MDp Whitepaper

MDp and business integration directions from IBM

David Bonaccorsi, Whitney Sande and Terry Borden

Edition 1
31 July 2000

Note:
Before using this information, read the information in "Notices" on page 69.

First edition (31 July 2000)

Highlights

This document enables you to:

1. Understand the IBM approach to enable MDp migration to the AIM (Application and Integration Middleware) portfolio of products.
2. Understand overall product direction in the AIM portfolio, which supports the above migration.
3. Provide feedback to IBM regarding your company's views of the above plans and directions, with specific relevance to the migration of your MDp implementation.

This document does not constitute an announcement of any product or offering from IBM. Names of products as yet unannounced by IBM may change prior to announcement. Functional content of products as yet unannounced by IBM may change prior to announcement.

Contents

List of figures

Preface

This document is intended primarily to provide background information for MDp customers considering the migration of their MDp implementation to IBM Application and Integration Middleware (AIM).

This document is intended for readers who have some knowledge of MDp function and applications. It discusses plans to migrate MDp function to the portfolio of Application Integration Middleware technologies, such as messaging, messaging applications, message brokering, workflow, process flow and adapters/connectors. This document indicates general product direction and function for new products being developed within the AIM portfolio. The document does not announce any products or provide final functional descriptions. As part of the development process, we solicit feedback from MDp users to ensure that MDp users are enabled to migrate to the AIM portfolio, and to assist the completion of product development.

This document is not intended to provide an in-depth coverage of the above topics. However, it should provide readers with a basic awareness of how MDp function maps to IBM AIM strategic business integration initiatives. It will also provide references to enable readers to access related information.

Readers of this document will be able to understand:

- Why the MDp product group will no longer be enhanced by IBM.
- The strategic IBM Software Group (AIM) technology that will facilitate migration.
- Techniques that may be used to plan migration, while continuing use with MDp.
- Potential migration scenarios to IBM's strategic AIM portfolio

As indicated above, the content of this paper is based on the current WebSphere Software Platform that includes the MQSeries portfolio products and offerings. See www.software.ibm.com. To achieve the document’s objectives, certain unannounced products and offerings are mentioned. Product plans for unannounced products are subject to change. If in doubt concerning any product or offering, please contact your IBM representative.

Comments concerning the content of this document and particularly your plans to migrate to the AIM portfolio should be sent to:

- David Bonaccorsi of IBM at `davebono@us.ibm.com` or
- Whitney Sande of IBM at `sande@us.ibm.com`

Acknowledgments

We wish to thank all those that helped with the content and preparation of this document. Kristof Kloeckner and Graham Tuttle for their contribution to development planning and customer guidance. Peping Fong for her work in coordinating of customer meetings and contact. Many thanks to Graham Charters, Tom Freund, Colin Devonport, Mark Weeks, Michael Poirier, Marc Thomas Schmidt, Michael Beisiegel, Mike Starkey, Geoff Sharman, and our other colleagues in product architecture for their review and insight to this document. Thanks to Bob Flaherty for his help in preparing and editing this document.

Also, a special thanks to our valued customers who hosted our meetings and provided input into the strategy discussed in this document.

1.0 Stabilization of MDp

As IBM enhanced and developed the products within the MDp and MQSeries portfolios, the overlap between the function of the product sets increased. In order to deliver the most appropriate and compelling offering within the business integration marketplace, a decision was taken to focus on one product family. The decision was taken to cease development of function within the MDp family, and offer migration to the AIM portfolio, particularly within the MQSeries product set. Part of the rationale was based on the limitations within the MDp product set, and part on the actual and intended usage patterns of MDp.

Limitations within MDp.

While MDp is excellent in many areas, such as high performance and high volume CICS back-end integration, the MDp portfolio has certain limitations:

- CICS-only environment
- Duplicates rather than uses current CICS capabilities
- Mainframe, 3270-based tooling
- COBOL Applications only
- Limited Recovery Scope
- Does not integrate well with IBM’s other Business Integration products

Usage Patterns

IBM, in August of 1999, commissioned an independent research firm to conduct interviews with current MDp customers to collect feedback and assess actual usage patterns. This research firm gathered information regarding both current and planned usage of MDp. Using this information, and following an IBM assessment of the options, IBM then decided to cease the enhancement of the MDp product and to move toward offering appropriate functional capabilities within IBM's AIM product portfolio, particularly within the MQSeries family of products. IBM then communicated that decision to customers. IBM's strategy is to develop a product offering that capitalizes on the generic strengths of MDp while also integrating that functionality into the AIM solutions.

1.02 IBM’s commitment to business integration and MQSeries family strategy

IBM is a major technology provider in the Application & Integration Middleware market. IBM’s future investments in this area will continue to focus on multi-platform initiatives that enable our customers to implement web-enabled business processes throughout the world. IBM has announced the WebSphere Software Platform (with a focus on the MQSeries portfolio) to indicate this. Refer to www.software.ibm.com for more information. IBM is committed to investment in the MQSeries family strategy, WebSphere, adapters and connectors.

1.03 MDp generic functionality

MDp, as currently structured, exists as a comprehensive end-to-end framework for constructing light weight short duration process flows (Business Requests) which contain "scripted flows" to interact with “legacy” applications.

MDp's high-level functionality can be decomposed into 4 basic functional domains:

- Front-end (Client) and Backend (Server) Connectivity Agents.
- Light weight process control infrastructure (Business Request management).
- Flow scripts (User defined Business Requests).
- Buildtime Infrastructure for constructing and administering Business Requests.

This high level functionality is also addressed in the following IBM AIM middleware:

- IBM's connector/adapter strategy
- WebSphere Software Platform (including MQSeries Family)
- Common Tooling inclusive of process modeling and build-time infrastructure, application development tooling and administration.

1.04 MDp Customer Strategy

Customers should be able to use this White Paper as information and a guideline for migration planning and development of new MDp related implementations.

Customers can expect the following as next steps in the process:

- Development activity of “MQSeries Integrator Agent for CICS Transaction Server”. See below.
- Participation in beta (of MQSeries Integrator Agent for CICS Transaction Server, for example), early acceptance and proof of concept activities.
- To begin migration planning and integration test planning.

Should you wish, professional services will be available from IBM to assist in interoperability/migration planning.

2.0 MDp functional overview

2.01 MDp Strengths

The Message Driven processor (MDp) is a message-oriented middleware product that satisfies the connectivity and interoperability requirements of the any-to-any application server concept. It operates as a server, supporting messaging to and from client applications that support MDp's architected common message set.

One of MDp's most powerful benefits is that it allows for the separation of business functions from the underlying technologies. In other words, it creates an environment in which users can execute business functions, for example, Customer Inquiry, without having any knowledge of the technology or legacy access necessary to perform that function. The user is thus shielded from the technology that handles process management and flows, messaging, protocols and business integration to disparate information systems.

MDp was built on a request-response model. One of its unique attributes is that it contains scripted process flows which can be nested to be executed either synchronously or asynchronously under the control of the MDp workflow manager. The resulting data from the sub-flows can then be reconciled to the reply data with user defined controls to handle error processing and compensation. Further discussion of these more advanced control flow capabilities will occur later in this document.

The following are MDp's key features and benefits:

- Provides full-service, lightweight scripted process flows with transaction support that provides seamless access to back-end legacy applications information allowing customers to leverage these systems without costly re-engineering.
- Built on CICS, a highly scalable, available application server.
- Provides an open client architecture and highly customizable, centralized business rules.
- Provides multiple front-end and legacy connectivity options (including 3270 emulation capabilities).
- Provides a number of process flow management features including:
  - Dynamic Routing
  - Early and Multiple Reply Processing

- Request Continuation
- Synchronous and Asynchronous Processing

2.03 MDp weaknesses

While MDp has certain strengths, it also has market-identified weaknesses.

The following are some key weaknesses attributed to MDp, which IBM plans to improve in its implementation of the new strategy:

- Development environment limited to 3270 green-screen, lack of a GUI based development environment.
- Development environment not seen as flexible where logical message definition and modeling capabilities are needed.
- Data Dictionary is not easily portable or usable in other run-times.
- Data Dictionary format limited to COBOL.
- Dictionary Definitions could not be imported easily from already existing COBOL RD's.
- Dependence on S/390 CICS resources (for example, VSAM).
- Duplication of features which have been subsequently added to CICS.
- Limited facilities for source code management.
- Production versioning and implementation require manual and external processes.
- Application development was limited to COBOL/CICS.
- Simple command style interfaces were not possible in MDp.
- The Store and Forward facility requires comprehensive analysis and application knowledge to implement.
- The Queue Management facility is limited to a predefined VSAM structure with a limited alternate key structure.
- System and Process Monitoring facilities require manual and external interfaces.
- Problem Determination requires additional system integration and manual processes.
- Limited integration with other key IBM Business Integration Middleware.

2.04 Transition of MDp function to IBM strategic middleware

IBM intends to leverage the strengths of MDp while also improving on its weaknesses by integrating that function, where appropriate, into the AIM family, particularly the MQSeries product family.

The first instance of MDp functional domain migration to the MQSeries Family is the development of the currently named MQSeries Integrator Agent for CICS Transaction Server, from this point on referred to as MQSI Agent for CICS. This product will capitalize on the legacy connectivity and optimized transactional process management flow capabilities that were available with MDp. Additionally, it will benefit from IBM's common tooling for business integration. IBM's common tooling for business integration will provide a define-once and deploy-many capability for process and message flows. The MQSI Agent for CICS will be one of the deployment options for this tooling.

The middleware run-time (the selected operational environment) will depend on the implementation requirements of each customer and could include one or a combination of the following of IBM's Application Integration Middleware products:

- MQSeries
- MQSeries Workflow
- MQSeries Integrator
- WebSphere Application Server
- MQSI Agent for CICS
- MQSeries and other adapter offerings
- Future IBM Adapter Offerings

In order to view the position of these products within the AIM WebSphere Software Platform, please go to the following url reference: www.software.ibm.com.

3.0 IBM’s commitment to a business integration strategy

3.01 Statement about WebSphere Software Platform

The WebSphere Software Platform describes, positions and explains the value of the IBM offerings of tools, web application servers and middleware which assist many customers and partners create the kind of fast, fearless and flexible e-business applications to succeed in the new economy. These offerings help the customer:

- Quickly build and deploy responsive, expandable, and innovative Web sites.
- Plug and play your business with others, including suppliers, distributors, customers and corporate acquisitions.
- Maintain and grow your customer relationships.
- Transform your business to keep pace with changing demands.
- Handle your needs regardless of your industry or business issue.

Within the WebSphere Software Platform you will see specific focus on the MQSeries portfolio of products, key products for integration and process automation.

3.02 Overview of IBM’s Transformation and Integration middleware

IBM defines business integration as the alignment of the business process with Information Technology. This includes integration of all processes that operate within a business to produce a unified, complete, and consistent view of the information needed to complete any business transaction. It means the creation of solutions that integrate diverse systems and applications to operate as one enterprise-wide business solution.

Aberdeen Research has discovered that these processes are typically expressed through *disparate* information technology systems that exist within a company and that always exist when extending across companies—systems that were not originally designed to provide such a tightly coupled enterprise view.

The challenge is the integration of these diverse IT systems to present a complete and consistent view of pertinent business processes to customers, partners, and employees. To help customers meet this challenge, IBM has introduced a series of products anchored by the MQSeries software product family that are all built to the specifications of the IBM Application Framework for e-Business. See the figure “IBM’s MQSeries family in application integration middleware” on page 17.

MQSeries

MQSeries base is a messaging product that provides fast and reliable information delivery across a variety of platforms and systems.

The MQSeries Integrator product is a central software hub that brokers messages between applications. It provides a complete solution to the problem of transforming, formatting, and routing vital business information between dissimilar application systems.

The MQSeries Workflow product allows companies to streamline their business processes. It can be used as a powerful mechanism for process automation and the triggering the business processes by both IT systems and people.

In addition to the MQSeries Family of products, IBM offers two other products within the WebSphere Software Platform that will help customers achieve their strategic business integration objectives. These are WebSphere Application Server and VisualAge.

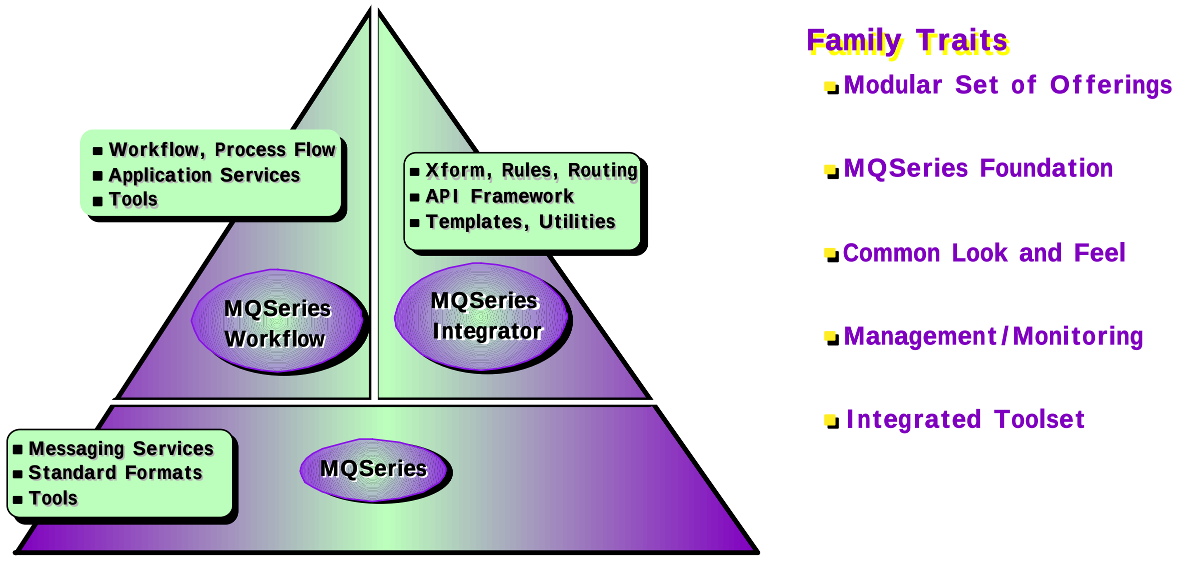


Figure 1. IBM’s MQSeries family in application integration middleware

WebSphere Application Server

WebSphere Application Server provides an open, standards-based Java runtime environment along with development tools and management software.

It has three editions to help customers develop simple Web publishing applications and transition to the development of advanced e-Business Web applications. It provides fine-grained interoperability through the use of technologies such as reusable JavaBean components. It supports Enterprise JavaBeans components (EJB) which are a set of extensions to the JavaBean component model to support scalable, distributed, multi-user, transaction oriented business applications. It provides for information sharing through the use of the Web standard, eXtensible Markup Language (XML).

VisualAge

The VisualAge product family offers an Integrated Development Environment (IDE) for a large assortment of programming languages.

Both WebSphere and VisualAge are integrated with the MQSeries product family, enabling the rapid development and deployment of MQSeries applications.

See "Appendix A. IBM MQSeries family of products" for details. Refer to the MQSeries Web page, www.ibm.com/software/ts/mqseries, for additional information.

3.03 IBM's adapter/connector strategy

Within the IBM AIM strategy, there also exists a complementary strategy for connectors and adapters. This is described in the section below.

3.03.01 Overview

There are currently development initiatives within IBM that provide connector and adapter technology. These offerings provide varying degrees of connectivity to and from IBM's strategic Application Integration Middleware and Application Server environments. What has been recognized within IBM development is that in order to move forward with a consistent development paradigm for this technology, the following requirements should be met:

- Consistent terminology and definition for what constitutes an adapter or connector.
- Standards conformance (where possible and practical).
- A consistent architecture/framework and domain responsibility for the components.
- Reuseability of customers' existing application infrastructure.
- A consistent meta-model/programming model and tooling for developing, configuring and deploying adapter/connector components.

This section and "Appendix B. IBM's Common Connector Framework" on page 61 will briefly describe IBM's adapter/connector strategy with particular attention and reference to architecture, frameworks, existing connectors and adapters for meeting specific needs.

We will also provide some initial exposure to "current" early implementations in this space, namely:

- MQSI Agent for CICS that assists in the migration of MDp users. (This implementation was previously known as the Message Adapter for CICS.)
- IBM MQSeries Adapter Offering. This works with MQSeries messaging to enable you to reduce the risk, complexity and cost of managing the point-to point integration of your business processes. The MQSeries Adapter Offering provides a framework for building "C" based ERP style adapters. See "Appendix D. IBM MQSeries Adapter Offering". For additional information, see www.ibm.com/software/ts/mqseries/announce/complete/adapters.html

Both offerings rely on the principles and architecture of IBM's adapter/connector and business integration strategy.

3.03.02 Adapters and Connectors

Our recent customer experiences and participation in IBM/industry business integration technology forums have only reinforced the thinking that the terms "connectors" and "adapters" denote different levels of understanding in the IT and development communities. This leads to confusion of definitions and misunderstanding of the functional boundaries and responsibilities in the adapter and connector.

In this paper we will use the following definitions that are based on current thinking within IBM’s adapter/connector architecture forums:

adapter

An adapter is a component that performs data transformation and/or flow composition on data to or from connectors. This activity can include:

- Connector flow inclusive of conditional or Boolean logic for sequencing or navigating connector API access on data flowing to and from connectors.
- Data transformation capability for API mapping to disparate data types, messages and interfaces.

An adapter usually uses a *connector* via an API to access a target system. In this way it can be thought of as containing one or more connectors and is capable of API mapping.

connector

A connector is a lower order "plugable" component (usually part of a server) that provides API mapping from the server to the exposed API of the target system. In this way, the connector moves information between the environments and abstracts the underlying protocol, transport and semantics. The connector’s plugability is provided via an API in the framework.

The connector is a type of component that interfaces to the target application system’s native API. The connector also exposes a client API that can be used by other components, such as interactions or EAB commands (see the note below) and navigator-like microflows on the server. See “Appendix C. Flow composition model” on page 65.

Together adapter and connector components work in a framework to create “higher order” middleware application services, for example, Customer Inquiry or Process Order.

Note: An example of such a framework is IBM’s CCF/EAB. See “Appendix B. IBM’s Common Connector Framework” on page 61 for details.

The figure illustrates an adapter flow.

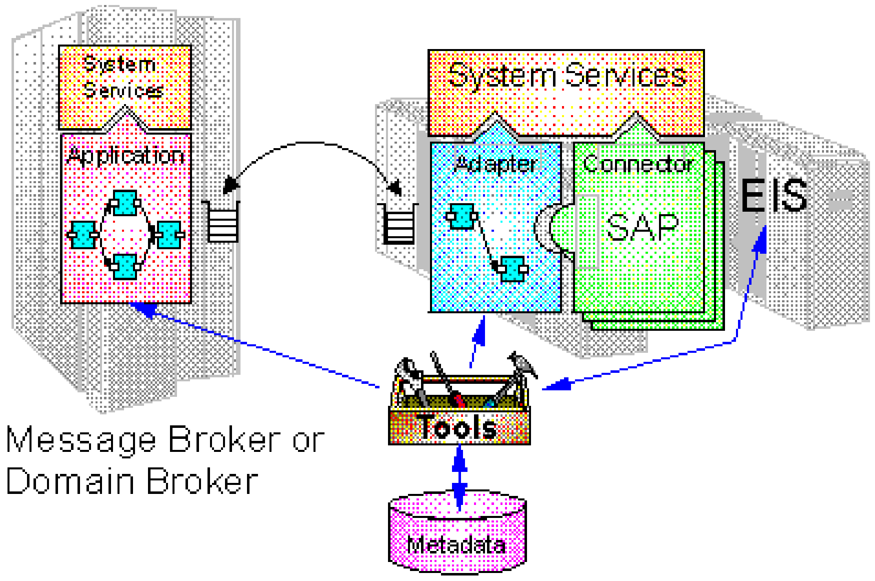


Figure 2. An adapter flow

Connectors and adapters play an important role in eBusiness business-to-business integration and are gaining significant focus within IBM. An important future requirement is that they work compatibly with IBM's strategic direction for Application Integration Middleware and business integration strategy. Also, it should be noted that the current connector offerings will be enhanced to adopt common tooling, common meta-model definition and evolution to a common framework based on flow composition and the J2EE Connector Architecture Specification.

Connectors and adapters, by nature, solve the problem that many application programmers struggle with: the need to consistently interact with disparate data sources and applications. Connectors and adapters assist the business integration efforts by attempting to free the application programmer from underlying complexities of heterogeneous computing environments. These complexities tend to be differing data representations, access mechanisms, protocols, and application interaction semantics. In order to accomplish the goal of isolating the programmer from the complexities of a heterogeneous computing environment, a consistent and unified the framework is necessary. This framework seems to require some or all of the following:

Connectivity

Via a uniform access to external systems and the means to map input/output data to underlying connections with transparent access to underlying transports. See "Appendix B. IBM's Common Connector Framework" on page 61 for details.

Data transformation and data mapping capability

Provides for a unified view of the information model in applications requiring mappings from disparate data type and interfaces to a common or canonical format (for example, XML for defining specific data patterns).

Note: XML specifically can be used in connectors at multiple levels such as configuration, deployment information, data representation and process/activity interactions.

Activity Definition Facilities

For capturing the behavior and relationships necessary to interact with target backend application and data sources. For example, a scripted microflow for Order Entry in an ERP scenario, or for Balance Inquiry in a banking environment.

In order to accomplish flow definition within the adapter, a common model for "flow composition" is necessary for defining generalized artifacts and behavior within the adapter framework. The generalized flow composition model and appropriate tooling allow a uniform (language neutral) means for components to be "plugged into a microflow consistently. The model applies similarly to flow definition as required by MQSeries Workflow and MQSeries Integrator. By providing modeling and definitional facilities for adapter building, the sequence or navigation required in the microflow to interact with one or many backend application systems is expressed in the meta-model. We believe this is a key customer requirement in order to fulfill the define-once, deploy-multiple paradigm.

One such connector framework available from IBM is the Common Connector Framework (CCF) and Enterprise Access Builder (EAB) model. While this was the first generation step in achieving a common architecture and services for connectors, follow on activity will be based on work effort that IBM is currently organizing around Common Connector Architecture (CCA) and the J2EE Connector Architecture Specification.

(For more information on J2EE, see http://java.sun.com/j2ee/connector.)

See "Appendix B. IBM's Common Connector Framework" on page 61 for details.

3.04 Flow Composition Modeling

Future IBM tooling architecture described briefly this document is built on a common architectural meta-model known as the flow composition model. The notion of flow composition requires a model that describes a runtime-neutral framework for describing complex business processes that are made up of elemental components known as *flow components*. Using the model “business services”, an entity that performs one or more business functions, can be composed into higher level entities known as flow models. The Flow Composition Model serves as the meta-model underpinning (XML artifacts, constructs, primitives) to the notion of flow composition in future IBM tooling strategies.

As identified in “2.0 MDp functional overview” on page 12, one of MDp’s key weak points is the need for a GUI builder environment. In the future, a formal model and graphical builder will be used to build and deploy business flows that can be realized into multiple deployments. Follow-on products to MDp will utilize this technology for use in transformation and integration efforts.

See “Appendix C. Flow composition model” on page 65 for details.

3.05 IBM MQSI Agent for CICS

MQSI Agent for CICS [1], known previously as the Message Adapter for CICS (MAC) or Message Adapter, represents the initial stage of evolution from MDp’s technology into the IBM MQSeries portfolio, and the adapter and connector architecture. It will enable any MQSeries-enabled application or any application that supports CICS external call interface (ECI) to access:

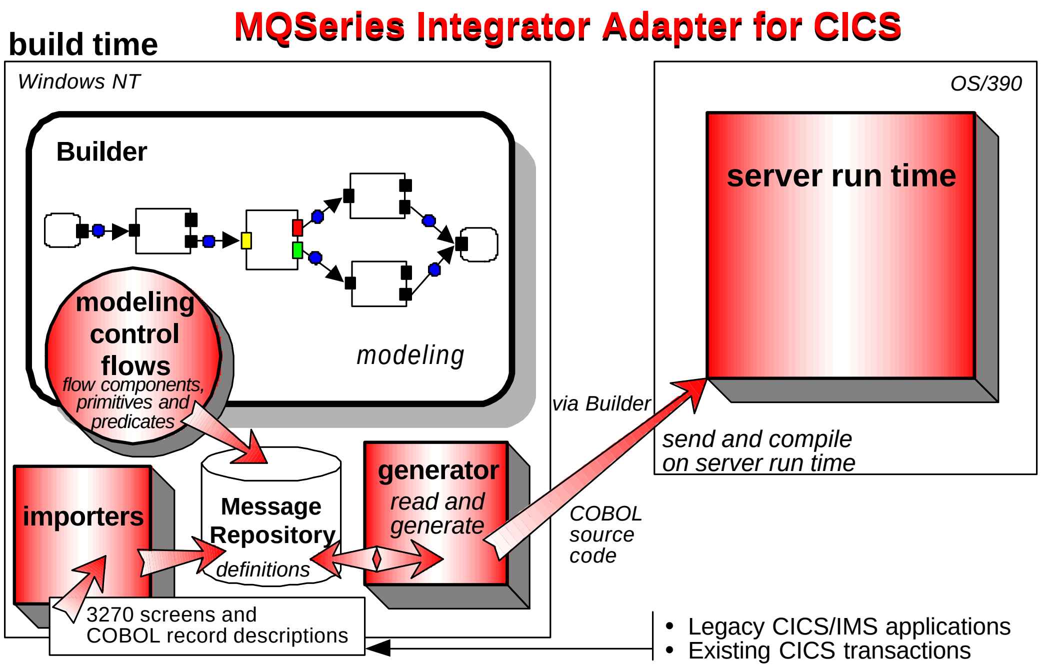
- Existing CICS transactions via a Distributed Program Link (DPL).
- Legacy CICS/IMS applications via a 3270 data stream.
- MQSeries-enabled applications via MQSeries.

The MQSI Agent for CICS has two environments: *build time*, under Windows NT and *run time* server, under OS/390 as a CICS application using CICS/BTS facilities.

See the figure “MQSI Agent for CICS build time” on page 23. In build time, the access is modeled and defined by means of a natural evolution of the Control Center component of IBM’s MQSeries Integrator (MQSI) product. It becomes a member of IBM’s common tooling for business integration that

[1] The name might be changed before it is released as an IBM product.

provides a define-once and deploy-many capability for process and message flows. This evolution of the control center also supports the MQSeries Integrator Adapter's modeling of microflows. The MQSeries Integrator Adapter's notion of modeling control flows consist of a group of flow components, primitives and predicates specified in the common flow composition model.



build time
MQSeries Integrator Adapter for CICS
Windows NT
Builder
modeling control flows
flow components, primitives and predicates
modeling
Message Repository
definitions
importers
generator
read and generate
3270 screens and COBOL record descriptions
via Builder
COBOL source code
OS/390
server run time
send and compile on server run time
• Legacy CICS/IMS applications
• Existing CICS transactions

Figure 3. MQSI Agent for CICS build time

The Control Center's tooling will be enhanced from MQSeries Integrator current capability of composing message flows which has publish and subscribe nodes to include the MQSeries Integrator Adapter's microflows. Modeling is supported by *importer* facilities for importing 3270 screens from legacy CICS/IMS applications and COBOL record descriptions from existing CICS transactions, making them available during modeling. The definition of the model is stored in XML in a common message and flow dictionary and will follow the evolution of the IBM common tooling strategy.

A *generator* facility reads the model's definition from the message repository. The generator also reads static templates that contain the portion of server run time that is never affected by modeling. The generator then transforms them and generates COBOL source code. The source code is sent to the server and is compiled into the server run time. In short, the generator enables the server run time to behave as modeled in the build time.

The runtime environment for this adapter executes under CICS on OS/390 and will be capable of operating in a CICSPlex/SYSplex environment.

See the figure “MQSI Agent for CICS run time”.

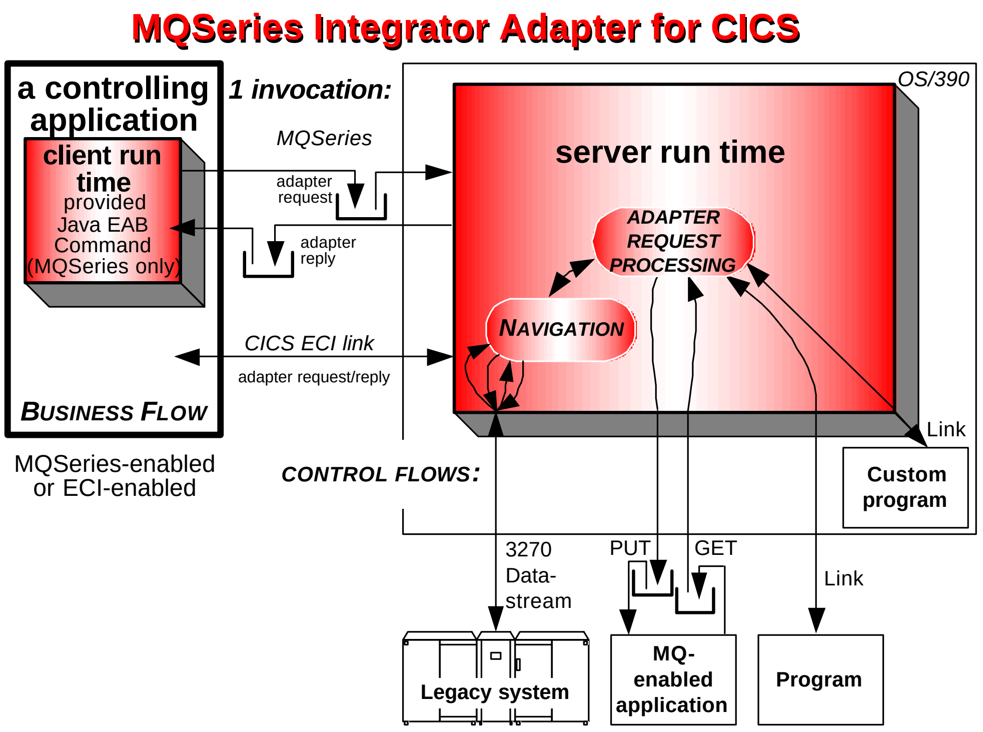


Figure 4. MQSI Agent for CICS run time

MQSI Agent for CICS supports *microflow “navigation”* that includes simple business logic. Connector flow and respective navigation is a scripted representation of the interaction with backend data sources (i.e., CICS transactions, legacy CICS/IMS applications, MQSeries-enabled applications and custom programs). Adapter request processing provides sequential navigation and conditional branching but not parallel asynchronous branches, waits or joins. Connector flow processing is modeled during build time.

This model conforms to the CCA, flow composition model, (tools and metadata) for building connectors and adapters.

Controlling application

MQSI Agent for CICS server run time is invoked from the *controlling application.* The controlling application is any application that controls business flow and that invokes MQSI Agent for CICS server run time via MQSeries or a CICS ECI link.

Examples of controlling applications could be hosted by:

- MQSeries Integrator

- MQSeries Workflow
- WebSphere application server
- Any local or remote CICS application that is ECI-enabled

Note: All controlling applications, regardless of how they invoke the MQSI Agent for CICS server, must use the MQSI Agent for CICS header (defined in the documentation for the product) in order to

- Function at all or
- Be compatible with future releases of the MQSI Agent for CICS.

A controlling application can manage business context, complex state, multiple requests and replies, asynchronous processing, and the continuation of one logical request through multiple requests, if so required. In short, the controlling application is responsible for the overall business flow.

In contrast, the MQSeries Integrator Adapter run time is responsible for a step in the business flow that requires legacy application navigation. The adapter contains microflow.

Invocation and state management

If the controlling application invokes the MQSI Agent for CICS via MQSeries then the MQSeries-CICS Bridge will provide the interface between MQSeries and the MQSI Agent for CICS. For more information on the MQSeries-CICS Bridge, see MQSeries Application Programming Guide SC33-0807-09.

The MQSI Agent for CICS uses CICS Business Transaction Services (CICS/BTS). Under the control of CICS/BTS, each adapter request message initiates a new instance of a MQSI Agent for CICS process which is a BTS process.

As MQSI Agent for CICS run time is invoked, it interacts with the CICS transactions, legacy CICS/IMS applications, MQSeries-enabled applications or custom programs. Each invocation of MQSI Agent for CICS run time is:

- Synchronous,
- In a single request/single reply or a single request/no reply model.

Except in the case of failure, MQSeries Integrator Adapter run time manages only enough state information required to support the work to satisfy one invocation. Once the invocation is completed, this state information is no longer retained. State information is not maintained across multiple invocations of MQSeries Integrator Adapter run time.

In the case of failure, MQSeries Integrator Adapter retains state information and application data for subsequent use in a compensation flow.

Connector flows

The MQSeries Integrator Adapter run time performs these four types of microflows:

1. Initiation of programs via one or a sequence of DPLs, via CICS LINK.
2. Navigation of 3270 screens. MQSeries Integrator Adapter can conduct an interactive 3270 request and reply dialog with legacy CICS or IMS applications. Via IBM’s FEPI product, it sends requests to and receives replies from any CICS/IMS application whose 3270 datastream is intended for a 3278 Model 2 terminal (24 rows by 80 columns), that is, the returned buffer in a single send/receive is not greater than 3600 bytes.

   MQSeries Integrator Adapter screen navigation flow:

   — Begins the FEPI session,

   — Parses screens sent by the legacy CICS/IMS application,

   — Identifies the screen and its fields, attributes and data,

   — Constructs and sends an appropriate reply, based on the modeling and on simple business logic,

   — Handles the next screen by parsing, identifying and constructing a reply or keystroke, and so on,

   — Manages state information about the status of LUs, and

   — Ends the FEPI session.

   Screen navigation is modeled at build time.
3. Synchronous MQSeries PUT and GET command processing. In other words, in response to each PUT, MQSeries Integrator Adapter expects either a single reply or no replies. This approach enables the controlling application to retain responsibility for the overall business flow.
4. Initiate custom programs via CICS LINK. Custom programs can contain complex rules such as logic and complex I/O. A custom program can be developed to augment the MQSeries Integrator Adapter. (Note that the mechanism to invoke the custom program

is exactly the same as the mechanism to invoke a microflow with CICS transactions, that is, DPL.)

Adapter request processing

By means of an adapter request message, the controlling application can invoke MQSeries Integrator Adapter once and then, via a modeled script, it can invoke one or multiple connector-flows. MQSeries Integrator Adapter can invoke multiple connector-flows without requiring action by the controlling application. Within the bounds of the work to process the same adapter request message, MQSeries Integrator Adapter can use the result of one microflow in another microflow.

The script of the microflows is called navigation.

For example, adapter request processing could consist of the following sequence:

1. Invoke a microflow with a legacy CICS/IMS application via a 3270 data stream and retain the result. This microflow consists of screen navigation.
2. Invoke a microflow with an MQSeries-enabled application and post the result of the first microflow to an MQSeries queue, perhaps expecting no reply.
3. Invoke a microflow with one or more CICS transactions via DPL and send the result of the first microflow. For example, it could write a record to a DB2 database.
4. Invoke a custom program via DPL that executes complex rules such as business logic or complex I/O.
5. Send an adapter reply message to the controlling application. It could contain the final result.

Client run time

The MQSI Agent for CICS includes optional client run time software in the form of a Java Enterprise Access Builder (EAB) command. MQSI Agent for CICS client run time performs a portion of the functions that the controlling application must perform. Under the direction of the controlling application, it

- Prepares and sends adapter request messages that invoke MQSeries Integrator Adapter server run time and
- Receives adapter reply messages and provides their application data and relevant control information to the controlling application.

The MQSI Agent for CICS client run time can be integrated with controlling applications. The MQSI Agent for CICS client run time is MQSeries-enabled. The controlling application itself is not required to be MQSeries-enabled as long as it is integrated with the MQSeries Integrator Adapter client run time.

If the controlling application is not integrated with the MQSI Agent for CICS client run time then it will need to perform the functions of the client run time.

Additional capabilities

MQSI Agent for CICS provides additional capabilities:

- Error logging.
- Support for compensation flows. The support includes journal logging. Compensation itself is modeled in the Control Center.
- Support for auditing, which can be enabled and disabled via modeling in the Control Center. Auditing is a function of BTS. The audit level is controlled by a BTS processtype passed in the adapter request message. The audit level determines the audit points.
- Security, which can be implemented in one of the following two ways:
  - — The controlling application treats the MQSI Agent for CICS as a trusted link. The controlling application takes full responsibility for security.
  - — The MQSI Agent for CICS is not treated as a trusted link. Security is implemented in the MQSeries Integrator Adapter server environment in addition to any potential security in the controlling application.

In general, MQSI Agent for CICS build time will be enabled for national language support but will be provided in English only. MQSI Agent for CICS run time's error, warning and informational messages will be enabled for national language support (NLS) but will be provided in English only.

The MQSI Agent for CICS for CICS runtime environment should also have good infrastructual recovery facilities and is being designed to be WLM/CICSplex/Sysplex-enabled.

MQSI Agent for CICS run time as provided includes no graphical user interface, BMS user interface or reports.

4.0 MDp functionality mapped to IBM’s AIM strategy

This section describes the MDp solution’s functional component boundaries and examines how this functionality can be mapped to the various other products within IBM's strategy for Application Integration Middleware and adapters/connectors.

For reference throughout section 4.0, see the figure “MDp base function: build time and run time” on page 31 and the figure “Overview of IBM MQSI Agent for CICS” on page 33.

4.01 MDp base function areas

At a high level we have observed that MDp’s base functionality can be isolated to the following domains of function:

A **build time environment** consisting of

- Build time scripting/definition environment containing message/script definition facilities and debugging facilities.
- Configuration, simulation and runtime utilities.

A **runtime environment** consisting of

- Front-end (Client) agent and Back-end (Server) connectivity agents.
- A process flow management framework for runtime processing of business requests.
- User defined scripted flow scripts (Business Requests) that allowed for customization exits.

See the figure “MDp base function: build time and run time”.

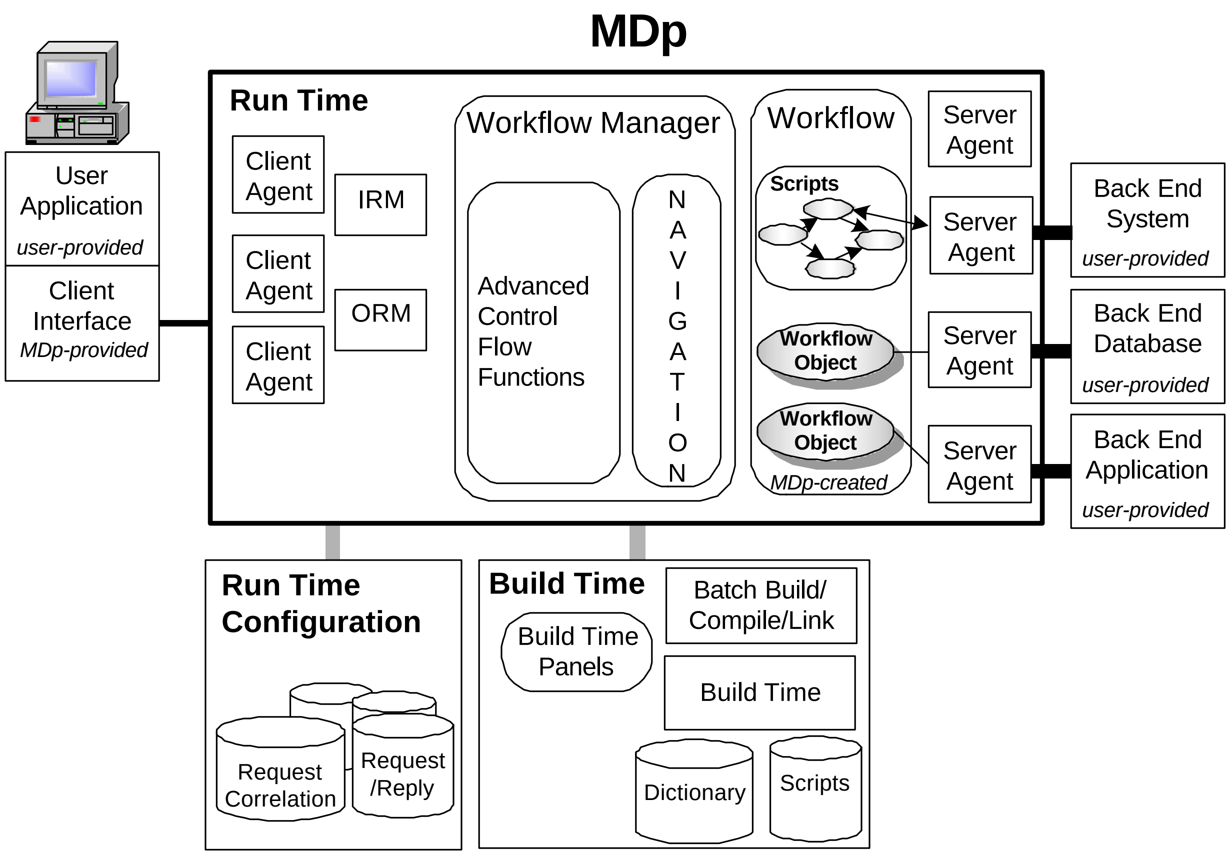


Figure 5. MDp base function: build time and run time

4.02 MDp build time functionality

The current MDp development environment is 3270 based and allows for:

- ***Definition and capture*** of both message interfaces and workflow scripts and workflow script object behavior.
- ***Generation facility*** for generating, compiling and then executing/testing the generated scripts.
- ***Customized user exits*** can be included within the application development environment
- ***Debugging and simulation*** support for test purposes
- ***Configuration and administration*** tools as required to capture runtime parameters/profile support required for runtime administration.

While this environment is an excellent framework for building complex process flows or “transactional” flows and allowed for good productivity enhancement, we have seen that the current MDp build time environment was considered one of the bigger product weaknesses due to the lack of a GUI and modeling environment.

4.03 Future IBM/AIM build-time functionality

In future IBM/AIM offerings, the design goal is a common model (flow composition model) and common tooling for:

- Modeling/building of flows
- Application development tooling
- Configuration and administration
- Systems management

See the figure “Overview of IBM MQSI Agent for CICS” on page 33.

The ultimate goal for initial deliverables of these tools is conformance to the strategy for IBM common tooling directions.

- The development environment (known as the *adapter builder*) for the MQSI Agent for CICS is GUI based and should contain the following:
  - ***Definition and capture:***
    - Common information model for logical message and interface definition in XML meta-data.
    - Flow model and builder for defining business process flows and component relationships.
    - Importers for 3270 and COBOL record descriptions.
  - ***Generation Facility:***
    - Generators for building/generating the adapter business service (similar to an MDp business request).
  - **Customized User Exit Processing**:
    - This can be accomplished by using the “component” methodology described in “5.03.01 Preferred usage techniques” on page 42 and the DPL capability of the MQSI Agent for CICS.
  - ***Configuration and administration tools:***
    - Some utilities for defining runtime parameters or properties.

The ability to model the adapter is viewed as a substantial improvement over the MDp development environment because business function can be modeled and defined in the meta-data (XML) in a language neutral format. This allows for multiple realizations (deployments) of the model. For example, the same modeled Customer Inquiry flow could be deployed in a CICS COBOL environment today and deployed to an EJB environment in the future.

The MQSI Agent for CICS development environment should be recognized as a substantial improvement over MDp. It allows for component based development, reuse and deployment.

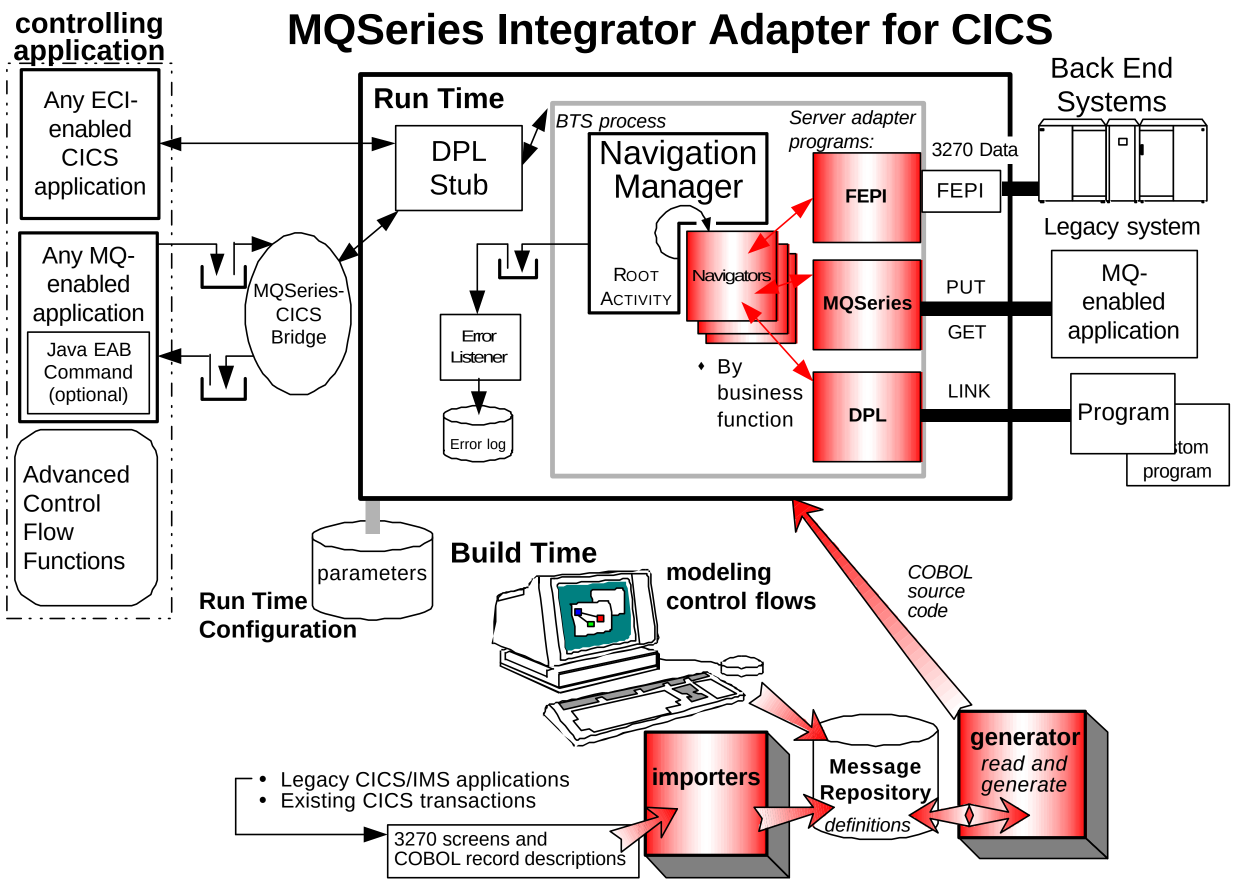


Figure 6. Overview of IBM MQSI Agent for CICS

Contrast this figure with figure “MDp base function: build time and run time” on page 31.

It should also allow for reuse of MDp business requests (where possible and practical). See “5.03.01 Preferred usage techniques” on page 42.

4.04 Anticipated future utility and simulation function

It is anticipated that some facility for defining the necessary parameters, properties, resources, etc. will be provided with the MQSI Agent for CICS. This may be tactical initially, since the preferred direction will follow standard conventions for the common tooling and administration direction for the MQSeries family and the common tooling strategy. We will also attempt to utilize the RAS services such as trace, log, policy, etc. within the existing IBM/AIM runtime services where possible.

It is also anticipated that a simulator and potentially a debug utility will be provided, at least as a services tool kit. We would expect to refine our assumptions with customers during the MQSI Agent for CICS beta period.

4.05 MDp run time functionality

The existing MDp runtime facility allows for the following high level function:

Front end/back end connectivity agents

For interfacing from and to disparate client and server protocols. MDp provided MQSeries, LU2, LU6.2, TCP/IP as front end agents, and MQSeries, LU2, LU6.2, CICS peer to peer techniques as back end agents.

Workflow Management environment

A canonical or common processing environment was provided for Business Request execution. Business Request execution was over an MDp architected message set, state model and memory management framework.

Business request execution was high volume, proven, supported multiple interaction patterns such as request/reply and multi-reply.

User defined business request scripts

A “stateful” environment for defining scripted control flow for creating a new business service by interfacing to existing legacy systems and data sources. Business rules and conditional navigation could be contained in the flow. The scripting model allowed for user custom programming exits.

MDp also supports what we have termed *“advanced flow control functionality”*. See “4.07 Advanced flow control functionality” on page 37.

4.06 Future IBM/AIM runtime functionality

Within the WebSphere Software Platform there are facilities and offerings to control workflow or create application control flow. The precise implementation characteristics to completely migrate any MDp implementation, including the existing business requests, is beyond the scope of this document. However, as the products within the WebSphere Software Platform are developed, function will become available to assist in further migration of generalized MDp workflow manager functionality.

The target runtime services mapping for existing MDp runtime model is as follows:

Frontend/Backend connectivity agents

The future direction for front end and back end agents will reside in IBM's adapter/connector strategy. This will include the current CICS/MQSeries gateway and bridge support. Future offerings evolve toward Common Connector Architecture (CCA) and the follow on functionality of the Common Connector Framework (J2EE Connector Architecture Specification). See "Appendix B. IBM's Common Connector Framework" on page 61. The new implementations should support the strategic IBM application server environments as well as key ISV application environments (such as ERP and B2B) in the future.

The first release of the MQSI Agent for CICS will support these "inbound" methods of activation:

- Java Command bean with MQSeries
- MQSeries
- ECI

The following back end connectors will be supported:

- CICS DPL
- MQSeries (synchronous PUT/GET)
- LU2 via CICS/FEPI

Workflow Management environment

The current MDp workflow management environment will be replaced by the functionality in the MQSI Agent for CICS and whatever target runtime is identified for handling the advanced flow control functionality. Further discussion occurs in "4.07 Advanced flow control functionality" on page 37.

The adapter will use:

- The general MQSeries Message header and the MQSeries-CICS bridge process and structure (MQCIH). BTS services in Transaction Server 1.3 for recoverable adapter state management (BTS containers) and activity management.
- Flow navigation capability, built in navigation support (general adapter flow/navigation), will be packaged with the Transaction Server 1.3 runtime and packaged in later releases of Transaction Server 2.x. This will be a CICS/TS deployment option.
- User defined navigation (microflow), inclusive of simple business rules, is modeled using common tooling and then generated for execution in the CICS runtime. See "3.05 IBM MQSI Agent for CICS" on page 22. The MQSI Agent for CICS Builder will initially allow the modeling and definition of the following scripted interactions (microflows) to back end systems:
  - CICS DPL dialogs
  - 3270 via FEPI dialogs
  - MQSeries synchronous dialogs

  It is planned that scripts (microflows) will also be generated more comprehensively than the MDp builder environment since field level mapping and simple business rules can be modeled and held in the meta-data. This allows for generation of mapping statements and simple business logic for sequencing microflow navigation.
- Generalized error logging support. Customization of this component to interface to systems management facilities such as Tivoli is possible.

4.07 Advanced flow control functionality

The following functionality was supported by MDp in varying degrees but will not be supported in the MQSI Agent for CICS. These functional requirements are targeted to be absorbed, based on the generalized customer needs, in the MQSeries family or WebSphere family of products. These requirements can be thought of as "workflow" or "business process flow" instance requirements.

- Parallelism/Asynchronous processing: the ability to activate, control and coordinate simultaneous parallel units of work from a single client request.

- Multi-request/reply model: the ability to process, control and coordinate multiple request/reply patterns from a single client and/or single client request
- State Management: the ability to hold state information across client requests that is useful to application logic and the business process.
- Compensation: At the business process level, coordinated through modeling.
- Continuation: At the business process level, coordinated through modeling.

For purposes of this document, we will refer to the above functions as *advanced control flow functionality*.

Before this advanced control flow function can be included in the AIM Family Strategy, further qualification of the user requirements are needed to understand the generalized usage patterns anticipated to be deployed by current users. We received mixed reviews in our current analysis. We anticipate these requirements will exist at varying degrees of granularity by customer. MDp customers are expected to have input to these requirements. Additionally, customer input will help IBM determine where this advanced control flow function best maps on to IBM's middleware strategy.

Note: In the course of our customer visits we received requirements for the MQSI Agent for CICS runtime to support the patterns of “early reply”, continuation and compensation. Continuation and compensation would only be considered within the adapter scope and would need to be coordinated by the business process. These requirements are necessary when deployed in current customer service/CRM environments. We are evaluating the design considerations of this functionality.

4.08 Additional requirements and issues with the MDp mapping

Through the course of our recent interactive customer meetings we recorded some remaining issues, that are open and will require further investigation. Customers need to consider the following questions.

These requirements fall within the following areas:

- Do you have a requirement for what is termed as advanced flow control functionality. See “4.07 Advanced flow control functionality” on page 37. As stated above, while these functions were interesting attributes of MDp, many customers were not able to implement them. Also, MDp functionality/model, though rich, was not granular enough to deploy "simple" connector requirements.

- What additional system administration tools and utilities will be required to support the target deployment options for the typical MQSI Agent for CICS scenarios? While we are planning to deploy some of these tools with MQSI Agent for CICS, the strategic direction should be toward common systems management, administration, etc. for IBM/AIM products. The direction should also to leverage any generalized RAS support, administration, configuration and policy definition that exists within the hosting application server.
- What additional tools/capabilities, if any, are there to support MDp user exit migration and interoperability? While we have formulated an approach to MDp migration, covered in "5.0 How do I get there?" on page 40, this will require manual analysis and professional services. We will investigate some tools that may assist in the analysis of MDp user exit content.
- Many customers have expressed interest in continuing to develop and support OS/390 MDp-style middleware solutions for the near term, in COBOL. The first deployment for MQSI Agent for CICS runtime is OS/390 COBOL/CICS. However, we did register interest in Java and XML from customers through the course of our recent analysis. IBM Connector and Adapter directions will trend toward Java J2EE Connector Architecture Specification and XML, as well. We would like to understand customer deployment plans and quality of service objectives for these newer technologies at the CICS/TS 2.x level and WebSphere Application server.
- What supplementary services offerings are needed to assist customers in the following areas:
  - — Planning to continue using MDp for next few years but want to comply with preferred development techniques to ease migration in the future. See "5.0 How do I get there?" on page 40.
  - — Need initial assistance to deploy the MQSI Agent for CICS as a standalone project (coexistence mode) for proof of concept work or initial migration.
  - — Need to integrate MQSI Agent for CICS with existing MDp business requests in order to reuse existing function and begin an interoperability migration stage.
  - — Proof of concept or integration work of MQSI Agent for CICS with MQSeries Integrator, MQSeries Workflow, or other AIM eBusiness solution integration work.

5.0 How do I get there?

5.01 Overview

From this document, which has described the framework approach, MDp customers should now review their specific implementations, optionally with AIM or IGS Global services, with the local IBM account representative, and preferably including the representative responsible for AIM middleware. Through the course of our customer sessions, we understood that existing MDp customers will be interested in maintaining or protecting their investments in MDp with continued use and will require a smooth migration, where possible, to the new IBM AIM strategic middleware offerings.

In order to proceed with a migration approach, the owner of the MDp applications should begin to consider the following initial questions:

1. Does the function outlined herein for MQSI Agent for CICS, combined with the existing WebSphere Software Platform offerings, meet the functional need to migrate my current MDp application?
2. Is it planned to "interoperate", in other words, to have a mixed MDp/MQSI Agent for CICS implementation?
3. In which types of environment do I wish particular functions to operate? For example, an implementation of MQSeries Workflow on OS390 controlling a MQSI Agent for CICS run-time implementation might be suitable for some customers. Alternatively, some customers might want to implement a message broker, MQSeries Integrator, to perform existing or new function planned in the MDp environment. MQSeries Integrator could be used with the MQSI Agent for CICS. For help in answering this question, see "5.05 Current MDp deployment scenarios mapped to AIM products" on page 47.
4. Do I have the required skills within my own team to plan and implement the migration?
5. If it is viewed that MQSI Agent for CICS only partly replaces the required function, what specific functions are required to complement the adapter in order to replace the existing MDp implementation?

5.02 Customer commitment to continued use

While customers recognize MDp's perceived strengths and weakness, they are benefiting from their current implementations. Since customers have invested and implemented MDp solutions in their environments, it is necessary to understand customers' commitments and strategies in their usage of MDp. A task force was assembled to meet with a subset of customers who volunteered to review IBM's approach and provide input validation.

As a result of these meetings, we have a better understanding of how customers are using MDp and plan to use MDp in the future. We have also captured some additional requirements for a strategic approach to the development of middleware solutions.

Based on our feedback, we have defined these customers into three distinct categories with regard to their investment and usage of MDp:

I. Significant investment in MDp

These customers have invested significantly in MDp and will continue to invest in MDp for some time. These customers plan to continue development and investment in MDp for several years. They have typically deployed MDp as a central brokering hub for business integration.

These customers require interoperability and longer term migration needs.

II. Moderate investment in MDp

These customers have begun to implement MDp in multiple channels. However, they would be interested in and are capable of expanding usage through IBM's strategic application integration middleware products: MQSeries Integrator and/or MQSeries Workflow in conjunction with the MQSI Agent for CICS. These customers will continue to invest MDp over the next few years. However, they will employ the new product set as it becomes available for new development.

These customers will consider interoperability and short term migration needs.

III. Small investment in MDp

These customers have implemented MDp for defined or simple applications and would prefer to move to IBM’s new strategy for business integration now rather than investing in MDp further. These customers would be interested in migrating off of MDp as soon as the new product set is available.

These customers are interested in migration and would prefer to invest in new strategy now.

5.03 Existing applications

5.03.01 Preferred usage techniques

For future developments that require design and implementation of new MDp applications, the following guidelines have been identified which could ultimately ease the migration from MDp when the appropriate replacement becomes available. It should be understood that these are only guidelines and compliance does not guarantee migration or portability. They should however lessen the impact and even promote reuse where possible.

5.03.01.01 Preferred MDp Usage

These are some preferred usage techniques that, if put into effect, should ease migration to the MQSI Agent for CICS in the future:

- User exits and business logic should be isolated in separate modules or programs and executed via an EXEC CICS LINK w/COMMAREA.

  Note: The technique above, EXEC CICS LINK w/COMMAREA should be used to create a “wrapper” for customer specific business logic or processing. This creates a well defined data area for inputs and outputs, (one in/one out, one in/zero out) synchronous approach that should be capable of being modeled into a new flow. The activity behind the well defined interface is “opaque” hidden from the model.
- Use of MQSeries as the communication interface from/to MDp Client/custom client application and MDp Host.
- Use of the JAVA implementation of the MDp Client Fastpath API and related processing.
- Use of MQSeries, FEPI and CICS DPL mechanisms for back-end connectivity. Use of Request Profile or Workflow Object Definition for all WFO routing needs (Use of External or Static Routing.)
- Utilization of Data Dictionary for all record formats, storage requirements and application logic.
- Isolation of compensation spheres and implementation as separate distinct units of work from the originating unit of work.

The following are functions or usages of MDp that are NOT recommended and would make migration more complex:

- Customization of the IRM and ORM modules.
- Customization of workflow skeletons used to generate workflow objects.
- Usage of MDp header information for application use.
- Usage of MDp system areas (SCB/PCB/MCB) for application usage.
- Usage of Dynamic Routing.

The following MDp functions do not have identified functional equivalencies within the MQSI Agent for CICS:

- Queue Management
- MDp System Requests (Msg Type =04) related to MDp Queues and MDp Queue processing
- Store and Forward

The following functions do not have identified functional equivalencies within the MQSI Agent for CICS, however, could be developed as a custom stand-alone process through a custom interface or stand-alone program:

- Translation Table Processing
- LU6.2 back-end server agents.
- CICS ‘START’ back-end server agents

Note: If LU6.2 is required as a means backend interface then an appropriate wrapper using EXEC CICS LINK w/COMMAREA (a well defined interface) as described above should be used.

These guidelines are based on the current IBM MQSeries Business Integration tooling and product plans and are subject to change, as the IBM product portfolio evolves.

5.04 Potential stages of migration

There are several thoughts to consider when determining the best migration path from an MDp based solution to the new IBM AIM/MQSeries Business Integration solutions. Depending on customers' requirements, there are several possible scenarios that may be applied. Customers wishing to capitalize on the benefits of common tooling and the capabilities of the MQSI Agent for CICS have several options or stages to follow which will help them reach their goals.

The following stages have been reviewed with customers and identified as a potential phased migration path from MQSI Agent for CICS:

These guidelines are based on the current IBM MQSeries Business Integration tooling and product plans and are subject to change, as the IBM product portfolio evolves.

Coexistence

- Front-end uses MDp Client API for existing development.
- Front-end uses MQSI Agent for CICS API for new development.

Interoperability

- MDp Business Requests/Dictionaries can be surfaced in new tool as components.
- MQSI Agent for CICS can activate invoke MDp Business Requests.
  - — Will require the development/use of a bridge, importers, to complete.

Migration

- Migration of content out of MDp.
- Utilize component approach (EXEC CICS LINK w/COMMAREA) as means to recast MDp content.
- Possible usage of utilities to facility content migration.

Stage 1 - Coexistence

In this stage, customers are able to use the new tooling in conjunction with the MQSI Agent for CICS for new application development within their enterprise. This is the approach that needs to taken initially in order to provide proof of concept and familiarity with the modeling tools, importers and generators. This approach can also help customers understand the performance capabilities of the MQSI Agent for CICS runtime.

This stage introduces the notion of modeling/containing the information model (inclusive of the logical messages and flow compositions) in XML meta-data. Customers will begin to model their development using the new tools and generate to the TS 1.3 runtime for subsequent processing using the MQSI Agent for CICS.

The Front-end applications will interface to the new Java command API that will invoke the MQSI Agent for CICS. Existing MDp Business Request Processing will continue to be executed via the vehicle already implemented by the customer.

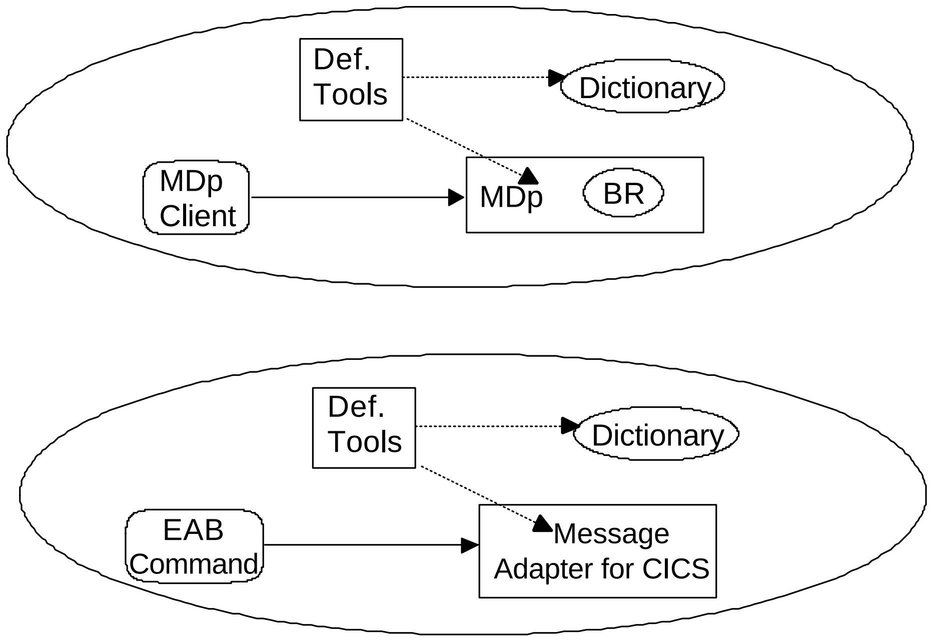


Figure 7. Coexistence stage of migration path

Stage 2 - Interoperability

In this stage, customers begin to develop new applications using the MQSI Agent for CICS, while also, reusing and interfacing to already developed MDp Business Requests. Under this scenario, customers can surface MDp Business Requests (at the business request boundary) as components with the tooling.

In order to accomplish this, a “bridge” will need to be developed that provides the MQSI Agent for CICS the ability to invoke an MDp business request, wait for the reply and continue through the microflow in the adapter. The bridge will be a second stage deliverable. See “5.06.02 Planning horizons/timing” on page 53. An importer utility for certain profile support from MDp inclusive of a COBOL importer step to get the current MDp request/reply chapters recognized in the information model.

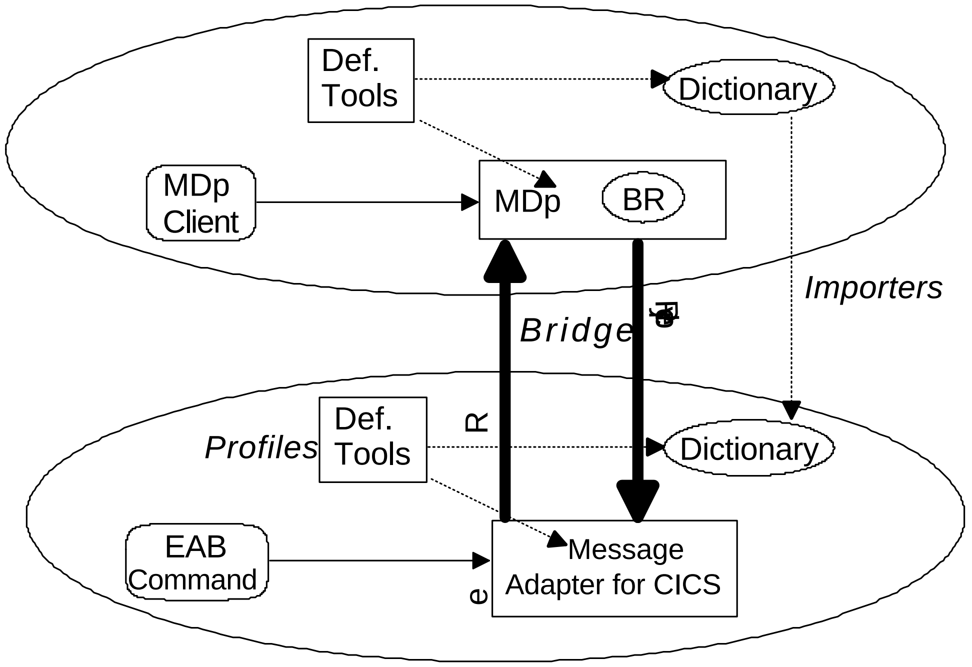


Figure 8. Interoperability stage of migration path

Stage 3 - Migration

This final stage of migration begins the evolution for customers to replace MDp functions with functions developed using the strategic IBM/AIM business integration products. (These products include the AIM strategic runtimes discussed, and the MQSI Agent for CICS.)

As customers enhance their existing business requests, customers should begin to migrate their business logic from MDp user exits, where possible, to the common modeling tooling and deploy to the MQSI Agent for CICS run-time (or other IBM AIM Runtimes based upon requirements) to accomplish their business integration objectives.

This stage requires in depth analysis of application requirements applied to the capabilities of the IBM/AIM runtimes and the MQSI Agent for CICS.

It is anticipated that, through the early validation and beta period experiences, we will have a better understanding of best-usage practices and deployment methodologies. This information will be useful in the field to ease customer migration activities.

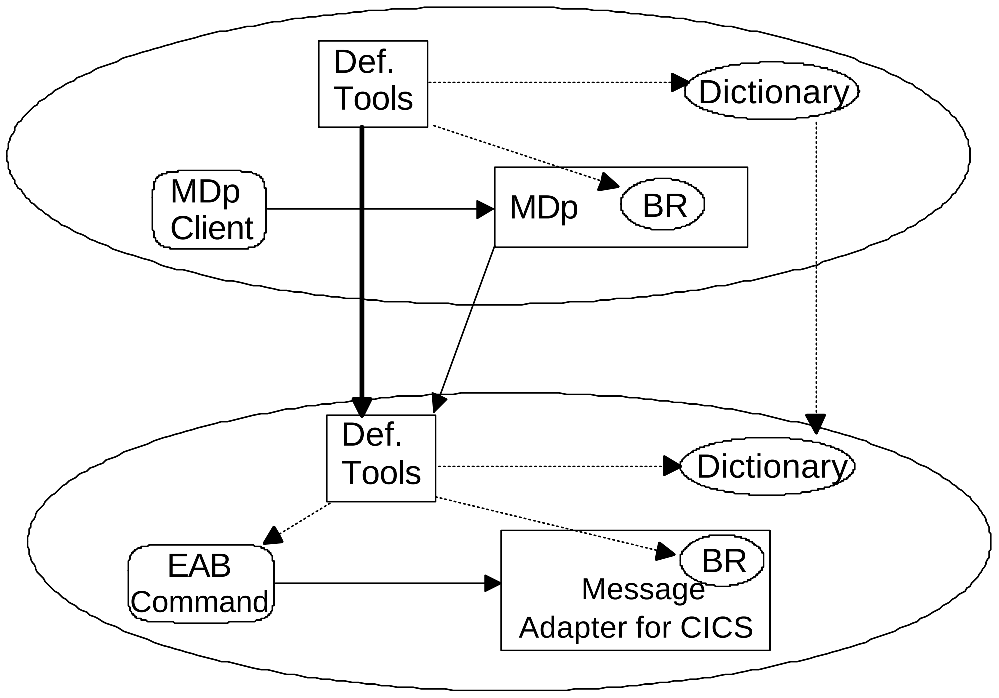


Figure 9. Migration stage of migration path

5.05 Current MDp deployment scenarios mapped to AIM products

MDp's middleware application is utilized across many different industries including banking, health care, transportation and utilities. MDp is used to provide business integration for call centers, Internet commerce and web enabled customer service (CRM) applications.

The following are some deployment scenarios that various MDp customers employ in their environments:

5.05.01 Session Based Processing

(**Single Controlling Application** or One to One)

This is the general scenario for customers deploying MDp in conjunction with CRM applications to access a specific legacy back-end application for their call centers. Under this scenario customers use MDp to assist in re-engineering their front-end processing scripts around the needs of a Call

Center (for example) and streamline their data access procedures by using MDp to access their legacy data.

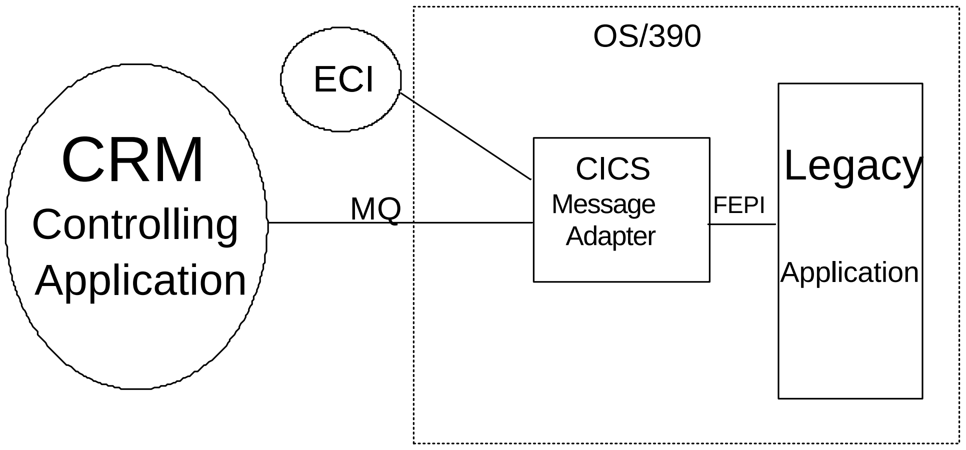


Figure 10. Single controlling application scenario

Under the above scenario depicted in the figure "Single controlling application scenario", customers could implement the MQSI Agent for CICS as a standalone process that interfaces to the controlling CRM application over MQSeries. Activation of the MQSI Agent for CICS via ECI/DPL is also possible.

The MQSI Agent for CICS would be responsible for the navigation of the 3270 emulation to the specific legacy back-end application.

5.05.02 Defined Application Multiple/Disparate Back-end Systems

(One to Many)

Customers requiring data integration between disparate back-end systems use MDp to combine the access in order to provide a single interface/presentation to the user. Customers employed this scenario when multiple legacy applications were required to perform a business process.

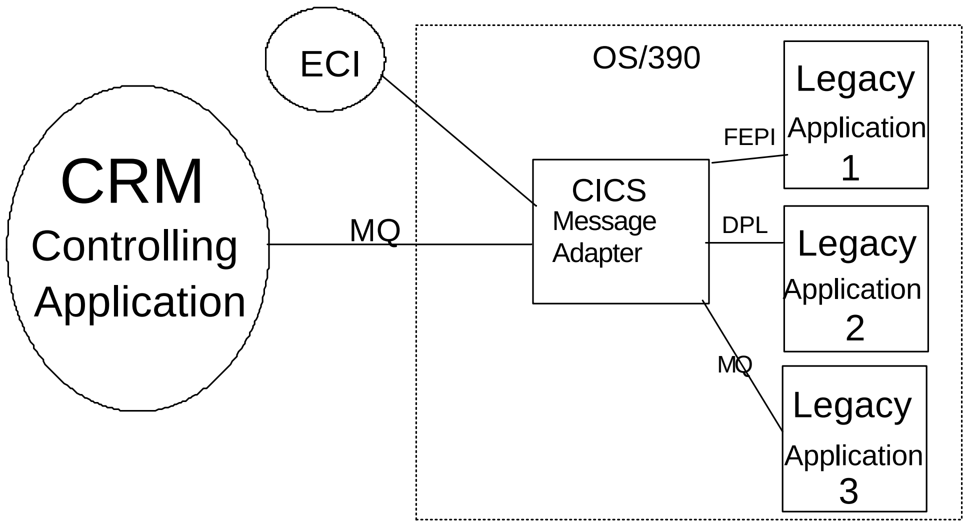


Figure 11. One to many scenario

The scenario depicted in the figure “One to many scenario” above could be deployed by customers requiring legacy access to multiple OS/390 or MQSeries enabled back-end legacy applications.

The MQSI Agent for CICS would be responsible for navigating to legacy application 1 using FEPI, to legacy application 2 over a DPL process, to legacy application 3 the MQSeries put/get, mapping all of the desired data to the reply format and coordinating and sending the message back to the CRM controlling application for continued process management.

Customers requiring additional mapping capabilities could implement the following solution depicted in the figure “One to many scenario with additional broker mapping” below. Under this scenario, additional back-end application access could be achieved using MQSeries Integrator over MQSeries and the MQSI Agent for CICS for OS/390 application access.

MQSI could be used to achieve additional data retrieve over MQSeries to an MQSeries enabled application.

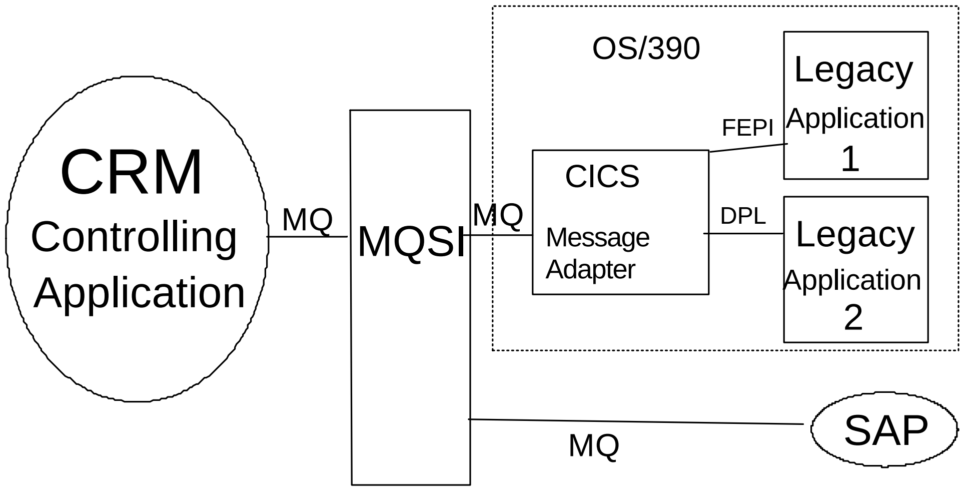


Figure 12. One to many scenario with additional broker mapping

5.05.03 Multiple Channel Implementations

(Many to One)

Customers capitalized on the investment in MDp business request development by reusing MDp business request development from additional front-end channels.

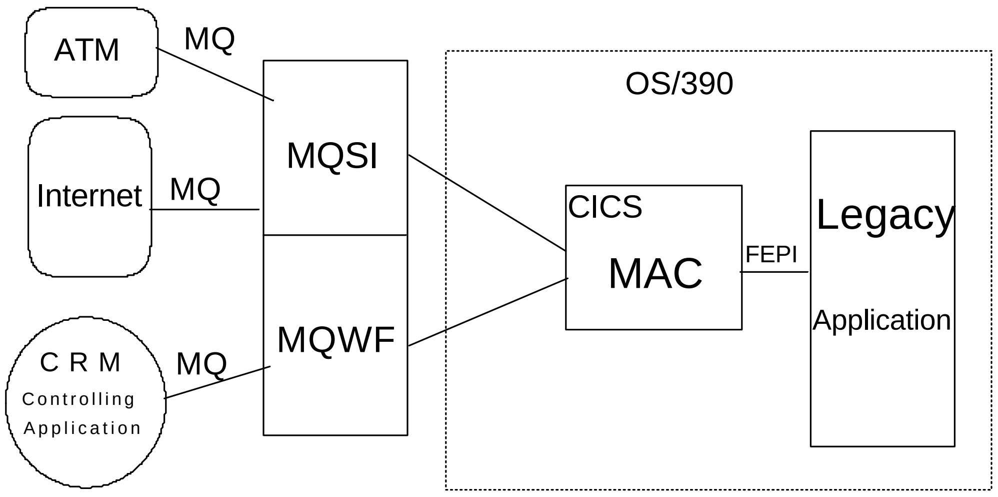


Figure 13. Many to one scenario

The scenario depicted in the figure “Many to one scenario” illustrates how customers could use MQSI as the broker to handle multiple disparate channels to the MQSI Agent for CICS (MAC) and another distributed application. For the CRM application MQSeries Workflow could be used to handle the longer duration business processes or queueing facilities.

The MQSI Agent for CICS would be used for the legacy back-end data access.

Multiple Channel Disparate Back-end Systems

(Many to Many)

Customers employed MDp as their enterprise middleware (business integration) hub. In this way they capitalized on the development and investment in MDp for business request reuse, from all channels, and scalability.

Both Session and Non-session based processing (Multi-Channel/CRM Goals)

Customers using MDp for enterprise integration/brokering hub would employ both session based and non-session based processing depending on the business channel or application that connected to MDp. This is deployed for customers using MDp for multiple channels as well as a Call Center to provide consistent response for a centralized CRM approach.

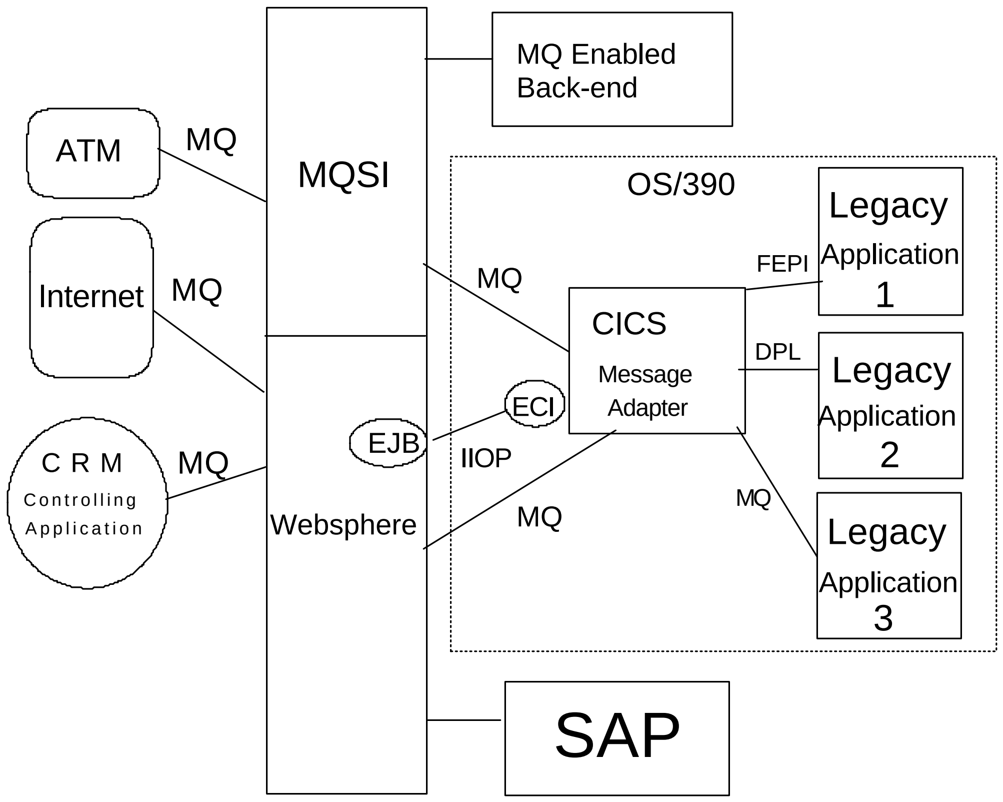


Figure 14. Many to many scenario

Customers requiring a message broker or hub implementation might deploy the architecture described in the figure “Many to many scenario” above. Depending on the requirements of each business channel, customers could implement WebSphere and MQSI to achieve different business channel requirements and connect to the MQSI Agent for CICS for the OS/390 legacy back-end data requirements.

5.06 Future goals

In this section we will attempt to express the kinds of validation we would expect from early customer experiences throughout the beta period of MQSI Agent for CICS and impressions derived from this writing and past visits to customer sites.

5.06.01 Functional areas to be validated by users

We anticipate user feedback in the following areas during development and though the beta period:

- Tooling Support
- Importer function usability
  - — Logical message view
  - — Desired wire formats, standards
- Generator framework usability
- Definition and capture of the business flow
  - — Conditional mapping, general mapping/usability
  - — Usability of expressing simple business rules in the flow
- Runtime efficiencies/usability
  - — Configuration and administration support
  - — Performance considerations
  - — Levels of recoverability
  - — RAS issues, tracing, logging, security
- Migration and training feedback

5.06.02 Planning horizons/timing

The following are tentative milestones for future planning of work efforts related to MDp function migration.

These timelines are based on the current IBM MQSeries Business Integration tooling and product plans and are subject to change, as the IBM product portfolio evolves.

Near term activity (3Q00 - 1Q01)

This is primarily a development period for the initial stages of the MQSI Agent for CICS and a validation period for the early stages of the common tooling. This period will be the beginning of a beta period for the MQSI Agent for CICS. This is also anticipated to be a customer feedback period from customer visits and this document.

Tentative milestones in this period are the following:

- Customer continued use of MDp
  - — Planning implementing preferred techniques

- Development of the MQSI Agent for CICS
- Tooling development and refinement
- Beta period target begin 10/00
- Feedback from beta period, customer planning
- Planning for potential services offering
- Migration component development
- Stage 2 deliverable planning

Mid term migration activity (1Q01 - 4Q01)

- 1Q00 - Generally Availability MQSI Agent for CICS Stand alone packaging on Transaction Server 1.3.
- Migration software development, stage 2 development.
  - — MDp/MQSeries Integrator Adapter Bridge
  - — Importers/some utilities
- Customer evaluation period and migration testing. See “5.03.01 Preferred usage techniques” on page 42.
- Potential planning/deployment MQSI Agent for CICS with MDp of interoperability strategy for new applications.
- Beta period migration utilities/bridges.
- Potential service offerings for analysis current MDp" Business Requests

Longer term activity (2002 - 2004)

- Next generation of strategic AIM products/offerings, (MQSeries Integrator, MQSeries Workflow, WebSphere) begin to appear with new functionality.
- MQSI Agent for CICS is integrated with AIM Business Integration direction (WebSphere, MQSeries, TS Series)
- Customers begin to 'Migrate/Reuse' existing MDp Content

6.0. Summary

IBM understands the investment that the current MDp customers have made and, where possible, IBM is considering tools for migration/interoperability to get to newer technology.

The functionality that customers find valuable in MDp today is being generalized as requirements for the strategic IBM/AIM middleware offerings. The MQSI Agent for CICS is an initial piece of this functionality.

During our customer meetings, many agreed that the current proposed functionality of the MQSI Agent for CICS met most of their requirements. Some customers saw the need for advanced control flow functionality used in MDp in future IBM AIM product strategy but could wait for a more generalized and portable deployment as future requirements.

We have defined a tentative plan for definable stages of migration to IBM's strategic middleware products, including connectors and adapters. It is our intent to continue to work with customers from migration planning, realigning MDp functions, through implementation.

IBM is committed to MDp customers and would like to continue to provide them with strategic AIM/business integration products and services. This document has been released to you to assist in your migration analysis and planning, and to enable IBM to obtain direct feedback concerning your plans.

In addition to sharing your plans with your local IBM representative, please copy initial comments to David Bonaccorsi at `davebono@us.ibm.com` or Whitney Sande at `sande@us.ibm.com`.

Appendix A. IBM MQSeries family of products

Let's look at MQSeries base messaging, MQSeries Integrator and MQSeries Workflow. For more information see http://www.ibm.com/software/ts/mqseries

MQSeries base messaging

With MQSeries, you receive a family of four APIs designed to make programming straightforward for any messaging task, from the simple to the most advanced. All four of the APIs can interoperate with each other.

Three APIs can be used for exchanging messages:

1. **MQI** is the API that provides full access to the underlying messaging implementation, available for all key languages and environments.

2. **JMS** or Java Message Service, is the Java standard providing much of the function available through the MQI. Java Message Service (JMS) is a specification of a portable API for asynchronous messaging. JMS has been developed by Sun Microsystems in collaboration with IBM and other vendors interested in promoting industry wide standard frameworks.

3. ***AMI*, ** or Application Messaging Interface, simplifies the handling of messages with a higher level of abstraction that allows policy handling and many messaging features to be provided by the middleware. The new Application Messaging Interface (AMI) is a high level API that greatly simplifies programming for application messaging and publish/subscribe.

   IBM specified and developed the API, which has now been adopted as a standard by the Open Application Group Inc. (OAGI). The AMI, with bindings for standard programming languages including Java, C, and C++, reduces the amount of code required to be written for new applications. It provides a high level of abstraction, moving message-handling logic from the application into the middleware. Previously, programmers coding messaging applications needed to select queue names and decide message characteristics such as priority, retries, or expiry time. Now, they can register sets of standard characteristics as policies for message-handling, and use the AMI to set the appropriate policy for a message.

IBM will provide a suite of common policies, and an open policy-handler framework that encourages additional policies to be created by the enterprise or third-party software vendors.

In addition to policy, the AMI also allows programmers to associate a service name with a message. The service is a high level of abstraction that represents an MQSeries queue, but can also be implemented to communicate with a database, a printer, or e-mail, for example. An important benefit of the AMI is that programmers can focus on business logic and message content. Connectivity code is reduced to specifying a service and a policy to be used when sending or receiving messages.

4. The fourth API, the ***CMI,*** or Common Messaging Interface, simplifies the creation of message content. CMI is a logical message construction API, used in conjunction with a message delivery API, such as the MQI, the AMI or MQSeries Support for JMS.

   With CMI, programmers will be able to dynamically construct and parse messages, and provide query and update facilities on constructed messages regardless of their physical representation. CMI will provide programming support for both language dependent and language independent data structures in a consistent manner. It will handle tagged value data such as XML, and language dependent structures found in C and Java, used in conjunction with the message dictionary support provided in MQSeries Integrator V2. CMI will provide a powerful tool for manipulating both simple and complex data as a single entity.

MQSeries Integrator

MQSeries Integrator Version 2.0 is compatible with, and a major development of, Version 1 functionality. It provides graphical tools for constructing how critical data or business events are handled, by visually connecting a sequence of processing function to dynamically manipulate and route messages, combine them with data from corporate databases, warehouse in-flight message data for auditing or subsequent analysis, and distribute information efficiently to business applications.

MQSeries Integrator has evolved into an open framework that can combine built-in processing components with those from third-party software vendors or the enterprise.

Message formats can be defined through a message dictionary, either the one supplied with the product, the MQSeries Integrator Version 1.x compatible dictionary, or a third-party dictionary.

MQSeries Integrator Version 2.0 has a publish/subscribe service, which is compatible with the MQSeries base publish/subscribe function, and includes:

- Routing a message to interested subscribers on both the message topic and content.
- Authorization based on multiple levels of the topic name.
- Support for more flexible topologies of publish/subscribe brokers.

MQSeries Workflow

IBM MQSeries Workflow helps you run your business processes. MQSeries Workflow is used to design, document, execute, control, improve and optimize the business processes, so you can focus on the company's key objectives. After modeling your processes with the MQSeries Workflow Buildtime, the MQSeries Workflow Runtime runs them by navigating through the workflow models. Applications are invoked automatically, and work items are created and distributed to the worklists of people involved.

MQSeries Workflow provides the following benefits and capabilities:

- Core component of enterprise architecture by integrating business processes even across enterprises acting as a workflow broker.
- Integrates existing and new applications to power eBusiness solutions, ranging from CICS and IMS transactions to Web applications.
- Fast and flexible execution of business-to-business and business-to-consumer processes to address fast changing markets.
- Using XML, applications can trigger MQSeries Workflow process.
- Using XML, MQSeries Workflow can start any application.
- Higher productivity through business process automation.
- Better service to customers at reduced costs, because of reduced cycle times.
- State-of-the art visual workflow modeling.
- Manages fully-automated application-to-application workflow, called communication workflow.
- Manages workflow processes including applications that require human intervention.

Appendix B. IBM's Common Connector Framework

Experience has taught us that a framework architecture is one of the best ways to deliver common functionality while allowing extensibility. It is important that adapters and connectors be deployed using the same principals to ensure consistent architecture and access to common services in the framework. IBM's current answer to this problem is the Common Connector Framework (CCF). As stated previously, this work is currently evolving to the J2EE Connector Architecture Specification and CCA. For more information, see http://java.sun.com/j2ee/connector. In this context, the CCF described in this section is intended to be used as an example of a deployment of a common architecture framework for connectors.

The CCF and associated connectivity framework model is the framework and programming model currently shipped with VisualAge Java and WebSphere Enterprise (Component Broker). The CCF provides a unified connectivity programming model by:

- Enabling the users of the connection to have a normalized view of different underlying protocols and infrastructures.
- Allowing the underlying connection services a standard mechanism to define the execution environment.

Each connector implementation may support a differing levels of services, known as “quality of service” (QOS). At deployment and runtime, the appropriate QOS can be selected for the application.

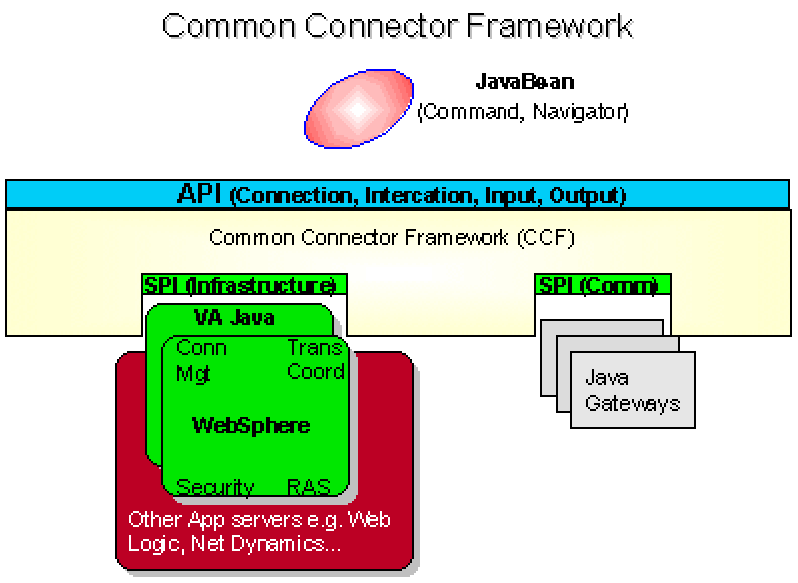

Common Connector Framework
JavaBean
(Command, Navigator)
API (Connection, Intercation, Input, Output)
Common Connector Framework (CCF)
SPI (Infrastructure)
SPI (Comm)
VA Java
Conn Mgt
Trans Coord
WebSphere
Security
RAS
Other App servers e.g. Web Logic, Net Dynamics...
Java Gateways

Figure 15. The Common Connector Framework model

By using a common framework, other benefits can be achieved:

- Common architecture.
- Cross platform consistency.
- Facilitates vendor buy-in and exploitation, which for connectivity means it enriches the range of capabilities by allowing uniform access by systems which include Enterprise Resource Planning (ERP), Customer Relationship Management (CRM) and Supply Chain Management (SCM).
- Consolidation of existing connector solutions.
- Provides a migration path by shielding applications from underlying technologies, i.e. specifies a uniform view of underlying protocols and infrastructures.
- Allows a connector to be reused in many environments, i.e. uniform access by systems as mentioned previously.
- Facilitates the construction and testing of applications that are built to standard interface.
- Provides enablement of an overall IBM architecture that includes:

— Common tooling for development, deployment and maintenance and diagnostics.

— Uniform administration, services (e.g. directory, naming, security) and monitoring model.

Additionally, a connector framework also relies on both data transformation services and activity definition facilities which can be used in a common fashion when/if required by the application. Therefore it is important to understand that the overall connector framework needs to support all sorts of environments across wide-range of applications and in varying degrees of complexity, including:

1. Simple application development on existing platforms (i.e., without the need to introduce additional technologies such as components, objects or Java). Such applications require either a data mapping and/or connectivity to host-base applications.
2. Application built using distributed object or component technology that wish to instantiate data from existing application with new data objects or components. Additionally, the framework should be selectable at a level appropriate to the application program.
3. Application that require more complex process modeling which require additional capabilities, such as activities and navigation.

Connectors and adapters available today

As stated previously, there are many groups within IBM already producing and shipping useful connectors that cover all major AIM application servers and middleware.

These connectors solutions fall in to the following major product categories:

- IBM e-Business connectors
- WebSphere Enterprise connectors
- Domino Enterprise Integration Tools
- Common Connector Framework deployments

A common goal for many if not all of these group is to comply with IBM’s Common Connector Delivery efforts with future goals to conform to CCA/CCF Framework standards with common tooling/services as described above.

The following figure, “Evolution from today to the Common Connector Architecture”, indicates the evolution path for today’s connectors.

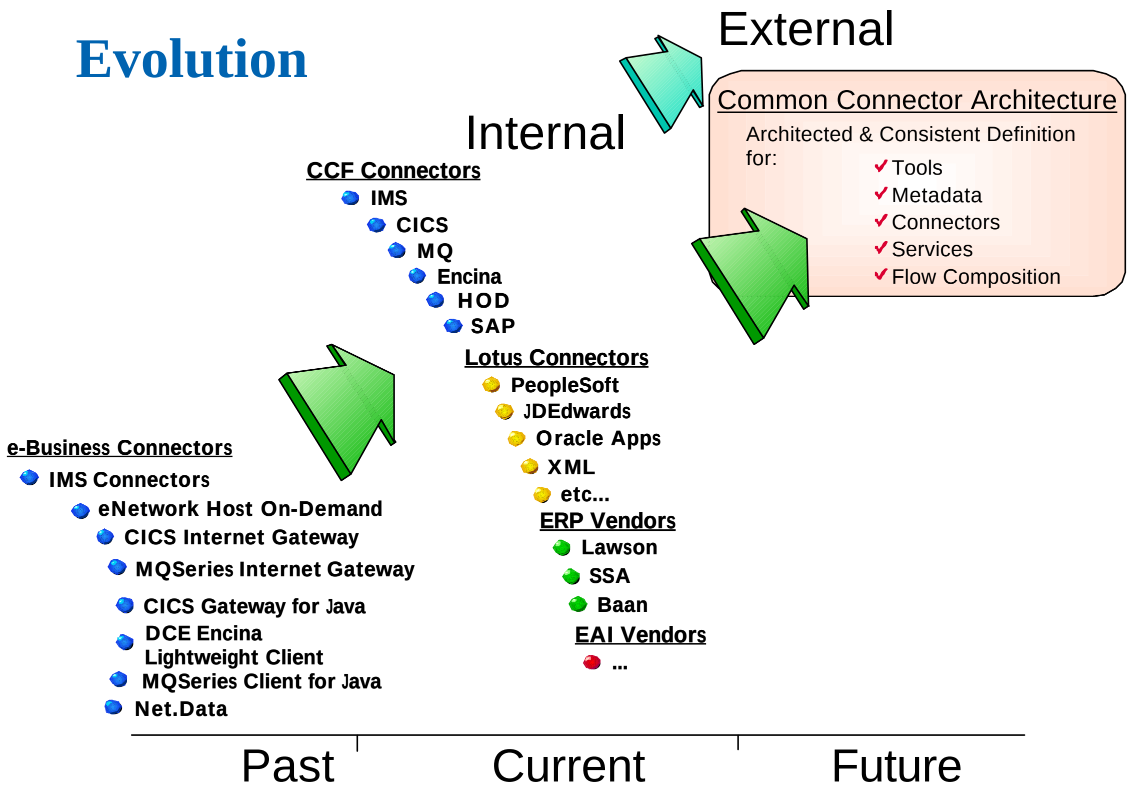


Figure 16. Evolution from today to the Common Connector Architecture

Appendix C. Flow composition model

This section will briefly describe the flow composition model (FCM) and related tooling.

Essentially the architectural components of the flow component model are:

- The logical information structure defining the explicit information exchanged between components in the form of message types, interfaces and parameters.
- Flow components which represent the usage of business services in the context of the flow model. A flow component type provides a template for flow components and describes its interfaces.
- Formal specifications of flow models that define the interaction between flow components in support of higher level business functions.

The figure illustrates the architectural components of the FCM.

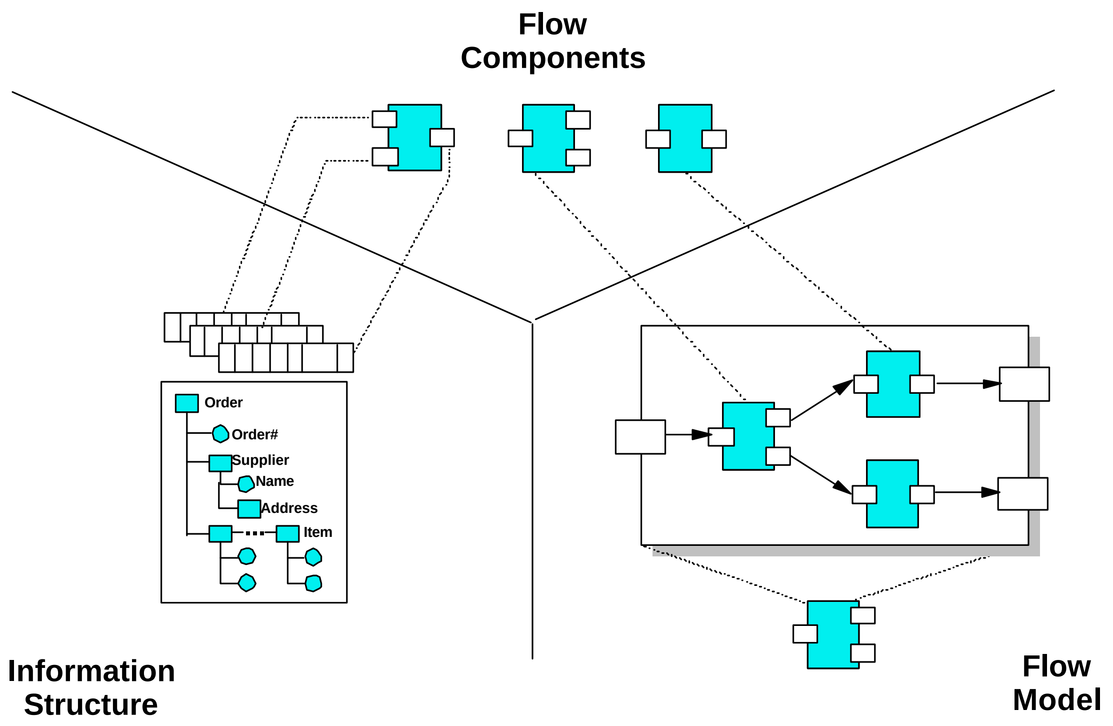


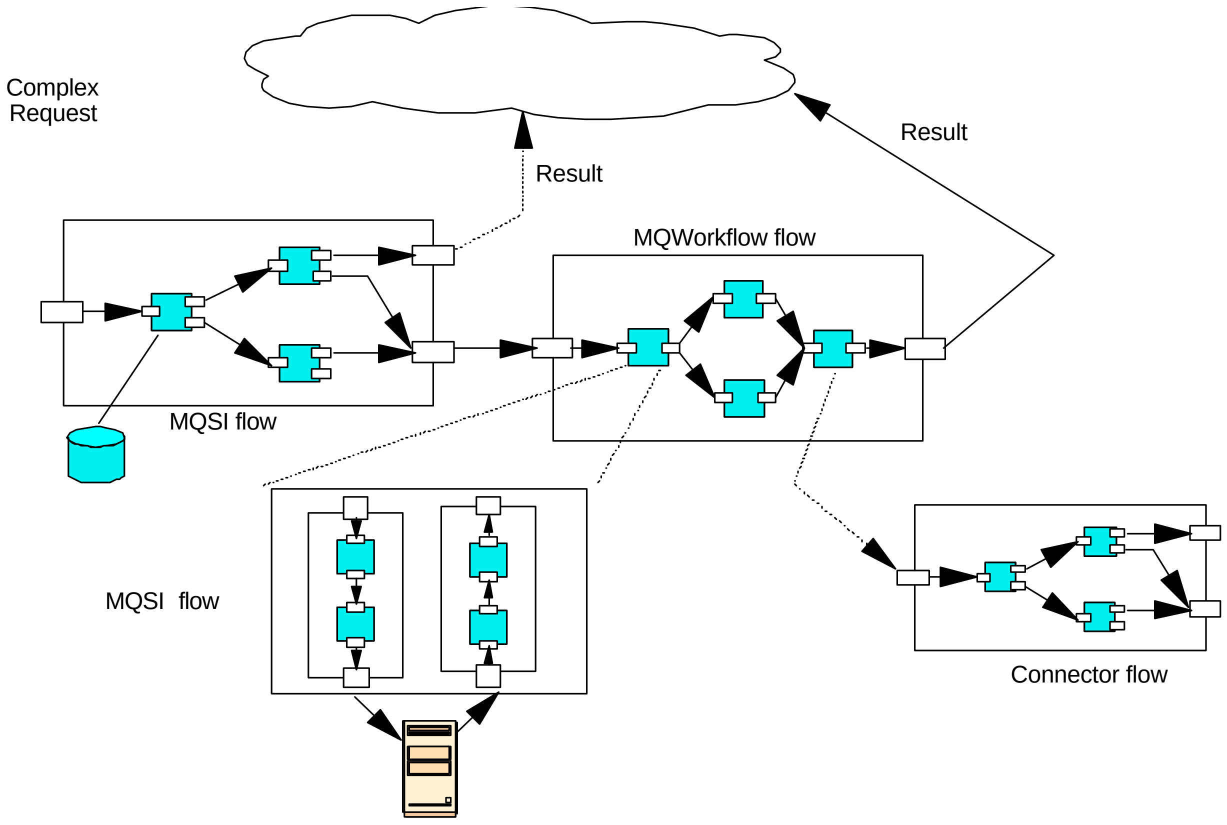
Figure 17. Architectural elements in flow composition

The FCM also supports the development of flows at various levels of granularity. At the top level can be more course grained flows termed macro-flows that may be the result of business process analysis. In a banking scenario, the course grained flow may represent a workflow process such as “open account” which invokes a subflow or micro-flow in a connector that “identifies the customer”.

The FCM supports the modeling and annotation of complex flows with deployment parameters that enable the deployment of the flow model into different runtime environments. A complex flow may be made up of sub flows that are each deployed in separate runtime environments. In this way business integration can be accomplished between domains involving various domains of IBM’s strategic middleware (WebSphere, MQSeries Workflow, MQSeries Integrator and adapters).

The figure illustrates mixing multiple levels of flow.

Figure 18. Mixing levels of flow



The FCM is supported by a Flow Composition Modeller and an integrated set of tooling that supports the definition of flow component types and composition of these components into flow models. The FCM will be offered as part of a strategic integrated application development environment.

The FCM flow modeller supports graphical composition of flow components from a palette of flow component types. Flow components can be connected by control and data connections and business rules for interaction coordination and information mapping can be defined in the flow graph. In addition, the tool supports static checks of flow model for correctness and simulation of flow model behavior.

The FCM modeller uses information captured in the Message Type Dictionary to define flow component interfaces and to support specification of mapping rules between flow component interfaces.

The figure illustrates integrated tooling for business integration.

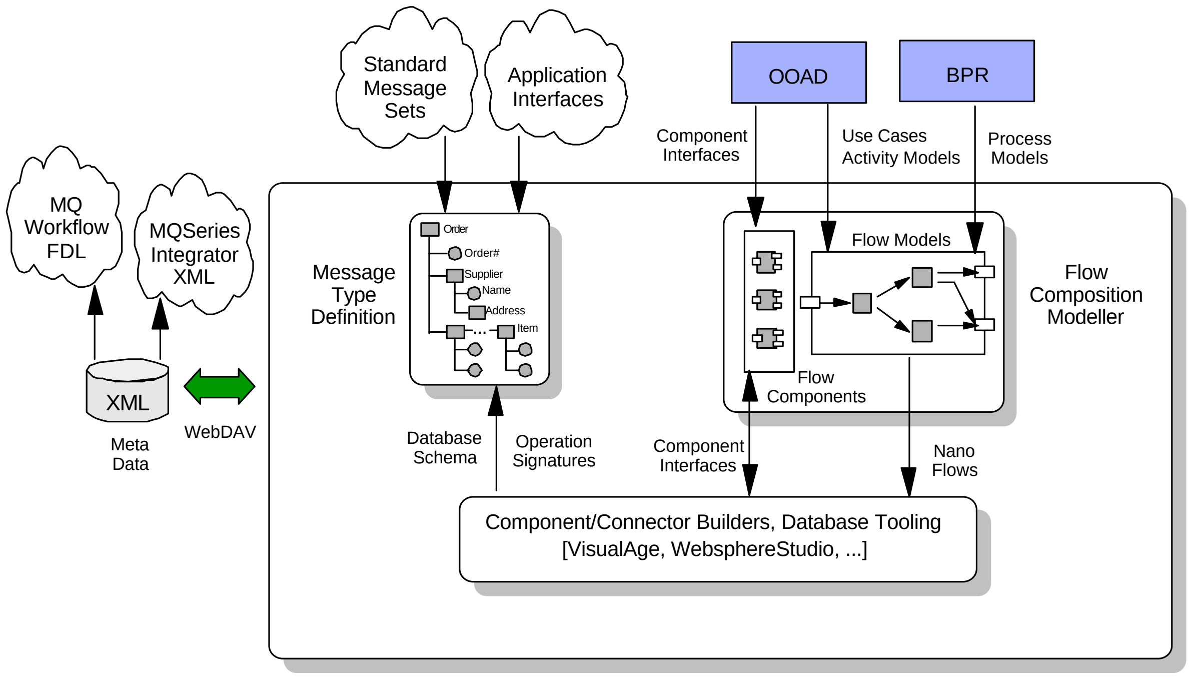


Figure 19. Integrated tooling for business integration

Previous sections of this paper have defined the importance of a common model and tooling that allow the developer to:

- Define business flows.
- Indicate usage of the respective logical message model in the flow.
- Describe business rules indicating the component interaction required for the business process.

The flow composition model and the respective modeling tools described above play an important role in the follow on development efforts for MDp functionality as seen in the next section describing the Message Adapter for CICS and follow on adapter/AIM runtime offerings.

Appendix D. IBM MQSeries Adapter Offering

IBM MQSeries Adapter Offering works with MQSeries messaging to enable you to reduce the risk, complexity and cost of managing the point-to-point integration of your business processes.

In point-to-point integration, each application interfaces with all the other applications, individually. Each interface is different and there are many different interfaces. One change in one application typically requires changes to many interfaces. As the number of applications increases, point-to-point integration becomes rapidly less cost-effective. Adding each new application typically requires more work than adding the most recent-added application.

With MQSeries Adapter Offering, you can evolve from using point-to-point integration to one-to-any integration.

- All applications can use one common interface.
- Data from a source application, in the form of a message, is routed to one or more target applications.
- A change in one application typically affects only that one interface.
- Using a common interface that is application-neutral, such as an industry standard, can be even more cost-effective. More applications can be supported with less effort.
- As the number of applications increase, one-to-any integration becomes even more cost-effective. Adding each new application typically does not require significant changes to the interfaces to all the other applications.
- The MQSeries Adapter Offering can be deployed without changing applications or business processes at all. Typically, all the integration work is performed in MQSeries Adapter Offering.
- Integration work can be automated and can be based on templates. MQSeries Adapter Kernel can optionally be deployed with MQSeries Integrator to perform brokering and message transformation.

In MQSeries Adapter Offering, the interface to or from one application is provided by an adapter. All applications need an adapter to provide the interface between the application environment and the MQSeries environment. Each adapter is specific to an application.

Several examples of adapters are:

- Add a sales order.
- Synchronize a customer record.

- Synchronize an inventory record.
- Synchronize an item.
- Synchronize a sales order.

Build time and run time

The MQSeries Adapter Offering consists of two environments:

MQSeries Adapter Builder

Via an intuitive visual interface, the MQSeries Adapter Builder enables you to build an adapter for virtually any application. The user interface is similar to MQSeries Integrator's user interface. For more information, visit the MQSeries website at http://www.ibm.com/software/ts/mqseries/.

run time: adapter

The output of the MQSeries Adapter Builder. An adapter provides the interface to an application or from an application. Typically, you build each adapter to be specific to one message type that is sent from or to an application. Thus, the adapters themselves are not part of MQSeries Adapter Offering.

An adapter consists of C source code that compiles to a shared library. When the adapters and the MQSeries Adapter Kernel execute together at run time, they perform the run time functionality of the MQSeries Adapter Offering.

Depending on how you modeled it in the MQSeries Adapter Builder, the adapter can contain a wide variety of functionalities such as controlflow, dataflow, sequential navigation, conditional branching including decision and iteration, data typing, storing data context, transformation of data elements, logical operations and custom code.

You may reuse adapters that you have created.

There are two types of adapters:

- Source adapters, for the application that sends the data.
- Target adapters, for the application that receives the data.

An adapter is required for each message type. To send one type of message from one application to a second application typically requires one source adapter and one target adapter. If the second application must send one type of message to the first application, another source adapter and another target adapter are required. Thus, in this case, in order to

- Send one type of message from the first application to the second application and
- Then to send another type of message from the second application back to the first application,

four adapters are typically deployed.

You might be able to reuse some adapters for the two applications.

run time: MQSeries Adapter Kernel

A set of APIs and several executable programs, in C and Java, and several configuration files. The kernel enables the deployment and execution of the adapters. In addition to directly supporting adapters, the kernel performs related functions, among the most important: simple routing of messages and infrastructure services such as message construction, tracing and interfacing with MQSeries.

The kernel is installed on each computer on which a source adapter or a target adapter run.

With MQSeries Adapter Offering, business processes and each application can remain isolated from the specifics of middleware, message details and other applications. A common interface for messaging enables adding new applications without changing existing applications or business processes.

MQSeries Adapter Offering can reduce the need to write custom code. MQSeries Adapter kernel can be deployed in two tiers. One tier is the source side of the run time; the other tier is the target side of the run time. Two tier deployment can provide more efficient operation and less administrative overhead. A third tier for routing and delivery is not required to reside between the two sides of the run time.

As an option, MQSeries Integrator can be added to perform brokering, such as complex routing, data transformation and data mediation. It would add a third tier.

MQSeries Adapter Offering can be complemented by service offerings from IBM and others.

This page intentionally left blank.

Notices

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

Trademarks

Bibliography

J2EE Connector Architecture Specification.
See http://java.sun.com/j2ee/connector.

Message Adapter for CICS Functional Specification. 2nd ed. Middletown, RI, USA: IBM Corporation, 2000.

MQSeries Application Programming Guide. 9th ed. SC33-0807-09. IBM Corporation.