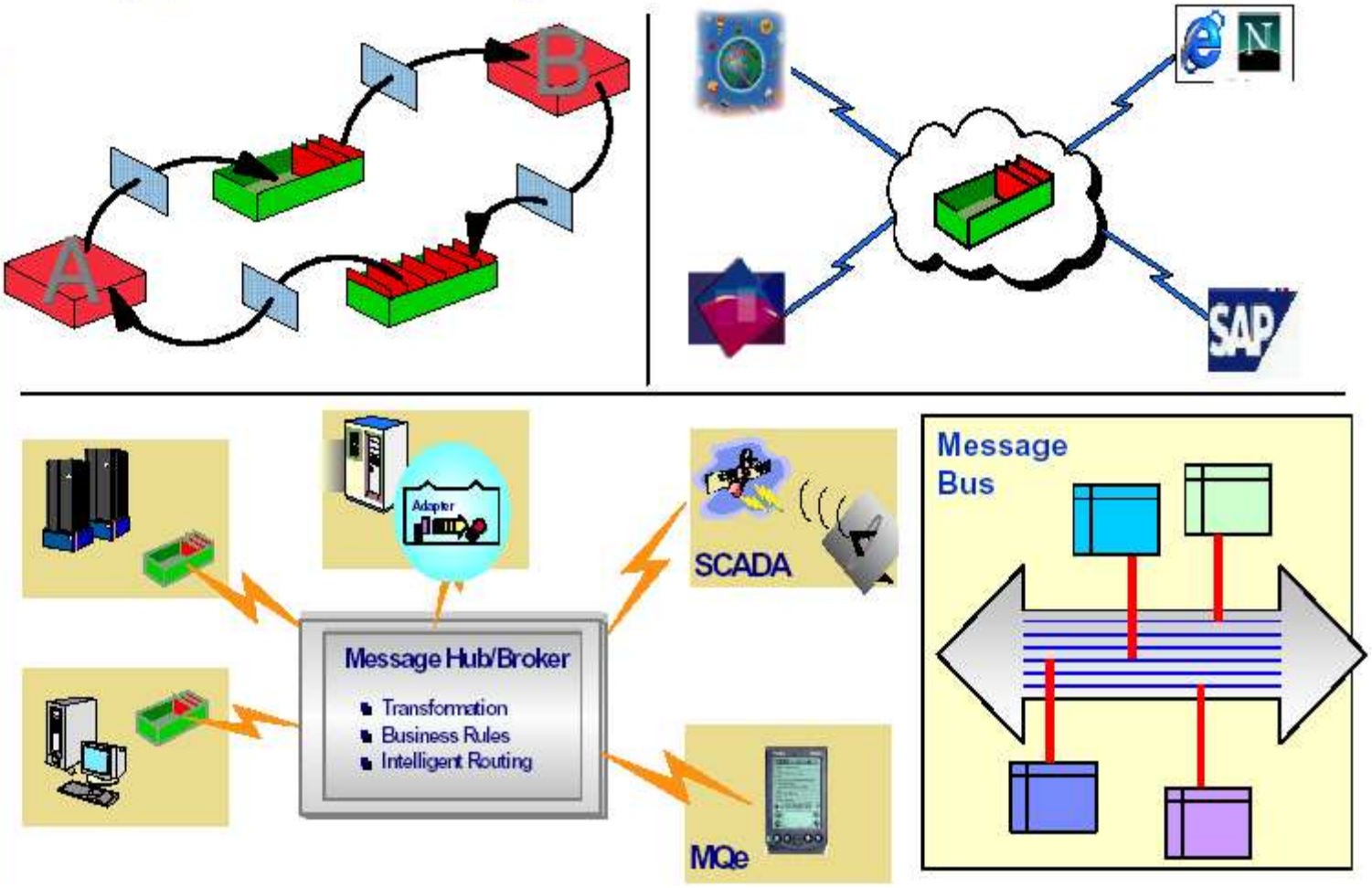
Data delivery - The physical movement of data through the system. The properties of the transport layer in use, such as assured delivery, confirmation of delivery reports, transactionality, and possibly audit trailing or logging are reflected here.

Data aggregation - A feature that allows collecting a set of related messages from potentially different sources and combining them into a single consolidated message.

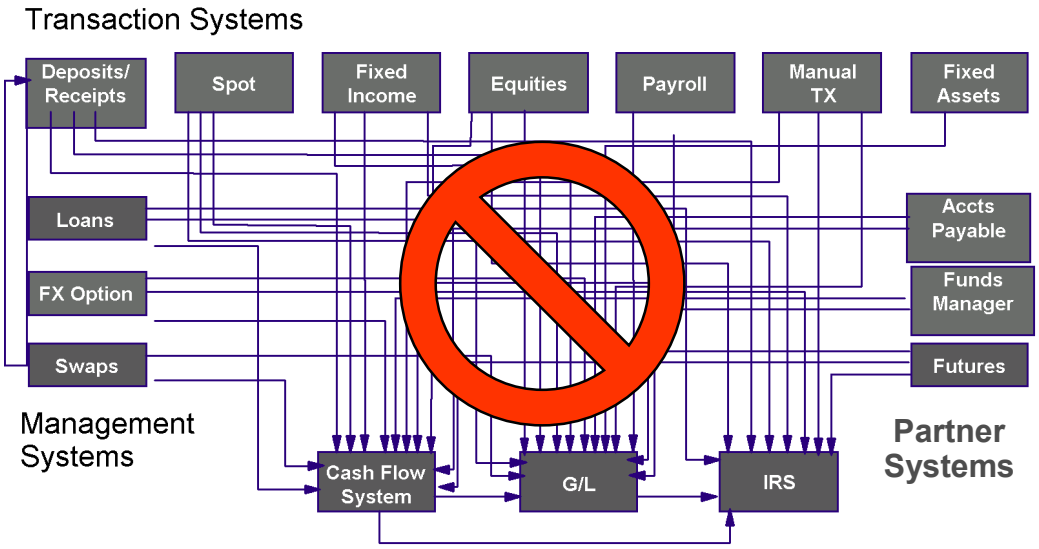
Data augmentation (also referred to as “enrichment” at times) - Refers to the capability of the application connectivity layer to provide extra benefit by retrieving additional information, for instance from external databases, and to incorporate this information into the outgoing message.

Enterprise Application Integration

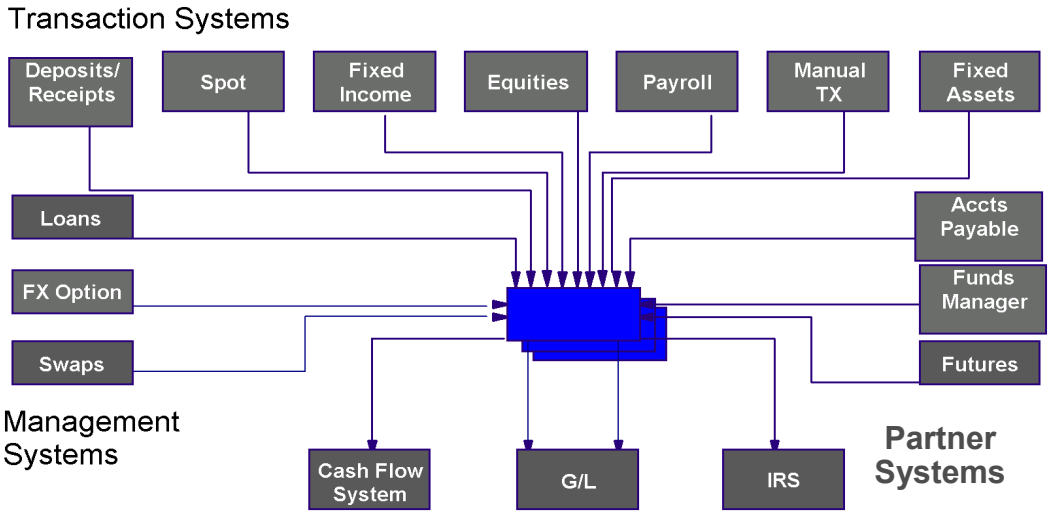
Starting with the benefits of MQSeries and the messaging model, moving to bridges and adapters, we arrive at WBI Message Broker and Event Broker. These enable arbitrary applications to connect in innovative ways to form new value integrated applications.



Application and Partner Mediation…



Point to Point Integration
Connectivity issues
Data Format issues

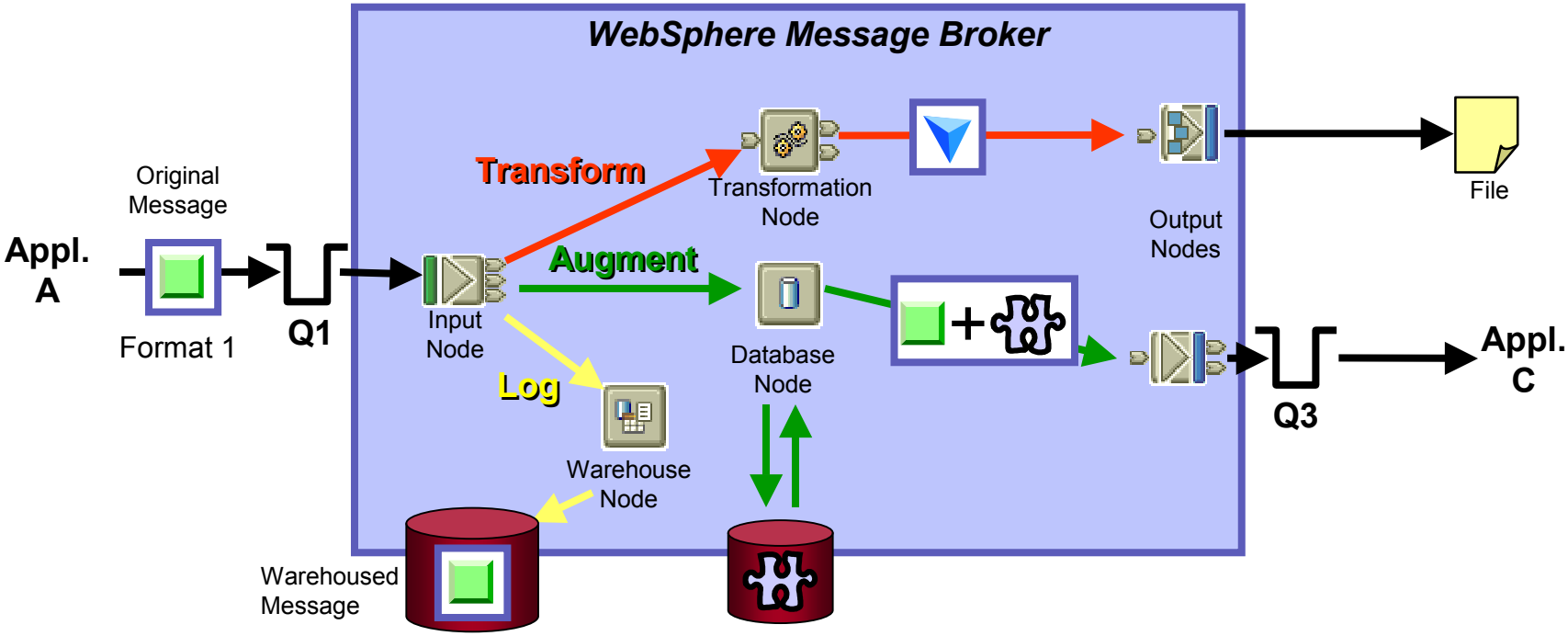


Application Mediation: WebSphere Message Broker

Delivers information targeted to the specific needs of each receiver.

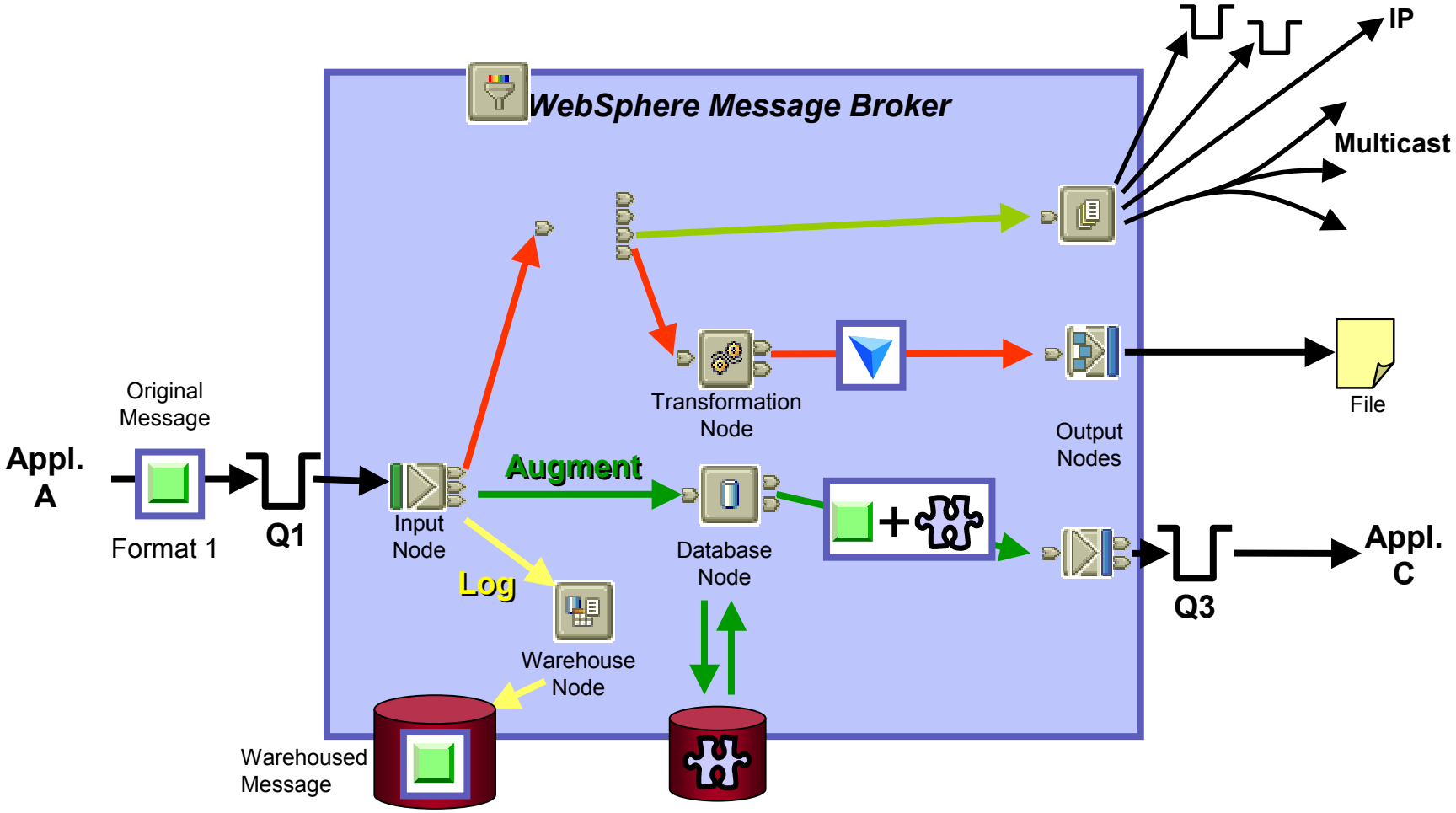
- **Examines content and routes accordingly.**
- **Transforms content .**
- **Augments content.**
- **Logs content.**
- **Matches and compares content.**

…with end-to-end transactional delivery…



…and graphical tooling built on the Eclipse framework.

Application Mediation: WebSphere Message Broker…



… plus support for many more data formats than just XML

Pattern selection

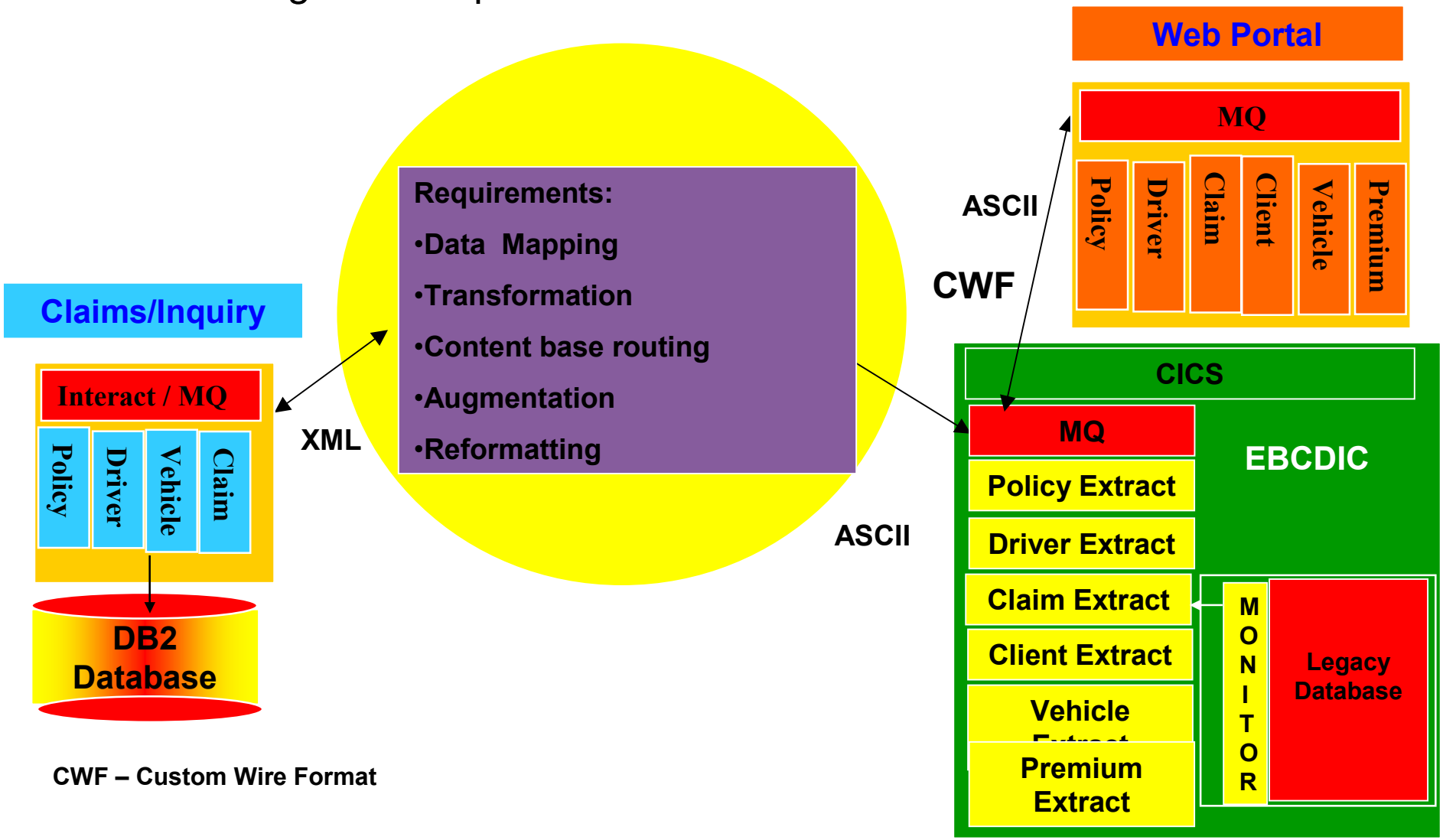
Business drivers	Direct Connection Message variation	Direct Connection Call variation	Broker Router variation	Broker	Serial/Parallel Process
Improve the organizational efficiency	✓	✓	✓	✓	✓
Reduce the latency of business events	✓	✓	✓	✓	✓
Support a structured exchange within the organization	✓	✓	✓	✓	✓
Support real-time one-way “message” flows	✓		✓	✓	✓
Support real-time request/reply “message” flows		✓	✓	✓	✓
Support dynamic routing of “messages” to one of many target applications			✓	✓	✓
Support dynamic distribution of “messages” to multiple target applications				✓	✓
Support more flexible, time-sequenced business and human process flows					✓

Pattern selection

IT drivers	Direct Connection Message variation	Direct Connection Call variation	Broker Router variation	Broker	Serial/Parallel Process
Leverage existing skills	✓	✓	✓	✓	✓
Leverage the legacy investment	✓	✓	✓	✓	✓
Enable back-end application integration	✓	✓	✓	✓	✓
Minimize application complexity	✓	✓	✓	✓	✓
Minimize enterprise complexity			✓	✓	✓
Exploit parallelism				✓	✓

Real life example - Insurance company

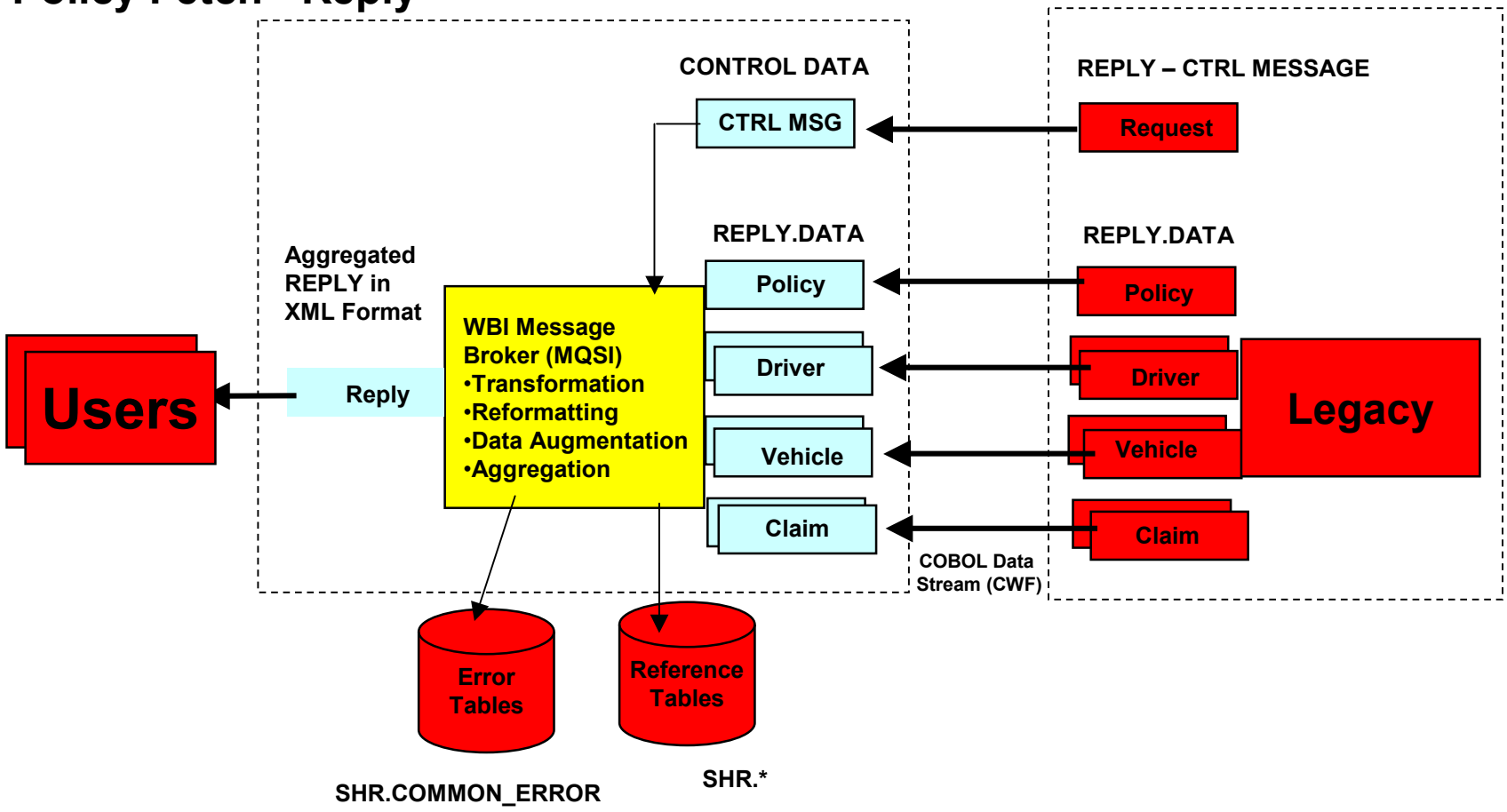
Business Integration requirements



CWF – Custom Wire Format

Real life example – Insurance Policy inquiry

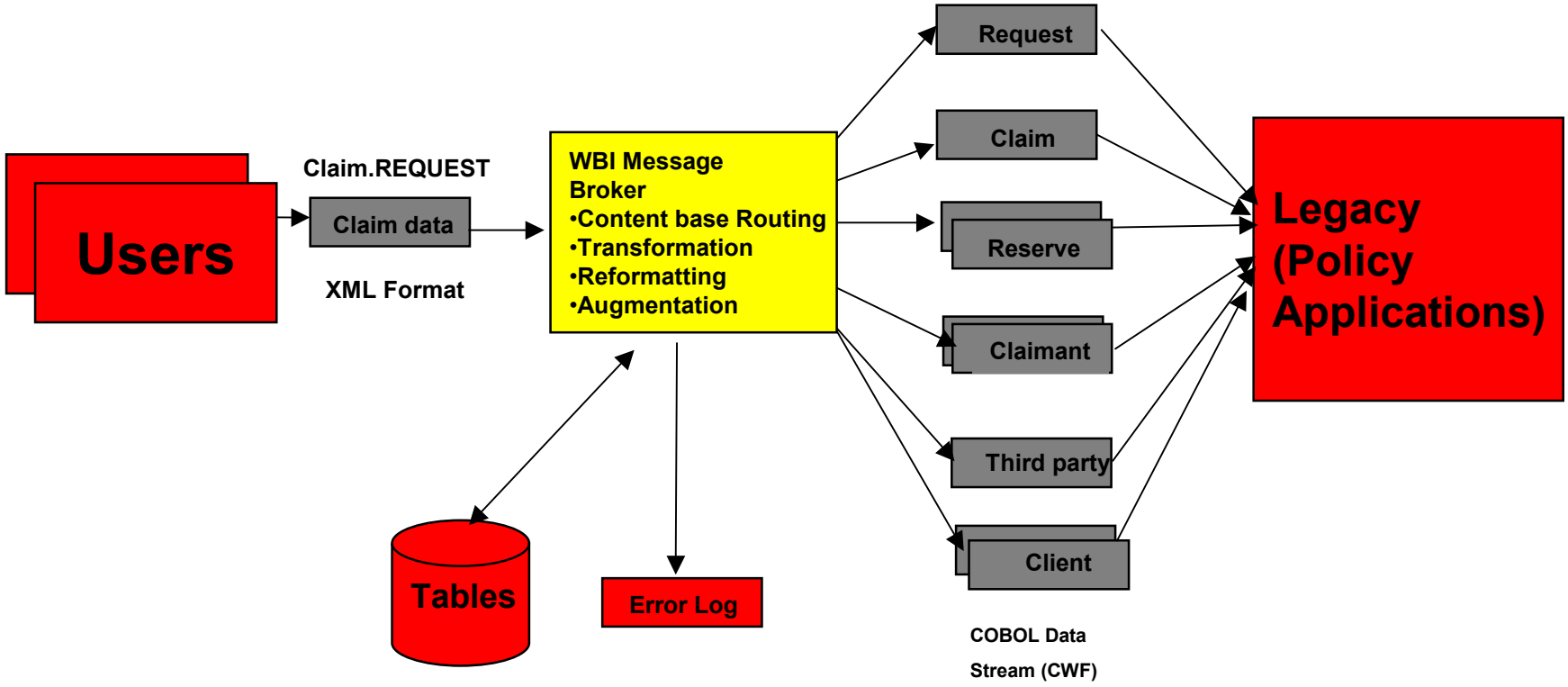
Full Policy Fetch - Reply



Legacy systems – mainframe OS/390

Real life example – Insurance Claim

Claim Summary Put – Push Request



Data-focused Application patterns

When applications need to share information rather than coordinate processing, data-focused application integration is more appropriate than a process-focused approach.

Note, however, that when the frequency of data update is extremely high (for example, when integrating an order entry system with a back-end ERP system), process integration is the best solution.

In delineating Data-focused Application Integration patterns, two key environmental questions should be asked:

Is the enterprise data topology centralized or decentralized?

•Centralized: This integration effort will bring about centralized access to all or a subset of the enterprise data model.

•Decentralized: Applications will retain their isolated repositories but now with cohesion based on data integration.

What is the database affinity type?

•Homogeneous: all repositories are of the same type.

•Multi-vender Relational: all repositories are relational with ODBC/JDBC support for interoperability but are from different vendors.

• Heterogeneous Structured: repositories are not all relational but all have a structured layout.

• Structured/Non-Structured: the need to integrate non-structured (for example, free-form text) with structured data sources.

Quality of Service (QoS):

- Operability

  This QoS concern focuses on the systems management requirements of the deployed solution. It focuses on issues such as monitoring, logging, traces, recovery, and manageability of the solution during operations in a production environment.

- Availability
- Federation

  Federation is fundamentally about enabling services to interoperate across trust boundaries. It lets access control functions span across multiple domains, crossing application, product, platform, site, business unit, and organization boundaries.
  Federation requires that each partner domain is trusted to authenticate the identity of its own users. Mechanisms are needed for passing resource and user authentication and authorization information between domains.

- Performance
- Security

  •Data protection through encryption
  •Authentication of users and subscribing applications. Authorization of the user for participation in an integration activity

Quality of Service (QoS): continue

- Standards compliance

  Standards compliance is an important factor for controlling development and integration costs. Even private standards are beneficial, but widely accepted public standards have the added advantage of
  enabling interoperability in the broadest contexts.

- Transactionality

  Transactionality enables multiple application operations to be coordinated to provide an atomic
  deterministic end result.
  Resource managers are used to control access to the resources involved in a transaction.
  A transaction manager is responsible for coordination and transaction control.
  Transactional considerations include:
  - ACID versus compensating transactions
  - Flat versus nested transactions
  - System versus client commit control
  - Local versus distributed transactions

ACID

Atomicity

In a transaction involving two or more discrete pieces of information, either all of the pieces are committed or none are.

Consistency

A transaction either creates a new and valid state of data, or, if any failure occurs, returns all data to its state before the transaction was started.

Isolation

A transaction in process and not yet committed must remain isolated from any other transaction.

Durability

Committed data is saved by the system such that, even in the event of a failure and system restart, the data is available in its correct state.

Five styles of integration :

- User interaction

  This style delivers information drawn from multiple sources in a personalized fashion through diverse channels. It creates a single user experience across applications on a variety of devices. This style is implemented by using portal, host integration, and mobile device technologies, including such functions as transcoding, translation, and personalization. Federated database searches are a part of this style, as well as a consolidated view of applications that provide related information but are physically disparate and not integrated at all. This style also meets the user requirement for a unified and consolidated view of his IT resources, including such features as single sign-on.

- Application connectivity

  This style ensures enterprise-wide access to information and ensures its timely and reliable delivery. It provides connectivity between applications and thus forms the basis for many EAI solutions. It is based on reliable messaging as a foundation for transformation and routing functions. It provides a choice of transports, APIs, and adapters, and supports a variety of data formats. It is not concerned with the activities to be performed by the participating applications, but rather - while connecting them - provides for a degree of isolation between them. This allows the applications to exchange information without any need to concern themselves with the characteristics of other applications in the system, such as their availability or functional specifications.

- Process integration

  This style coordinates and controls activities that may span multiple systems and involve people in a variety of roles. It implements, automates, and manages business processes while providing runtime measurements that will then assist in improving the process models.

  Process integration can support long-running transactions and roles-based human activities. The flow of a business event through the process can be modified by external input either by parameters provided when the process is instantiated, or by information retrieved from external data sources, such as an application database, or by human decisions, such as in an approval step.

  An important goal of process integration is to facilitate reuse of the various components in a process flow. This could be the programs implementing work steps or whole subprocesses. It is a common and central feature of process integration middleware that processes can invoke other processes and be invoked by external processes or applications themselves (nested process layers).

- Build to integrate

  This form of integration enables a business to build and deploy new composite applications that integrate existing assets, such as legacy systems or ERP packages, with new technologies, such as Web services.

- Information integration

  This style integrates information across systems via database federation, transformation, and replication technologies.

  This style of integration has been around since the introduction of database technology. It typically requires the systems sharing information also having to share the physical data models, processing rules, and constraints. It normally is not as explicitly event-related as a message-based style and therefore has its limitations. Still, in the real world of IT, which includes legacy technology and closed systems, there are many cases where this is the only feasible style of integration, and in those cases it does good service.

  There are systems that are not open to integration technologies. In such cases the only way to build a connector or adapter into them may well be on the basis of data integration.

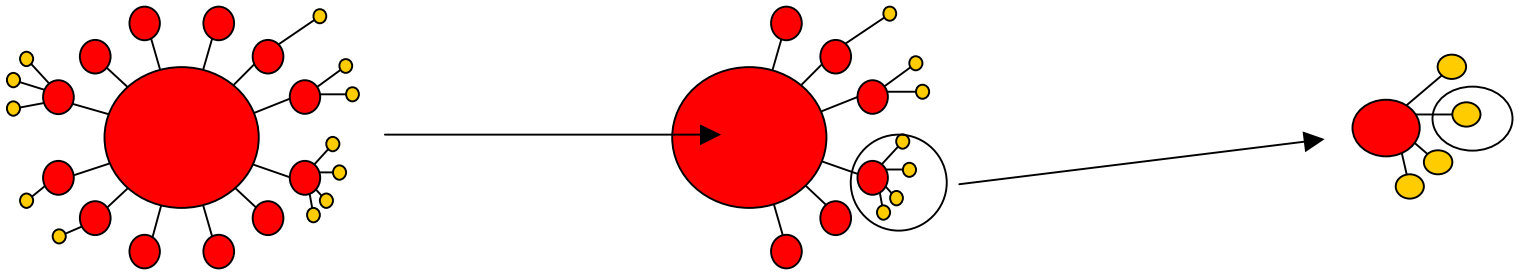
The Next Steps

The next steps, already under way, in the evolution of integration technology are these two:

- Inclusion of external systems, such as those of your business partners, suppliers or customers, into your process automation - usually referred to as Business-to-Business Integration (B2B).
- Development of a comprehensive set of standards for Web-based service infrastructures, known as Web services. Standardizing the way services can be exposed and invoked will make such services easily accessible from anywhere on the Internet and thus will make a huge contribution towards doing business on the Internet.

Partner Services

The Community Integration Ecosystem



Community Manager

- Mandates standards adoption to a number of its downstream trading partners.
- The Community consists of the dominant trading partner and the first tier of Community Participants
- Community Manager focuses only on the first tier suppliers, who integrate directly to it, but need to “see” their entire value chain

Peer Participant

- Needs to be able to react to the rapidly changing dynamics of their environment
- They will be, at times, a community manager, a community operator and a community participant according to needs and the exchange of information required
- They operate in a fluid, ever changing environment to which they must adapt in a cycle of continuous improvement

Community Participant

- Driven by the need to maintain its business with its customers
- Looking for continuation of its business rather than driving dramatic change and growth
- Integration with trading community provides opportunity to grow business
- Could become a Peer within the community if they integrate further with other partners
  - Visibility of new business will drive the decision

Application Integration Characteristics

Facilitates communication between services

Facilitates interactions with existing information and application assets

Connect with trading partners

Exercise

Use what we have learned so far on application integration to design a claims clearing house solution.

The objective of this exercise is to map the solution to see how application integration fits into the overall Business integration Architecture.

Example: Claims Clearing House Solution Architecture

Medical Claims arrive from multiple sources – multiple claims in a single document

- Mostly EDI format documents

Pre-process claims

- 1 verified claim per document
- Each claim assigned a unique ID
- Each unique ID returned to submitting institution

For each claim

- Establish routing … target payer
- Verify for target payer
- Post invalid claims to human reviewer … correct and re-verify
- Record claim in database
- Transform claim to payer preferred format … usually EDI
- Route claim to payer

Programs required (existing or to be developed)

- Validation of requests
- Translating and reformatting (between EDI and Application data)
- Claim verification
- Unique ID generation
- Reply sending

:

- Routing to target payer
- Verify target payer

:

- Invalid claim detection
- Invalid claim posting
- Claim database update

:

:

Etc....

High level Flow

Receive claim (Partner Services)

May need to split file back to individual request

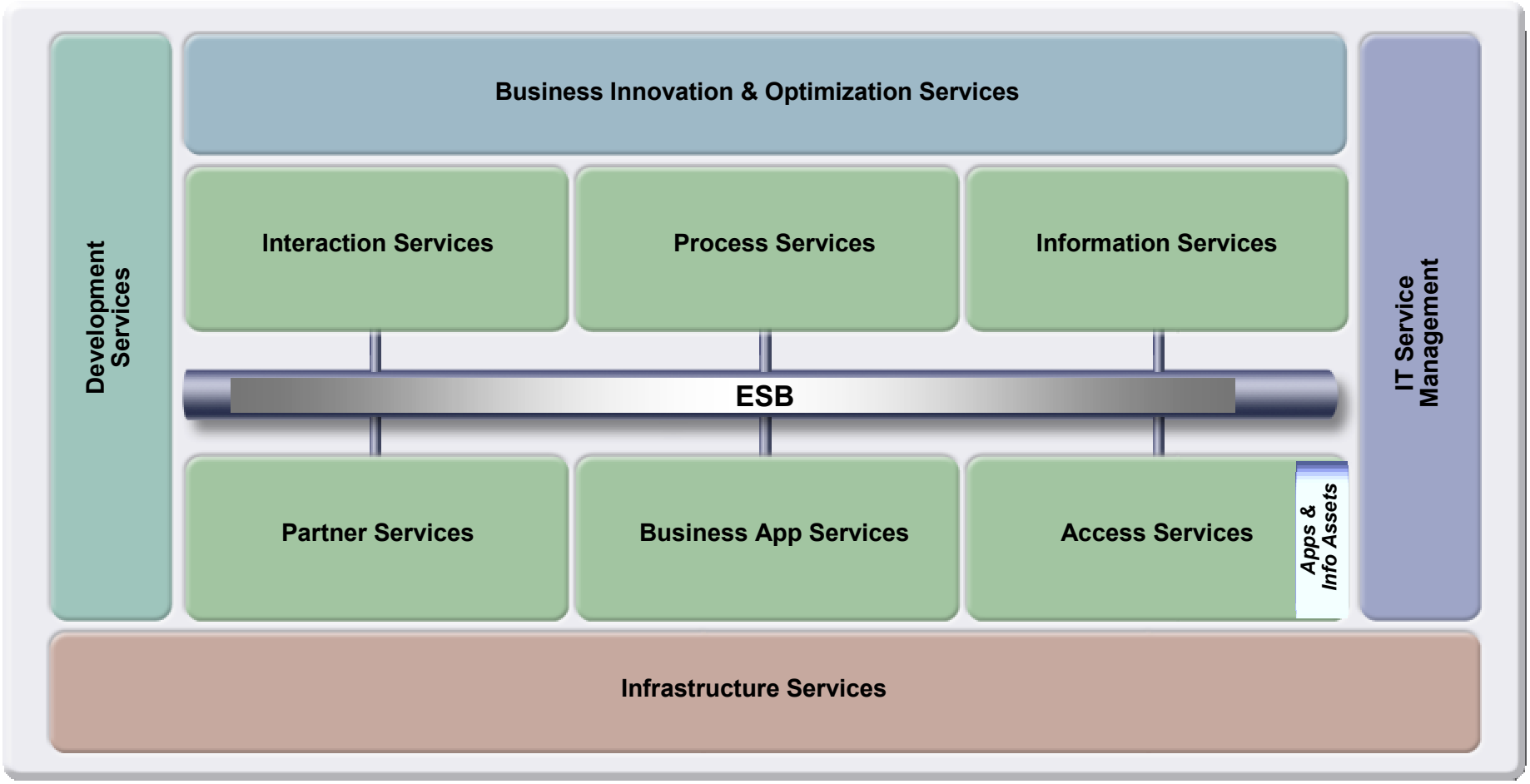
Claim pre-processing

- Translate / reformat to per application requirement
- Initial claim data verification
- Assign unique ID
- Update claim request data base
- Acknowledge and return ID
- Reject if invalid

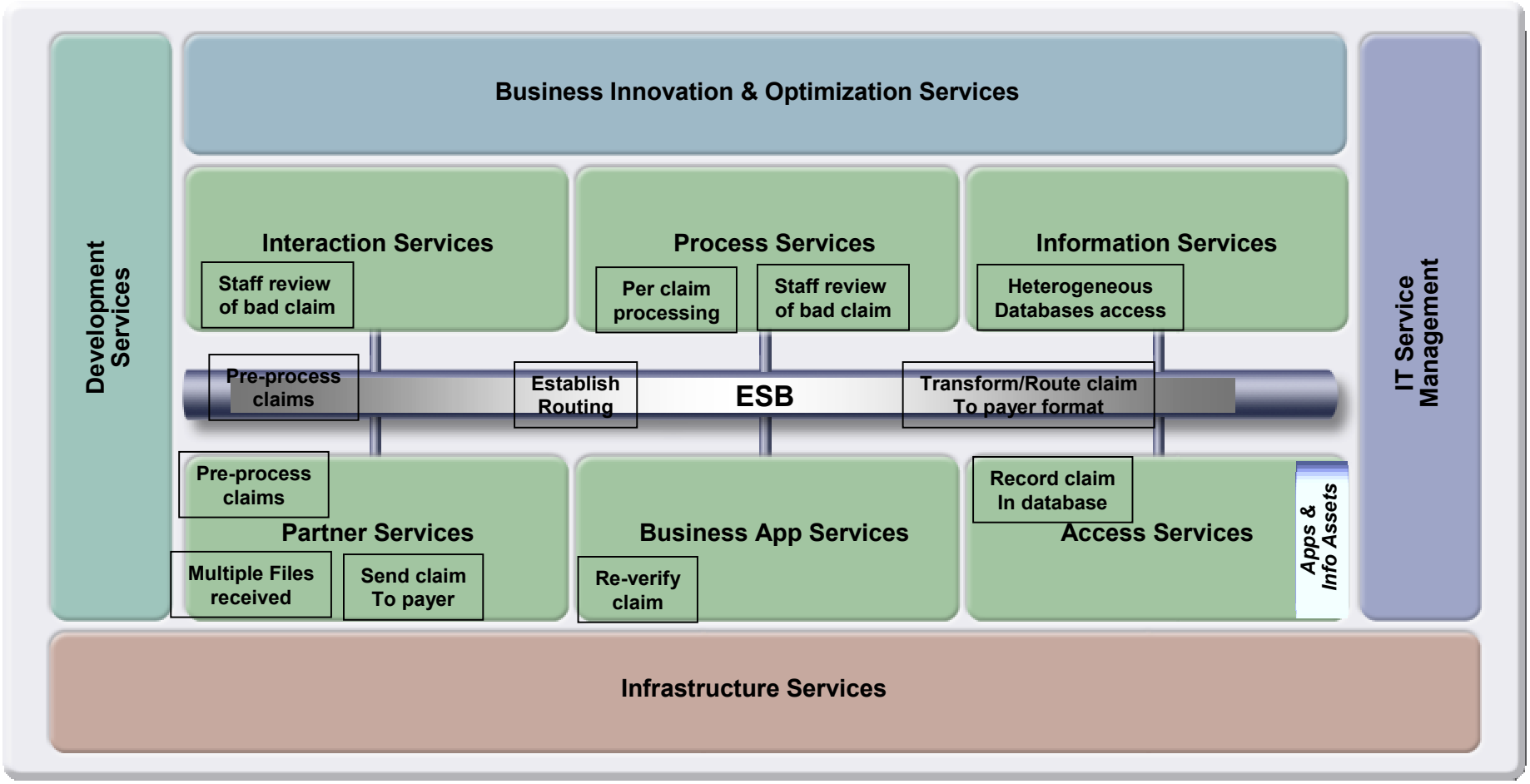
Per claim processing

- Verify target payer
- Data augmentation and route to proper claim processes
  - May require access to different databases
- Post invalid claim to human reviewer (Process Services)
- Staff review of bad claim
- Update claim in database
- Invoke payment process
- Transform claim to payer prefer format (EDI)
- Route claim to payer

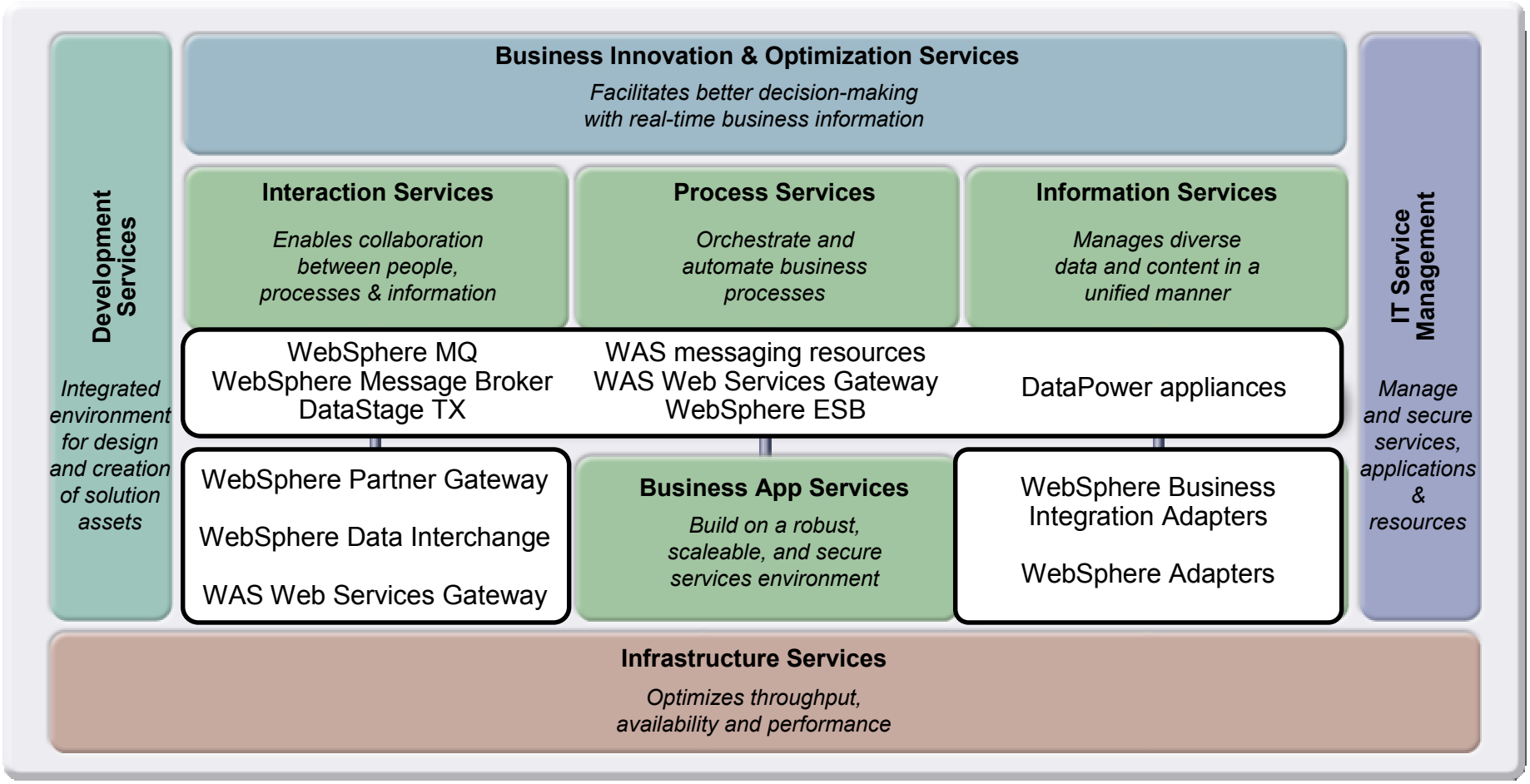
SOA Reference Architecture



SOA Reference Architecture



What is Application Integration … IBM Implementations



Summary of SOA Enablement

- An Enterprise Service Bus (ESB) is the latest stage in the evolution of application connectivity and integration technologies.
- The primary value of an ESB:
  - Cost reduction
  - Business flexibility
  - Progression to an SOA with minimal disruption
- Enabling all applications to participate in an SOA

Q & A

Thank You