IBM MQSeries Workflow

Concepts and Architecture

Version 3.3

GH12-6285-03

IBM MQSeries Workflow

Concepts and Architecture

Version 3.3

GH12-6285-03

Note!

Before using this information and the product it supports, be sure to read the general information under "Appendix. Notices" on page 47.

Fourth Edition (March 2001)

This edition applies to version 3, release 3 of IBM MQSeries Workflow (product number 5697-FM3) and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces GH12-6285-02.

Contents

About this book

This book introduces you to IBM MQSeries(R) Workflow, hereafter referred to as MQ Workflow. It describes how you can automate, manage, and control your business processes. The first part of the book explains the *concepts* that are relevant to workflow management. The second part describes the *architecture* of an MQ Workflow system.

Note: References to z/OS also apply to the OS/390(R) version of the IBM operating system.

Who should read this book

Decision makers
Who want to improve the way their company operates.

Business planners and analysts
Who want to evaluate the benefits of MQ Workflow.

System administrators
Who want to get an overview of the MQ Workflow architecture.

How to get additional information

Visit the MQSeries Workflow home page at
http://www.ibm.com/software/ts/mqseries/workflow

For a list of additional publications, refer to "MQ Workflow publications" on page 57.

How to send your comments

Your feedback is important in helping to provide the most accurate and high-quality information. If you have any comments about this book or any other MQSeries Workflow documentation, choose one of the following methods:

- Send your comments by e-mail to: swsdid@de.ibm.com

  Be sure to include the name of the book, the part number of the book, the version of MQSeries Workflow, and, if applicable, the specific location of the text you are commenting on (for example, a page number or table number).
- Fill out one of the forms at the back of this book and return it by mail, by fax, or by giving it to an IBM representative.

Summary of changes

This section lists the major revisions to this book for the current edition.

Changes for MQ Workflow Version 3.3

Changes in the following chapters include both editorial changes as well as technical updates.

- "Chapter 4. Running your business processes" on page 19 introduces the Web Client. The Client for Lotus Notes is no longer part of the product, except for the API support, which is still available.
- "Chapter 5. Architecture overview" on page 25 describes the extension of the system architecture to incorporate tier zero for thin clients (Web clients).
  - The section "The client components" on page 32 describes the new thin clients, such as the MQ Workflow Web Client and the custom Java program.
  - "Communication support" on page 35 describes the new client connection type, using the MQSeries Server API for clients with queue managers to support a client concentrator setup.
  - "Workload management" on page 39 adds information about MQ cluster support.
  - "API support" on page 41 contains a new figure showing extended API support.

Part 1. The concepts of workflow management

Chapter 1. What is workflow management?

If you look at the fundamental concepts of workflow, they seem to be familiar: work is started for different reasons, often from a customer request, and goes through many stages towards completion, until the request is satisfied.

However, if you look more closely at the scenario, it soon becomes more complex. To handle a request involves many different people and multiple activities, which can even be processed automatically. Processing a customer request also requires various procedures and diverse information sources. Consequently, the result is very often an ill-managed process. Another common problem is that in organizations no one is responsible for the entire process and no one knows the exact status of activities in the process.

This chapter describes the concepts and benefits of a workflow management system. It gives you an overview of IBM MQ Workflow.

Figure 1 shows a workflow scenario with various tasks and people involved in managing a business process.

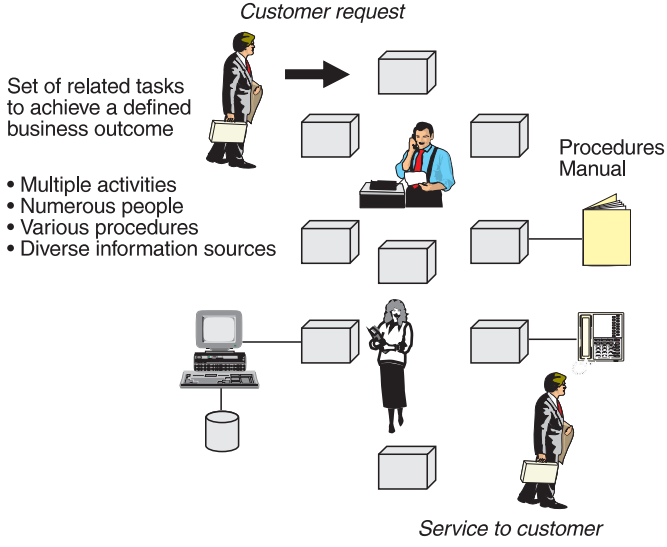

Figure 1. Managing business processes

To manage workflow efficiently, you must combine the activities of a *process* and its logic, the *organization* of all the people involved, and the *infrastructure*

of the resources that are needed, that is, computers and programs. If you combine the three views of process (logic), organization, and infrastructure, you can think of the three views of workflow management as navigating through a three–dimensional space as shown in Figure 2.

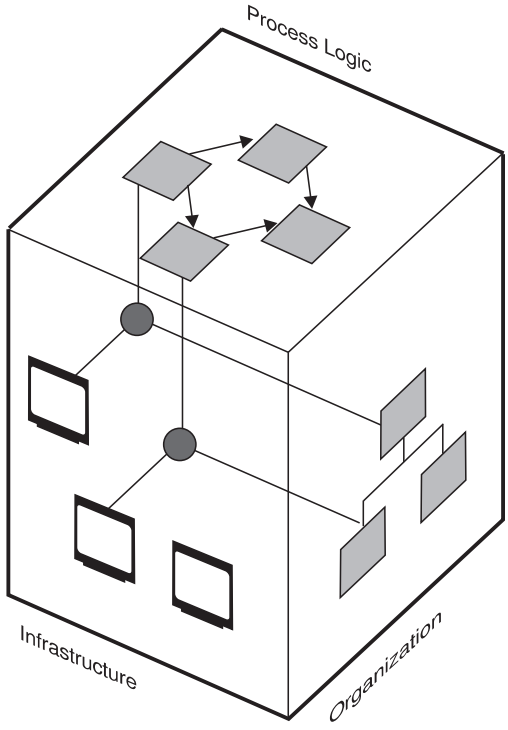

Figure 2. Managing the dimensions of workflow

If you want to manage these dimensions of workflow, you must define:

- The processes and their logic for your workflow model
- Your organization involved in the workflow
- The IT resources that make up your infrastructure

A *process* can consist of just one activity or typically of many activities, and even subprocesses containing more activities. For the various activities in a process you specify the control flow and the data flow. In addition, you add the application programs that you want to use within a process.

You also add your *organization*, defining roles that belong to certain members of your staff as well as levels of authorization.

As a third dimension, you specify your *infrastructure* with all the IT resources that you need.

Benefits of workflow management

Reengineering business processes and workflow management are important topics on the agenda of many companies. Triggered by a constantly changing business environment, companies must react faster. They must also be highly flexible in running their daily business. Business processes are no longer simply intraenterprise processes. Multiple enterprises are connecting their tasks together in interenterprise processes. This helps them to manage their global processes more efficiently. For example, the order activity in a production planning process for a car company starts the appropriate order entry process at a parts supplier. Organizing your enterprise according to the processes that need to be performed is the key to lean management. It is also a prerequisite for interenterprise business processes.

Fast and flexible execution

One of the key objectives of reengineered business processes is to minimize the time required for execution. Therefore, in a well-defined business process:

- Unnecessary tasks have been eliminated.
- Tasks are performed in parallel and even automatically.
- Tasks can be performed by different people.

Even different IT resources with diverse software can be used to perform the tasks. Often, these business processes run in a distributed and heterogeneous environment.

Workflow-based applications

Integrating business applications into a workflow management system means that you remove the flow dependency from the application. The routing features of a workflow management system allow you to extract all information that is related to the process flow from an application program. Equally, process-relevant data is under control of the workflow system. MQ Workflow is “middleware” and, therefore, similar to a database management system that allows you to extract standard data-management functions from an application program.

Whenever changes to the process flow need to be done, the applications that are part of the process model need not be changed. This also means that you can reuse your software components in other processes. Consequently, you can achieve significant cost savings.

For more information about the benefits of workflow-based applications, see *Frank Leymann, Dieter Roller, "Workflow-based Applications", IBM Systems Journal 36, no. 1 (1997): 102–123*.

Chapter 2. Managing business processes with MQ Workflow

With MQ Workflow you can design, refine, document, and control your business processes. Your company can focus on the work at hand while MQ Workflow manages your processes. The benefits are:

- Faster execution of processes
- Higher productivity through automation
- Better service to customers at reduced costs
- Improved quality of process execution
- Processes are enforced to achieve ISO 9000 compliance

MQ Workflow assists you in daily business operations, in planning and management, and also in the design of applications tailored to your business. You can do the following:

- Define and document your processes
- Run your processes to:
  - Support the people doing the work
  - Fully automate activities that do not require human guidance
- Administer your workflow

MQ Workflow is a client/server system and there are dedicated client and server components that are responsible for the different workflow management tasks.

Defining and documenting your processes

You can create a graphical representation of your processes with MQ Workflow. With it, you depict your business activities, add the staff that performs them and the programs and network infrastructure that support the people. You also define the flow of control and information between the activities. All of this modeling information is stored in the relational database of MQ Workflow.

The component that is responsible for these tasks is *Buildtime*. For more information about defining your processes in Buildtime, see "Chapter 3. Building a workflow model" on page 11.

Alternatively, if a process definition is available in text format and is written in MQ Workflow Definition Language (FDL), you can import the FDL file

using Buildtime. If you use a business process modeling tool that offers FDL as an exchange format for process definitions, you can directly import these definitions into MQ Workflow.

Running your processes

When you are satisfied with your workflow model, you translate it to run your business processes. For every instance of the process, the *server* components of MQ Workflow navigate through the process and move the work to the right person in the right sequence. MQ Workflow starts the programs, keeps process execution history, and provides recovery and restart procedures.

Activities that need to be performed appear in worklists of the *MQ Workflow Client* or the browser-based *Web Client* of your assigned staff members. When a staff member selects, for example, a program activity, the program is started with the necessary information. Users' worklists contain continuously updated overviews of their pending activities.

As you implement workflow management, people can use worklists as their primary user interface to other applications. They can then access applications and data on different platforms and user interfaces, such as Lotus Notes. For more information about running your processes, see "Chapter 4. Running your business processes" on page 19.

Instead of using worklists that require user interaction, you can model a complete process to run fully automated. Alternatively, you can define individual activities to run fully automated or with little human guidance. This can be useful if you want to run programs on backend systems, such as CICS (R) or IMS (TM).

Administering your workflow

MQ Workflow offers administration features for Buildtime. In addition, there is an Administration Utility to maintain and monitor your system.

Using Buildtime, the administrator can manage:

- Staff, program, data, and network definitions
- Staff authorization

Using the Administration Utility, the administrator can:

- Start and stop servers
- Monitor and analyze error logs

Complying with international standards

MQ Workflow allows you to have your business processes comply with ISO 9000.

As far as standards for workflow products are concerned, MQ Workflow adheres to the specified standards of the Workflow Management Coalition (WfMC). The WfMC was founded in 1993 and is an organization that focuses on the advancement of the workflow management technology and its use in industry. It is equally important for vendors and buyers of workflow products. The WfMC has more than 170 members, located in 24 countries around the world. IBM is a member of the WfMC organization. For details, see the *Workflow Handbook 1997, published in association with WfMC*.

Who is involved in workflow management

MQ Workflow is designed for everyone in an enterprise, who is involved in a business process.

Process modelers

Modelers build, test, and document process models. Modeling requires business analysis ability (for business processes) or systems analysis ability (for computer systems management processes). Modelers must understand the staff requirements, programs, and data structures used in the processes that they are modeling.

General users

General users perform activities assigned to them in the process models. They can select items on their worklists. Worklists reduce skill requirements by automating the task of finding and starting programs.

IT specialists and administrators

Programmers modify applications to be attached to process models, and also develop new applications. MQ Workflow supports these tasks with application programming interfaces (APIs), which are described in the *IBM MQSeries Workflow: Programming Guide*.

IT specialists and administrators install MQ Workflow, set up databases, define staff, authorize users, register programs, and define data structures used in process models. They also control the system to ensure that the servers run properly.

Process administrators

Administrators responsible for projects or processes control the running processes. Using MQ Workflow, they can start, interrupt, and resume processes, and they can change work assignments, for example, to balance workload.

Chapter 3. Building a workflow model

This chapter describes the components of a workflow model and how you can create such a model using the Buildtime component of MQ Workflow.

Components of a workflow model

The workflow model consists of three main components as illustrated in Figure 3. Building a model of your ″real″ business processes involves the definition of processes, including the network of activities. It also involves defining the workflow participants in an organization and the IT resources you need to implement your workflow.

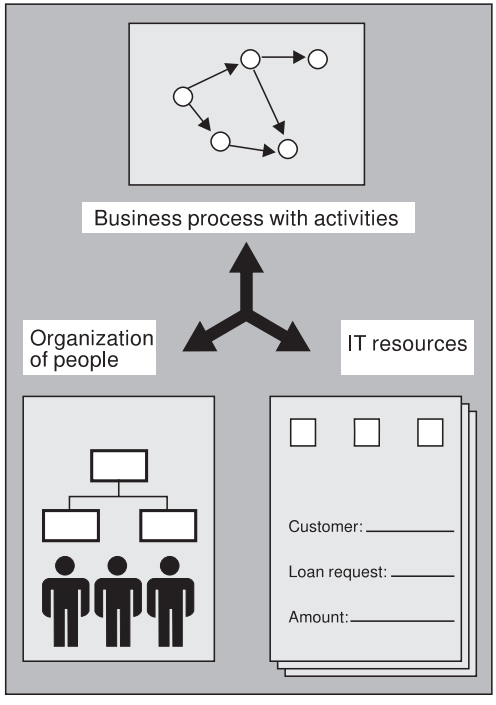

Figure 3. Building a workflow model

Looking more closely at a business process, for example, requesting a loan from a bank, there are many questions to ask. These might include:

- What are the activities that belong to a process?

They can be programs or even manual activities. They can also represent other processes, which are then called subprocesses.

- What is the sequence in which these activities must be carried out?

  To define the flow of work, you must specify the order in which activities must be performed. This includes the definition of when to start and end a certain activity.
- Can some of these activities be carried out in parallel?

  For your workflow to perform efficiently, you can define conditions for processing activities in parallel.
- Which organizational units are involved?

  The execution of processes often spans several different organizational units in an enterprise. You can define the staff and the organization to which they belong.
- Which application programs are involved in checking customer information?

  For program activities you define programs or tools that you need when running your processes.
- What kind of data is involved?

  For activities you also need to define the data and data structures that are part of your workflow.

The answers to these questions provide the basis for the workflow model. With MQ Workflow you can create a graphical model and define this information. To run your day-to-day business, you can then implement and use these process definitions to automate your workflow.

Creating a workflow model

With MQ Workflow you draw a diagram of a process model with its different types of activities. In addition, you define the properties for all the components that you define for your workflow, such as your organization, including staff, as well as the data, programs, and IT resources needed.

If a process definition becomes too complex, you can use process activities to encapsulate subprocesses. The advantage of such an approach is that you can reuse subprocesses in other processes. You can also define subprocesses first and then integrate them into other processes and thus constantly refine your workflow model. This approach offers you the flexibility to modify your model and add processes or subprocesses whenever you want. As an alternative, you can group several units of work and add them as a block to your process model. However, you can use a block only for the process for which you created it.

Drawing a process

MQ Workflow uses directed graphs to draw processes. This helps to prevent modeling errors, such as creating endless loops.

Figure 4 shows the tree view with processes that are already defined in the left pane of the application window. In the right pane, the diagram view of a selected process is displayed.

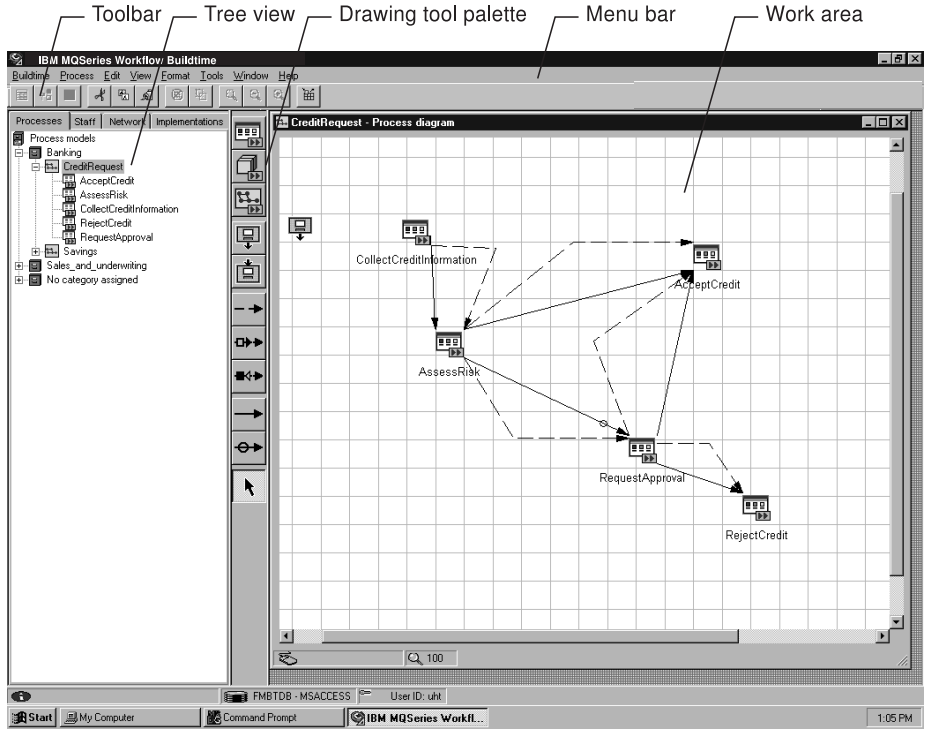

Figure 4. Modeling a process

The drawing tool palette is displayed between the tree view and the diagram view. The tool palette contains the icons you can use when modeling your activities. These are shown in Table 1.

Table 1. Icons for activities

	A program activity defines a program that you start from a worklist in Runtime of MQ Workflow.
	A process activity defines another process (subprocess) that you can define to start automatically. You can also define the process so that a user can start it from a worklist.

Table 1. Icons for activities (continued)

	A block activity defines a set of activities that can be repeated until an exit condition is met. The block is used to define a do-until loop. You can also define a block to group activities in a complex model.

You can also use your own icons to represent the different types of activities involved in your workflow. Your customized icons are then displayed on the worklists of Runtime users instead of the default icons of MQ Workflow.

Adding process logic

If the order in which activities start is important for your process, you can control this by linking them with *control connectors*, which you also choose from the tool palette. When the process is running, the *conditions* you define on these connectors are used to determine which activities are started and which are not. You can also link activities and blocks with *data connectors* if the data resulting from one activity is required by a subsequent one. For example, if the credit scoring of a loan request is positive, the next task to do is to send a letter to your customer, confirming that the loan request is accepted.

Figure 4 on page 13 shows an example of a process diagram dealing with a banking loan request. The customer requests a loan for a certain amount. There are program activities, which are called CollectCreditInformation and AssessRisk. Depending on the outcome, for example, of the credit scoring, the next activity to be started is either AcceptCredit or RequestApproval if the amount in question is too high for immediate approval.

Assigning staff to a process

In addition to defining processes and their activities, you assign your members of staff who must carry out the activities in running the day-to-day business. You can define your organization and staff, as well as specify roles that exist within your organization. For example, a staff member can have more than one role and staff members of different organization units can have the same role. This allows you to define your processes without specifying names of individuals. At run time, MQ Workflow resolves the defined organization units and roles with the specific individuals. This ensures that only eligible persons receive the work items for which they are responsible. This approach is called *dynamic staff resolution*. The advantage of dynamic assignments is that you do not have to change your process definitions if your staff changes or the responsibilities within your organization change. For example, the staff to perform an activity can be members of certain roles or they can be members of an organization unit.

If you specify names of persons in your workflow model who must perform an activity in a running process, the assignment is *static* and must be changed if the members of your staff change.

Figure 5 shows the tree view of the staff page. In the right pane, you can see the definitions for a selected organizational unit.

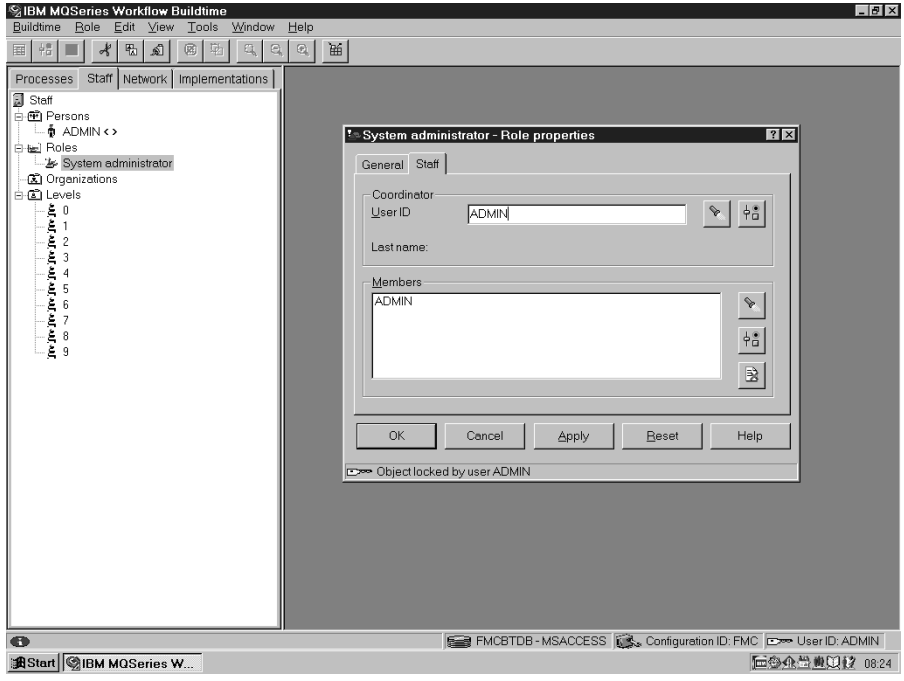

Figure 5. Assigning staff

Attaching programs to the workflow

In your process diagram, you define the properties for your business applications and tools that belong to the program activities. An application is started in Runtime when someone starts the corresponding program activity from a worklist or it is started automatically if it is defined that way. The applications can reside on other workstations or host systems using different operating systems.

If you choose to use a different application program for an activity in an existing process model, you can change the program registration without changing the entire workflow model. This means, you change the properties for the program. However, you must translate the process model again, before running the process. For more details, see "Translating the workflow model" on page 17.

Figure 6 shows the tree view of data and programs on the left of the application window and the properties for a program on the right. Adding programs to the workflow model includes defining the parameters

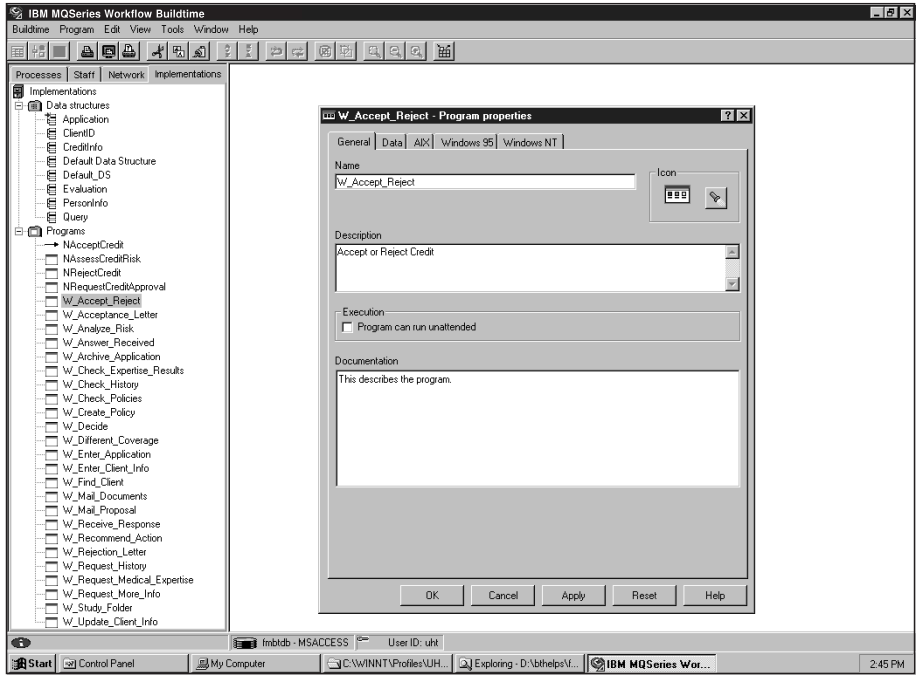

Figure 6. Adding programs and data

needed for starting these programs. In addition, definitions for passing data to a program as well as returning the appropriate data must be added.

Adding data to the workflow

Data that passes between a process and its activities is defined by the data connectors in your process diagram. For data to be available at run time of a process, you must define the properties of your data structures. For example, if process-relevant data needs to be passed from one activity to the next one, MQ Workflow uses *input containers* and *output containers*.

Using *data connectors*, you can define the data that must be passed to the input container of an activity for processing. At run time, program data is then processed by the program or subprocess that you defined as implementation for this activity. Any resulting data that is needed by another activity, is then passed to the output container of the processing activity.

For data that needs to be passed from one activity to the next, you must define the data structure to be used. Each data structure consists of members. For example, a data structure that is used to define an address could have

members for the street name and city name. The data type of a data structure member can be one of the basic MQ Workflow data types, such as string, long, or float. However, it can also refer to another data structure that you have previously defined. If a data structure refers to another data structure, this is called a *nested* data structure. MQ Workflow offers a default data structure that you can use. The data structures are also shown in the tree view as shown in Figure 6 on page 16.

If the origin and target data structures represent the same data structure, MQ Workflow automatically maps this data from the origin data container to the target data container. However, if there is a difference between the two data structures, you can specify matching types of data structure members for both input and output containers.

For example, when a user in Runtime starts a program activity to update the street name of a customer address in a database, the update is under program control of the activity. If the program started by the activity returns the street name to MQ Workflow using the appropriate API, the new name is passed from the output container of the first activity to the input container of the next activity. You find details about how to use the APIs in the *IBM MQSeries Workflow: Programming Guide*.

Adding IT resources to the workflow

For your workflow to be fully operational and execute all the activities automatically, you must add the IT resources needed to carry out the defined processes. In Buildtime you define the servers and other resources that you want to use in the workflow. For details about the resources that you need to define, see "Chapter 5. Architecture overview" on page 25.

Translating the workflow model

After you have created your workflow model, you import it into Runtime, verify, and translate it into a form that can be used by users of the Runtime component. It is then called a *process template*.

Translating the workflow model saves the current state of a process definition. All data structure and program information is copied into the process template.

MQ Workflow uses a built-in verify function that prevents you from modeling loops in your workflow model. MQ Workflow also checks if the data structures match and that conditions are semantically correct. This includes checking if program registration is done. In addition, MQ Workflow verifies start, end, and transition conditions for activities that you define for your workflow.

Using the MQ Workflow Client, you can then start an instance of a translated process. MQ Workflow navigates through the process and automates the sequencing of activities.

For information about the components of MQ Workflow that are involved in modeling processes, see "The server components" on page 29, and for details on how to create a workflow model, see *IBM MQSeries Workflow: Getting Started with Buildtime*.

Chapter 4. Running your business processes

This chapter describes how you can manage the daily work with MQ Workflow and what the tasks are you can do.

With an MQ Workflow Client or a Web Client, you can start and monitor the processes as they are defined in Buildtime. If you are authorized, you can manage processes that are already running. The administrator of MQ Workflow ensures that system resources are up-to-date and running.

The architecture of MQ Workflow allows you to use a standard *MQ Workflow client*, the *MQ Web Client*, or a *custom client*. To create your own custom client, MQ Workflow offers you APIs for the client functions. These APIs are described in the *IBM MQSeries Workflow: Programming Guide*.

Getting work done

With a client, you can start an instance of a process and work with predefined activities in a process. The MQ Workflow Client tree view is shown in Figure 7 on page 20. It displays the various lists you can use to work with predefined processes and their activities.

As described in "Creating a workflow model" on page 12, you define processes, their activities, the data and programs to be used as well as the processing conditions that are vital at run time. The MQ Workflow client uses *worklists* to display pending activities that belong to a defined process. The activities that must be performed by individuals are called *work items*.

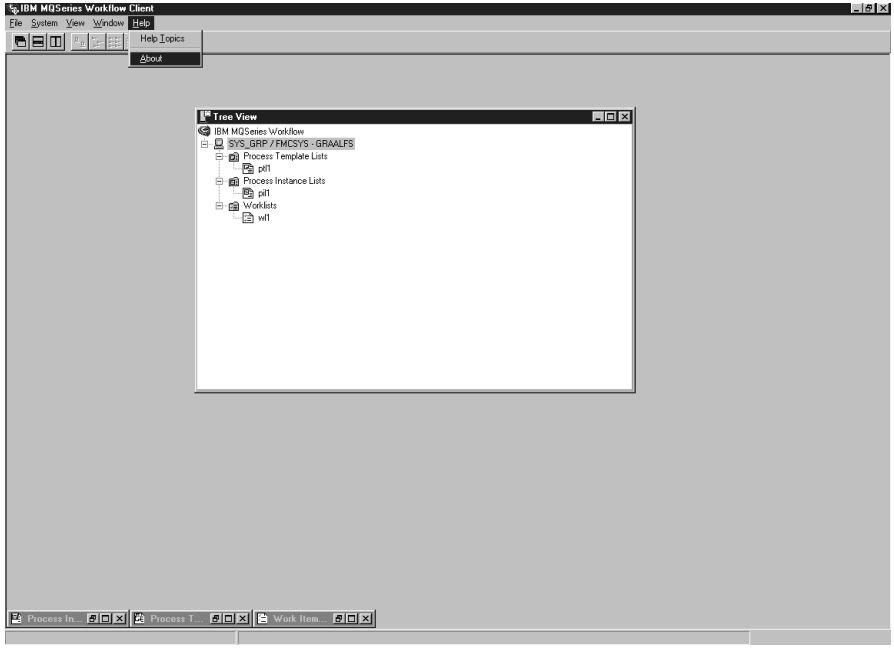

Figure 7. The Client tree view

Working with work items on a worklist

All activities that are due to be performed appear as work items on the worklists of the assigned staff. Programs that support a task can be started automatically in MQ Workflow if they are defined that way. The worklist in the MQ Workflow client displays all pending activities assigned to a user in any running process. Before sending an activity to a user's worklist, MQ Workflow makes sure that:

- The routing of activities is performed according to staff assignments
- The sequencing of activities is correct
- Activities are placed in ready status

The worklist helps to organize and do the work. MQ Workflow ensures that for an activity:

- The data needed is provided
- You can access the online documentation provided by the modeler
- Programs can be defined to start automatically

Note: MQ Workflow allows you to define the refresh policy in Buildtime for later use in Runtime worklists. Depending on the amount of activities that your workflow model contains, this can impact system performance negatively. You define the **Refresh policy (push)** for your

workflow model in Buildtime, and users of Runtime can then decide if they want to get their worklists refreshed automatically.

Choosing the worklist view

You can choose what is displayed on a worklist. For example, a worklist can show all activities that are ready to be started. You can sort your worklist by date and time received, process category, and other criteria.

You can also choose from predefined types of worklists. A modeler can create different worklists in Buildtime.

Intervening in the workflow

Anyone who is authorized can control running processes. You can start, interrupt, and resume processes. You can also change work assignments. You can transfer an activity to a colleague if you are authorized to do so. Similarly, if you are authorized, you can transfer an activity from one person's worklist to another's. This allows you to speed up execution of a certain process activity if there are too many work items on one person's worklist and the worklist of someone else is empty.

Working on notifications

Persons are notified if processes, activities, and notifications are not completed in a specified time. In Buildtime, the process modeler can specify a period of time in which:

- A process must finish
- Each activity defined for the process must finish
- The person who receives a notification has to act on the notification

MQ Workflow sends notifications automatically to the specified persons. This is yet another way in which speedy execution of processes is ensured.

Monitoring and analyzing processes

MQ Workflow enhances monitoring of work in progress that improves responsiveness to your staff and customers. You can monitor the status of a running process. You can see the workflow as it is happening.

MQ Workflow stores audit trails of running processes in a log file. Audit trails log significant events that occur as a process is running, for example, the times that activities start and end. You can use mining and analysis tools to analyze the log file and check the performance of your processes.

Part 2. The system architecture of MQ Workflow

Chapter 5. Architecture overview

This chapter describes the architecture of MQ Workflow and its hierarchical system structure. It also describes the components that belong to a system and the communication between components, which is based on MQSeries message queuing. There is also a section describing the relational database support and the APIs that are available.

Depending on the size of your organization, you can set up your installation using more than one MQ Workflow system. MQ Workflow is a client/server system with a hierarchical structure.

Figure 8 shows an example of the system hierarchy of MQ Workflow, with the domain name **Your company**, system group name **California Division**, and system names **San Jose Branch**, **San Francisco Branch**, and **Los Angeles Branch**.

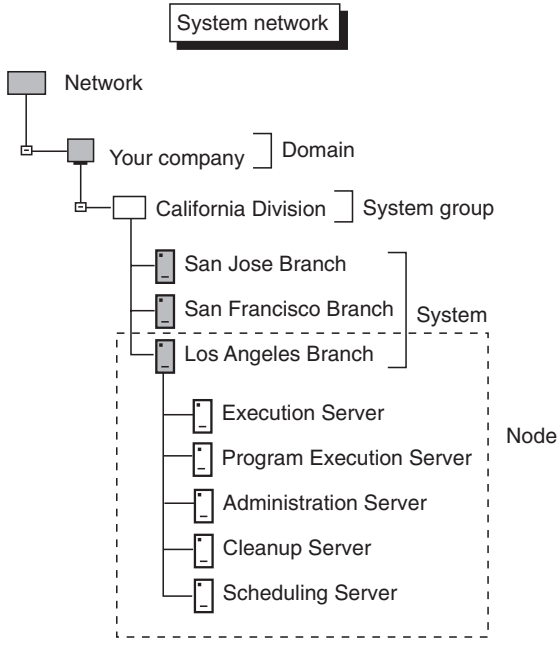

Figure 8. System network of MQ Workflow

The top level in the hierarchy is called *domain*, representing all or parts of your organization. The next lower level in a domain is the *system group*. For example, this can be a geographical area. Within a system group, there can be

several *systems* accessing the same database. A system contains the client/server components that you need to run your processes. You define the names with a maximum length of eight characters for each of the components in Buildtime.

Depending on the size of your installation and the operating system you want to use, the server components can reside on one or more physical machines. The system components that are installed on one physical machine are called a *node*.

The flexibility to distribute system components on more than one processor is a major advantage of the system structure of MQ Workflow. Using message queuing for communication between the various components and distributing the components on clients and servers provides a reliable environment for your workflow.

What is a domain?

The workflow model that you define or import into MQ Workflow is valid for all the systems in the domain. This includes all definitions for staff, data structures, programs, IT resources, and process templates. You can define properties that specify the behavior of your MQ Workflow installation at this highest hierarchical level. Whatever you define at the highest level is *inherited* by all lower levels. If you want to have different definitions at a lower level, you can define them explicitly and the settings are then valid for that level.

For example, if you specify for a system group that you do *not* want to keep audit trail information, this is valid for all the systems and the system group of the domain. However, if you want to change this setting for a particular system and want to keep audit trail information, you can then define it for this system.

What is a system group?

Within a system group, all the systems share the same database. If you decide to install more than one system for the system group, you can distribute the workload for process execution and still have the advantage of sharing the same data and the same workflow model.

Because any mission-critical system, such as a database management system or a workflow management system, must be highly available, the MQ Workflow architecture provides the appropriate mechanisms. On z/OS (OS/390), the systems in a system group can reside on different z/OS (OS/390) images, however, with the same database using DB2 data sharing.

For distributed platforms, the three-tier configuration, as explained in "What is a system?", is an important factor for implementing a highly available system.

Similar to the system structure, there is also a hierarchy for the communication that is needed between components to support fast and efficient message transfer. Communication between systems within a system group is optimized for the message traffic within the system group and is driven by the message queuing facility of MQSeries. By using the MQSeries support for clusters, a system group can also be addressed as a single logical system.

The advantages of using clusters are:

- Increased availability of your queues
- Faster throughput of messages
- More even distribution of workload in your network

For more details about MQSeries clusters, see "Workload management with MQSeries clusters" on page 40.

What is a system?

The components of an MQ Workflow system are designed for a multi-tier structure. The scope of each tier is clearly defined to exploit the available computing resources. The major components and their respective tiers are shown in Figure 9 on page 28. If you want to use a Web Client, this adds another tier to the overall structure of the system as shown in "Web-based Client architecture" on page 33.

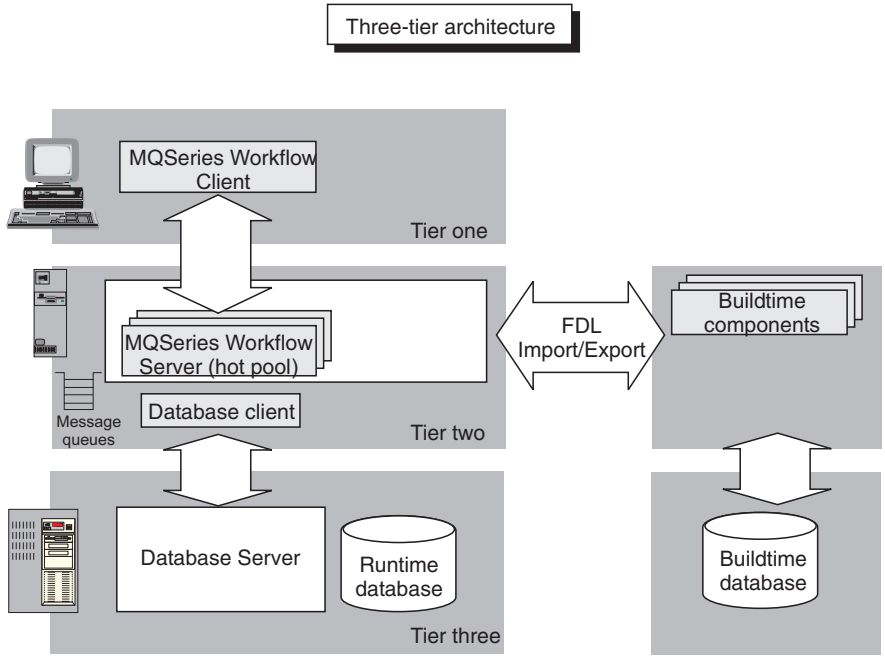

Figure 9. Three-tier architecture

Tier one: Client components

Tier one represents the client APIs of MQ Workflow and the clients that use these APIs. Clients are responsible for executing the program activities that interact with users. Clients are also responsible for giving users access to the workflow management system, that is, access work items, access running processes, and monitor processes. The communication with servers is through MQSeries, using the client message layer of MQ Workflow. For details about the Web-based client architecture, refer to "Web-based Client architecture" on page 33.

Tier two: Server components and Buildtime

Tier two represents the server components and Buildtime of MQ Workflow. The server components are responsible for managing the execution of processes at run time. You can distribute the components of the second tier across several machines to achieve workload balancing. For communication between server components, message queuing with MQSeries is used.

Tier three: Databases for Buildtime and Runtime

Tier three represents the database tier. The database holds workflow-relevant data for a system group of MQ Workflow. This includes status and setup information. For communication between the

Database Server and its client, the transport protocols supported by DB2(R) are supported. For additional information, refer to "Relational database support" on page 38.

Depending on the size of your organization and the size of the workflow model, the database can also reside on the same machine as all other server components. The system then consists of only two tiers as shown in Figure 10.

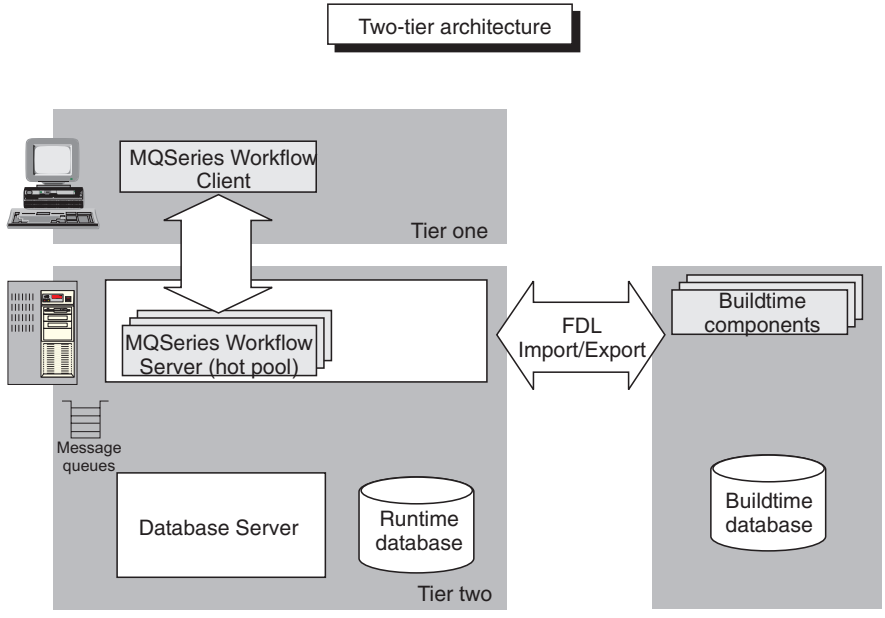

Figure 10. Two-tier architecture

In a two-tier structure, the server components, the message queuing components of MQSeries, and the database management system reside on the second tier.

If you install MQ Workflow on z/OS, the system always consists of two tiers, because the server component is connected to a database subsystem.

The server components

The server components coordinate and manage an MQ Workflow system and its clients. Server components are also responsible for keeping track of processes and administering process execution.

Figure 11 on page 30 shows the server components that make up an MQ Workflow system.

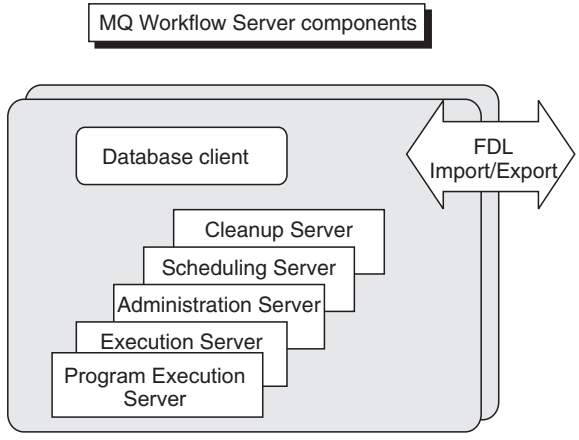

Figure 11. Server components of MQ Workflow

Execution Server

The Execution Server is responsible for the execution of processes. If a process contains activities for automatic execution, the Execution Server performs the right activities at the right time without human intervention. If staff is involved in a process, the right work item is moved to the right person at the right time. To achieve this, the Execution Server performs the following tasks:

- Interpreting the process definitions, that is, definitions for the process flow, staff, programs, and data
- Creating the process instances and managing their execution, including starting, stopping, or suspending them
- Navigating between activities and creating the work items needed for processing
- Managing process states and logging events
- Maintaining the worklists of Runtime users

The Execution Server maintains the information about the states of all activities in the MQ Workflow Runtime database. It acts as a Database Client using the communication mechanisms of the database management system to communicate with the Database Server.

Administration Server

The Administration Server manages an MQ Workflow system. The Administration Server communicates with all other components in a system or system group. It is the working center of the administration component. The Administration Server is responsible for the availability, operation, and error recovery of all server components.

To access the Administration Server, MQ Workflow offers an Administration Utility. You can use the command interface to send requests to the Administration Server using the Administration Utility on the distributed platforms or from the system console on z/OS (OS/390).

Scheduling Server
The Scheduling Server controls and manages notification for activities that must be performed within a certain time frame. For example, if activities or work items are overdue for a process, the Scheduling Server changes the activity state to *expired*. For work items, the Scheduling Server sends notifications to the worklists of the relevant persons.

Cleanup Server
The Cleanup Server is responsible for physically deleting process instances that are finished. Depending on the definitions you set for your system, finished processes are deleted immediately or later in the day when the system is idle.

Note: Within an MQ Workflow system group, only one Cleanup Server and one Scheduling Server are needed to serve all the systems within this system group.

The **Program Execution Server** is described in "The components for program execution" on page 37.

The Buildtime components

With Buildtime you can create workflow models and define system resources.

Buildtime offers a graphical editor for creating process models. Other features in Buildtime allow you to define your organization and the programs you want to use in your workflow as well as your network definitions.

You can also import existing workflow models (FDL) into MQ Workflow or export them in the MQ Workflow Definition Language (FDL). You can also export workflow models in HTML if you want to print them.

When you decide that a workflow model is ready to use, you translate the model into a template that can be started from an MQ Workflow Client and managed by the server components.

For more information about modeling processes, see "Chapter 3. Building a workflow model" on page 11.

The client components

The MQ Workflow Client starts processes and monitors their execution. The Administration Utility administers the system and the Program Execution Agent invokes application programs that are used in the workflow.

Figure 12 shows the client components that make up an MQ Workflow system.

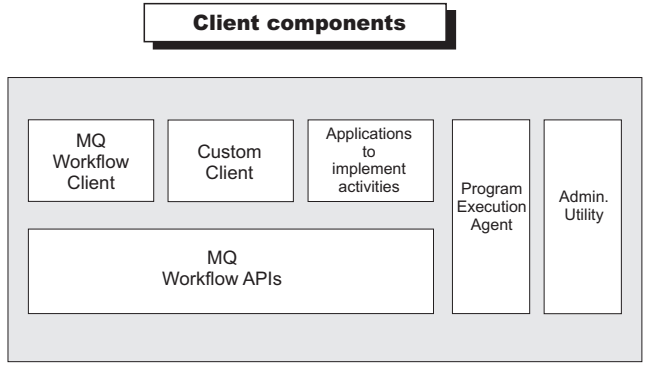

Figure 12. Client components of MQ Workflow

MQ Workflow Client

With an MQ Workflow Client you can start the execution of processes, use worklists to manage work items, and monitor your processes. MQ Workflow offers a standard MQ Workflow Client that is based on APIs. For more functional details, refer to "Chapter 4. Running your business processes" on page 19.

If you want to design your own interface to perform worklist or monitoring tasks with a custom client, you can use the APIs to implement the client functions. For a description of the APIs, refer to Figure 17 on page 42.

If you want to use a Web-based interface as your graphical user interface, you can use the Web Client or write your own custom Web-based client, as described in "Web-based Client architecture" on page 33.

Administration Utility

The Administration Utility is the administrator's user interface to request services from the Administration Server. Using the Administration Utility you can start and stop an MQ Workflow system. Although the Administration Server regularly checks the state of all servers, you can also use the Administration Utility on the distributed platforms or the system console on z/OS (OS/390) to list

the current state of any server. You can also access error logs to check if any problems exist within your system.

The **Program Execution Agent** is described in "The components for program execution" on page 37.

Web-based Client architecture

If you want to use a Web-based client, for example, to implement e-business solutions, MQSeries Workflow offers you alternatives in addition to the Standard Client. The MQ Workflow Web Client offers full process and worklist control as well as process monitoring functions.

Alternatively, you can create a custom client using Enterprise Java Beans (EJB) to perform client tasks. Figure 13 on page 34 shows the relevant components of the overall system structure.

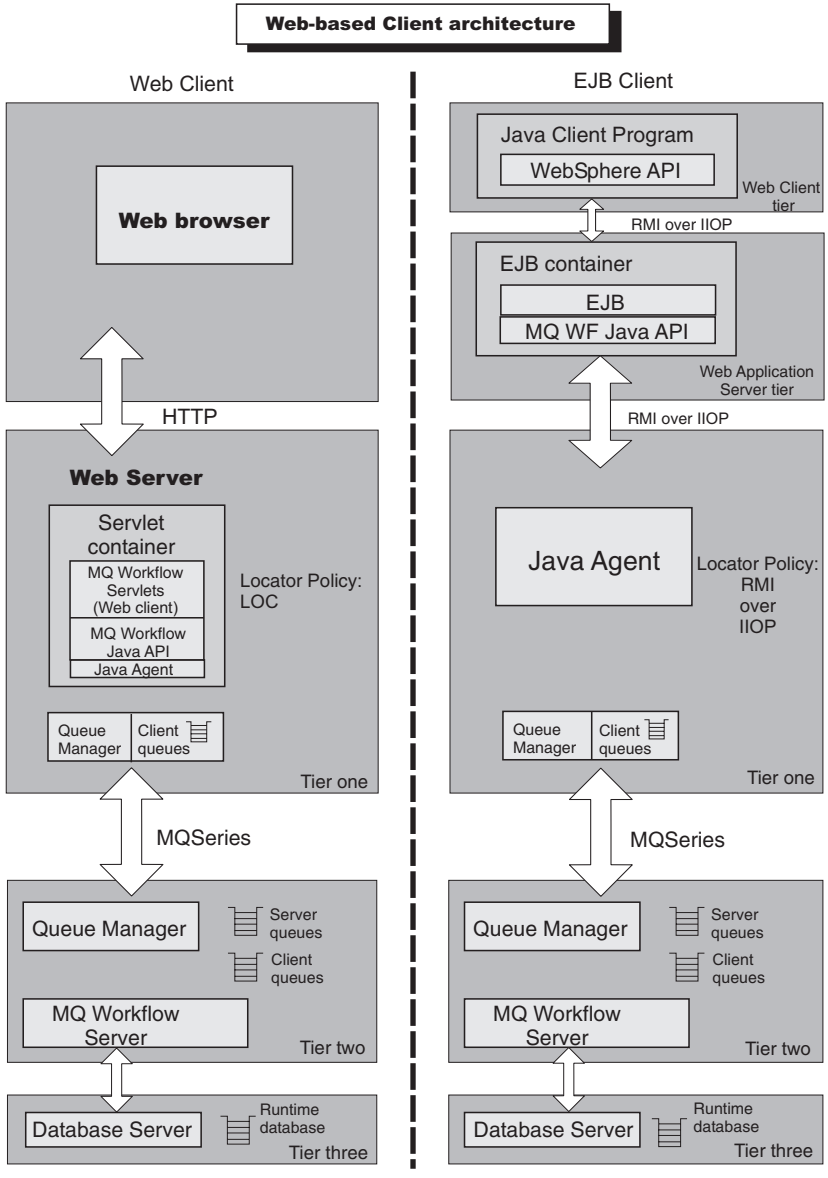

Figure 13. Web-based Client architecture

The Web Client, as shown on the left, is a servlet and runs on the Web Server. For the client, you need a browser with JavaScript support. To achieve the best possible performance results, use the MQ Workflow Java API with the LOC locator policy. The recommended client connection type for the MQ Workflow Java API is *client with queue manager* as described in "Communication support" on page 35.

If you want to use your own thin client solution, as shown on the right in Figure 13 on page 34, you can use the WebSphere API and also write your own applications that use the MQSeries Workflow Java API on the *Web Application Server tier*. Communication is based on RMI-IIOP. With RMI-IIOP, you can use a locator policy that combines the RMI ease of use with the IIOP protocol, which offers reliability and interoperability. The Java Agent can run as a generic server in WebSphere. The Java Agent can be configured to be started automatically by WebSphere and also to be restarted automatically in case of problems.

The configuration is similar for both of these thin clients and is described in detail in the *IBM MQSeries Workflow: Installation Guide*.

Communication support

Server and client components communicate with each other using the message queuing facility of MQSeries. Whenever a component of MQ Workflow requests a service from another component, a message is sent using the queue name of the appropriate component. Each server instance works on the basis of request messages. The clients get the responses through response queues.

Each system is logically connected to an MQSeries queue manager. Each system is connected to every other system in a system group through a message channel.

You can choose between different setup types to connect clients to the server components as shown in Figure 14 on page 36.

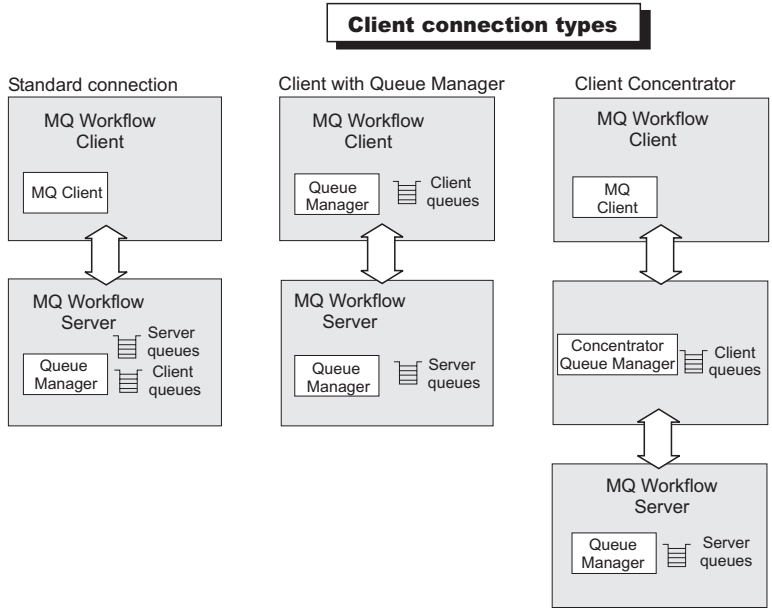

Figure 14. Client connection types

For your installation, you can choose between these connection types:

Standard connection

If you use a standard connection, the MQ Workflow Client has a fixed association with a particular server and uses the MQSeries client API. This type of connection is recommended only if you intend to attach a small number of clients where no workload balancing is required.

Client with queue manager

If you use the client with queue manager, the MQ Workflow Client uses the MQSeries server APIs to communicate with its local queue manager, which then uses message channels for the communication with the MQ Workflow Server. This configuration is recommended for MQ Workflow Clients that also act as servers, such as the Web Server in the Web Client configuration, as shown in Figure 13 on page 34. This setup type can also be useful if an MQSeries server installation already exists on a machine. The Client with queue manager configuration helps to improve performance of your system.

Client concentrator

If you use the client concentrator, the MQ Workflow Clients use the MQSeries client to connect to the concentrator queue manager, which then uses message channels for the communication with the MQ Workflow Server. This configuration helps to reduce the use of system resources for the MQ Workflow Server, because no clients are

connecting directly to the MQ Workflow Server. Many clients can connect to one concentrator and many client concentrators can be used within the configuration.

This configuration is suitable if you need to connect large numbers of clients. Such a configuration typically consists of a few client concentrators, which are connected to several MQ Workflow systems belonging to the same MQSeries cluster. This helps to balance the workload accross the MQ Workflow systems in a cluster.

For additional information, see "Workload management with MQSeries clusters" on page 40.

The components for program execution

To invoke application programs within your workflow, MQ Workflow uses a Program Execution Agent for the execution of executable programs (EXE or DLL) on a client machine and a Program Execution Server for the automatic and unattended execution of backend programs on the server.

The Program Execution Server is available for z/OS only and supports the invocation of IMS and CICS transactions. It has an open architecture and, therefore, new types of program invocation interfaces can be plugged in. In addition, MQ Workflow offers a message-based interface to support the interaction with applications that are not necessarily integrated in an MQ Workflow process. You can use your own User-defined Program Execution Server to start XML-based applications as described in "MQ Workflow message formats" on page 43. Figure 15 shows the components for program execution in a workflow model.

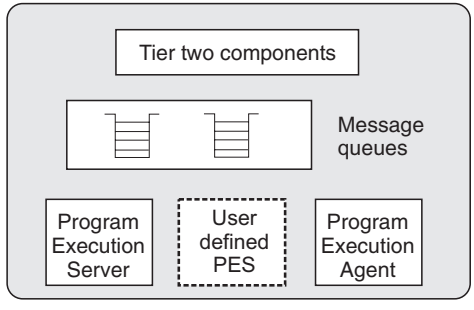

Figure 15. Components for program execution

Program Execution Agent

The Program Execution Agent invokes and manages task-relevant application programs or tools that you define in the workflow model. Application programs can run on a different operating system from

the one used for the server components of MQ Workflow. The Program Execution Agent is used to start attended programs from the client machine. You can, however, also start programs that run in unattended mode on platforms, where no Program Execution Server is available.

Program Execution Server
The Program Execution Server invokes application programs that you define in the workflow model. These programs run in subsystem environments of z/OS, for example, IMS or CICS. In addition to program execution in IMS or CICS, you can plug in programs, using the Program Execution Server invocation exit. These exits use invocation protocols for applications that are running in other environments. The Program Execution Server is responsible for the mapping of parameters to the format that is expected by the invoked application. This applies if an application does not use the container API to access its input or output data, for example, a data structure that is passed in a CICS COMMAREA. For backend applications that run in unattended mode, no user involvement is needed. The Program Execution Server is designed to incorporate security checking, such as z/OS Security Server (RACF). For details on how to incorporate application programs, see the *IBM MQSeries Workflow: Programming Guide*.

User-defined Program Execution Server
Compared to the Program Execution Server, which is a component of MQ Workflow, the User-defined Program Execution Server must be implemented as a custom program using the documented message formats that apply for it. This means that the User-defined Program Execution Server must be able to handle the messages as shown in Figure 18 on page 43. The message format is XML, as described in "MQ Workflow message formats" on page 43, and the protocol is restricted to the message types that are needed to invoke and finish a program. For error handling, a defined error message is used.

Relational database support

MQ Workflow is designed for relational database management systems, such as DB2(R) Universal Database to store the process models and process-relevant data. The Runtime database is also involved in the navigation logic between the process steps at run time, using SQL calls.

The database stores all data that is relevant for process execution. Whenever the state of a process activity changes, this information is saved. Equally, the data supplied to the input and output containers is saved in the database during process execution.

MQ Workflow can use the database support of DB2 Universal Database, with its multi-user features. If you define a server as a hot pool in MQ Workflow (hot pool instances), each server instance establishes its own database connection. This helps to improve performance of the overall system.

Access to the Buildtime database is through the Open Database Connectivity support (ODBC). This offers you to choose either a Microsoft Jet database engine or a DB2 Universal Database.

Because the database that is used for process execution is independent of the database that is needed for modeling purposes, there are several advantages:

- It allows you to use the modeling database without any performance impact on running processes.
- ODBC allows you to access different relational databases by using the appropriate ODBC driver, such as Microsoft Access or DB2 Universal Database. For example, this offers you the flexibility to choose Windows NT as the operating system for Buildtime and z/OS for your large production server at run time.
- The database tables are optimized for the purpose they serve. The Buildtime database has different access patterns to its data from the Runtime database. For example, the Runtime database is optimized to deal with many transactions, because it handles all the work requests. The Buildtime database is optimized for modeling purposes.

Workload management

The MQ Workflow architecture allows you to manage your workload dynamically, depending on the setup you choose for your enterprise.

Workload management within a system

Depending on the size of your installation as well as the number of processes and activities, you can define more than one instance of the Execution Server. If you have more than one instance of the Execution Server, the load is shared between these instances for better performance. This concept is called *hot pooling*. Each one of these instances has its own connection to the Database Server. This allows you to distribute the workload throughout the workflow system.

The MQSeries messaging and queuing functions allow multiple server instances to read from the same input queue. Each server instance executes in its own operating system process. In z/OS, you can configure MQ Workflow Server address spaces so that they contain multiple server instances and each one is running independently in a subtask of the server address space. All server instances that belong to the same MQ Workflow system group use the same database.

You can also define additional systems within a system group to balance the workload for your workflow system.

On a z/OS (OS/390) system, the workload manager (WLM) controls and adds or removes address spaces automatically, depending on the actual workload of the system. This means that control of the number of server instances in a hot pool is delegated to WLM. With WLM you can even separate different kinds of workflow transactions into different WLM service classes. If the service classes have different levels of importance and performance goals, WLM manages these subclasses of the workload individually to achieve the specified performance goals. To monitor the performance goals and to check the overall system behavior, use z/OS (OS/390) performance monitors, such as Resource Measurement Facility (RMF).

In the context of MQ Workflow servers, WLM is responsible for the server instance management as well as message priority management. This does not necessarily mean that the messages are handled in a first-in first-out order (FIFO), but rather according to the performance goals and level of importance as defined for the service classes of MQ Workflow transactions.

Workload management with MQSeries clusters

MQSeries queue manager clusters help to distribute the workload throughout all the systems of a system group. Within MQ Workflow, a system group represents an MQSeries cluster and the queue managers of the individual systems all belong to that cluster. Therefore, the Execution Servers of all the systems within an MQ Workflow system group represent a *logical* Execution Server and can be addressed through a single logical queue.

Messages can be handled by any one of the queue managers that host an instance of that queue as shown in Figure 16 on page 41. This means that MQ Workflow need not explicitly name the queue manager when sending messages. Balancing the load between the queues within the system group is done automatically by the underlying MQSeries cluster support.

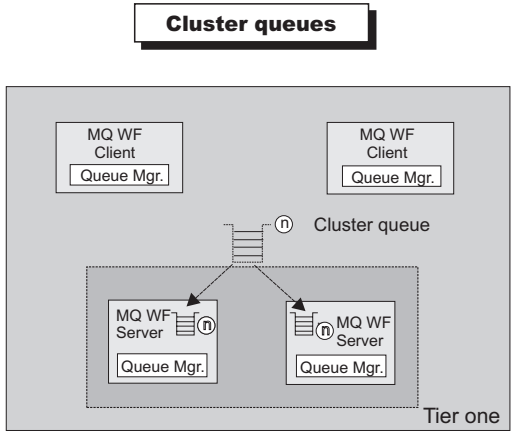

Figure 16. Cluster queues

System architecture for high availability

MQ Workflow is designed to support continuous operation for 7 days a week and 24 hours a day. There are different influencing factors for availability, such as the system architecture supporting hot pools, and cluster support with shared queues. Depending on the type of computer, different system features are available. Starting at the low end of the scale, there are small-processor systems and ending at the high end of the scale, high-availability systems can be chosen, such as the IBM zSeries systems, which are parallel sysplex systems.

On distributed platforms, the database tier (tier three) is still a single point of failure, where automatic recovery or restart is needed, which is offered, for example, by the AIX HACMP cluster support. On z/OS (OS/390), in case of a system failure in the cluster (sysplex), the remaining systems, except for the failed one, continue to process all of the workload.

Integrating applications

MQ Workflow offers a wide range of APIs that allow you to build your own components and integrate both new and existing applications into your workflow model.

API support

MQ Workflow offers APIs to support the interaction between the MQ Workflow server and client components. In addition, you can use APIs to invoke applications that you need for your workflow tasks. Using the client APIs you can build your own custom clients. For example, you can create your own MQ Workflow Client for users to manage their work items.

Figure 17 gives you an overview of the APIs that MQ Workflow offers.

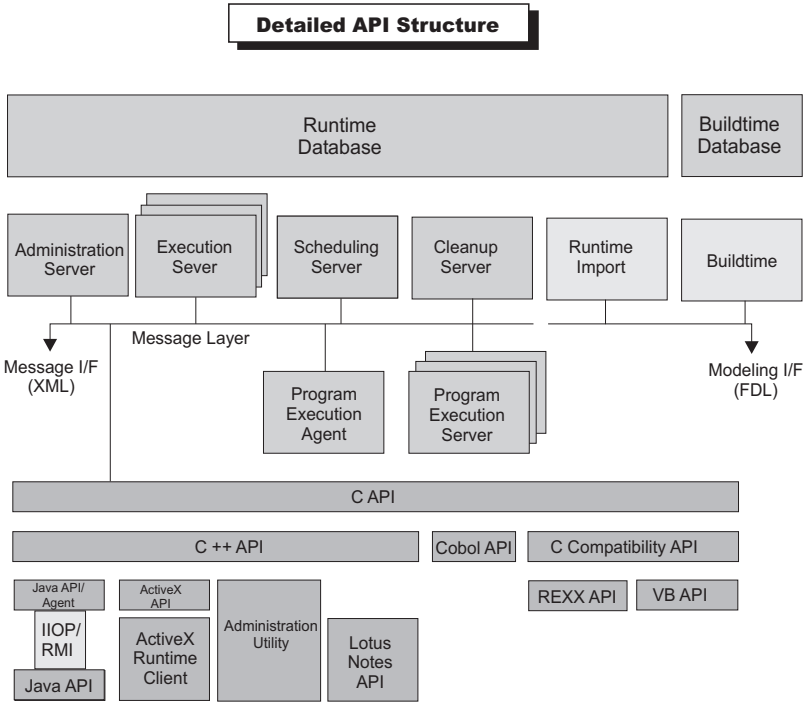

Figure 17. API structure

The C language API represents the collection of all client features that are needed at run time. The C language API is a compiler-independent interface, that is, you can work with the operating system and compiler vendor of your choice. For details about compiler versions, check the announcement information of MQ Workflow. You can use the APIs to perform the following functions:

Handling worklists
: You can implement the management of work items similar to the Standard MQ Workflow Client of MQ Workflow. The API supports features, such as starting and monitoring a process.

Managing containers
: This allows applications that are invoked by MQ Workflow to access the input and output containers of the program activities.

The API is additionally available in bindings for C++, COBOL, ActiveX, and Java.

For a detailed description of the APIs supported by MQ Workflow, see the *IBM MQSeries Workflow: Programming Guide*.

MQ Workflow message formats

For communication between the MQ Workflow servers and the components for program execution, different message formats are used, as shown in Figure 18.

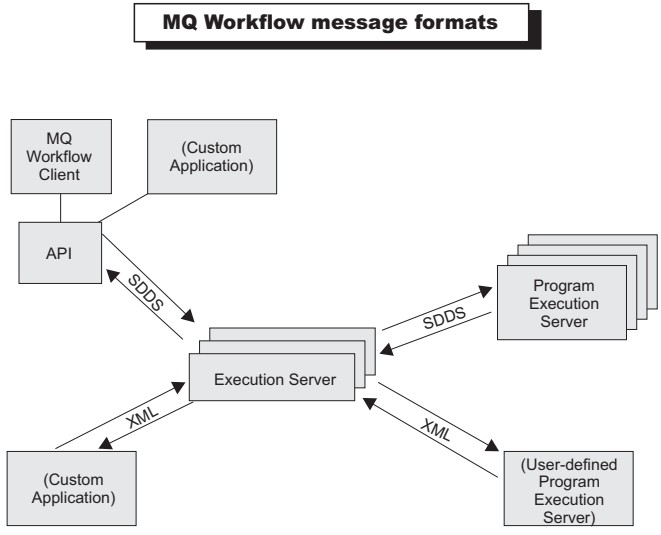

Figure 18. Message formats

MQ Workflow Server message format

The MQ Workflow Server uses its own message format, which is called *self-describing data stream (SDDS)*, to exchange messages with the Program Execution Server as shown in Figure 18. To start application programs that are implemented as a User-defined Program Execution Server, MQ Workflow uses messages in a defined XML format.

In MQ Workflow, you can incorporate applications running on any of the platforms that are supported by MQSeries without installing the MQ Workflow components for these applications. This allows you to run your processes in a flexible way.

Message-based interfaces using eXtensible Markup Language (XML)

In addition to the standard API support, MQ Workflow also offers a message-based interface. This message-based interface, which uses XML as its format, supports the interaction with applications outside the scope of MQ Workflow.

Instead of using APIs or the Standard Client interface of MQ Workflow, you can use the message-based interface into the server components. This allows you to start a process instance, using an XML message. This message can be created by your own in-house application or by any other application that can deal with XML messages, for example, MQSeries Integrator Version 2.0.

The message-based interface can also be used for invoking programs within a process. You can write your own User-defined Program Execution Server to interface with MQ Workflow or use a standard application, such as MQSeries Integrator. This allows you to use the message-based interface independent of the operating platform you are using.

For a detailed description about how to implement XML messages supported by MQ Workflow, see the *IBM MQSeries Workflow: Programming Guide*.

Part 3. Appendixes

Appendix. Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will

be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Deutschland
Informationssysteme GmbH
Department 3982
Pascalstrasse 100
70569 Stuttgart
Germany

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 information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

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.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include

the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. 1993, 2001. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States, other countries, or both:

- AIX
- CICS
- DB2
- DB2 Universal Database
- IBM
- IMS
- MQSeries
- OS/390
- RACF
- z/OS

Lotus Notes is a registered trademark, and Domino and Lotus Go Webserver are trademarks of Lotus Development Corporation.

Microsoft, Windows, Windows NT and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.

Glossary

This glossary defines important terms and abbreviations used in this publication. If you do not find the term you are looking for, refer to the index or the *IBM Dictionary of Computing*, New York: McGraw-Hill, 1994.

A

administration server. The MQ Workflow component that performs administration functions within an MQ Workflow system. Functions include starting and stopping of the MQ Workflow system, performing error management, and participating in administrative functions for a system group.

activity. One of the steps that make up a process model. This can be a program activity, process activity, or block activity.

activity information member. A predefined data structure member associated with the operating characteristics of an activity.

API. Application Programming Interface.

application programming interface. An interface provided by the MQ Workflow workflow manager that enables programs to request services from the MQ Workflow workflow manager. The services are provided synchronously.

audit trail. A relational table in the database that contains an entry for each major event during execution of a process instance.

authorization. The attributes of a user's staff definition that determine the user's level of authority in MQ Workflow. The system administrator is allowed to perform all functions.

B

bend point. A point at which a connector starts, ends, or changes direction.

block activity. A composite activity that consists of a group of activities, which can be connected with control and data connectors. A block activity is used to implement a Do-Until loop; all activities within the block activity are processed until the exit condition of the block activity evaluates to true. See also *composite activity.*

Buildtime. An MQ Workflow component with a graphical user interface for creating and maintaining workflow models, administering resources, and the system network definitions.

C

cardinality. (1) An attribute of a relationship that describes the membership quantity. There are four types of cardinality: One-to-one, one-to-many, many-to-many, and many-to-one. (2) The number of rows in a database table or the number of different values in a column of a database table.

child organization. An organization within the hierarchy of administrative units of an enterprise that has a parent organization. Each child organization can have one parent organization and several child organizations. The parent is one level above in the hierarchy. Contrast with *parent organization*.

cleanup server. The MQ Workflow component that physically deletes information in the MQ Workflow Runtime database, which had only been deleted logically.

composite activity. An activity which is composed of other activities. Composite activities are block activities and bundle activities.

container API. An MQ Workflow API that allows programs executing under the control of MQ Workflow to obtain data from the input and output container of the activity and to store data in the output container of the activity.

control connector. Defines the potential flow of control between two nodes in the process. The actual flow of control is determined at run time based on the truth value of the transition conditions associated with the control connector.

coordinator. A predefined role that is automatically assigned to the person designated to coordinate a role.

D

data connector. Defines the flow of data between containers.

data container. Storage for the input and output data of an activity or process. See *input container* and *output container*.

data mapping. Specifies, for a data connector, which fields from the associated source container are mapped to which fields in the associated target container.

data structure. A named entity that consists of a set of data structure members. Input and output containers are defined by reference to a data structure and adopt the layout of the referenced data structure type.

data structure member. One of the variables of which a data structure is composed.

default control connector. The graphical representation of a standard control connector, shown in the process diagram. Control flows along this connector if no other control path is valid.

domain. A set of MQ Workflow system groups which have the same meta-model, share the same staff information, and topology information. Communication between the components in the domain is via message queuing.

dynamic staff assignment. A method of assigning staff to an activity by specifying criteria such as role, organization, or level. When an activity is ready, the users who meet the selection criteria receive the activity to be worked on. See also *level*, *organization*, *process administrator*, and *role*.

E

end activity. An activity that has no outgoing control connector.

execution server. The MQ Workflow component that performs the processing of process instances at runtime.

exit condition. A logical expression that specifies whether an activity is complete.

export. An MQ Workflow utility program for retrieving information from the MQ Workflow database and making it available in MQ Workflow Definition Language (FDL) or HTML format. Contrast with *import*.

F

fixed member. A predefined data structure member that provides information about the current activity. The value of a fixed member is set by the MQ Workflow workflow manager.

(FDL) MQ Workflow Definition Language. The language used to exchange MQ Workflow information between MQ Workflow system groups. The language is used by the import and export function of MQ Workflow and contains the workflow definitions for staff, programs, data structures, and topology. This allows non-MQ Workflow components to interact with MQ Workflow. See also *export* and *import*.

fork activity. An activity that is the source of multiple control connectors.

form. In Lotus Notes, a form controls how you enter information into Lotus Notes and how that information is displayed and printed.

formula. In Lotus Notes, a mathematical expression that is used, for example, to select documents from a database or to calculate values for display.

fully-qualified name. A qualified name that is complete; that is, one that includes all names in the hierarchical sequence above the structure member to which the name refers, as well as the name of the member itself.

I

import. An MQ Workflow utility program that accepts information in the MQ Workflow definition language (FDL) format and places it in an MQ Workflow database. Contrast with *export*.

input container. Storage for data used as input to an activity or process. See also *source* and *data mapping*.

L

level. A number from 0 through 9 that is assigned to each person in an MQ Workflow database. The person who defines staff in Buildtime can assign a meaning to these numbers such as rank and experience. Level is one of the criteria that can be used to dynamically assign activities to people.

local user. Identifies a user during staff resolution whose home server is in the same system group as the originating process.

local subprocess. A subprocess that is processed in the same MQ Workflow system group as the originating process.

logical expression. An expression composed of operators and operands that, when evaluated, gives a result of true, false, or an integer. (Nonzero integers are equivalent to false.) See also *exit condition* and *transition condition*.

M

manager. A predefined role that is automatically assigned to the person who is defined as head of an organization.

message queuing. A communication technique that uses asynchronous messages for communication between software components.

N

navigation. Movement from a completed activity to subsequent activities in a process. The paths followed are determined by control connectors, their associated transition conditions, and by the start conditions of activities. See also *control connector*, *exit condition*, *transition condition*, and *start condition*.

node. (1) The generic name for activities within a process diagram. (2) The operating system image that hosts MQ Workflow systems.

notification. An MQ Workflow facility that can notify a designated person when a process or activity is not completed within the specified time.

notification work item. A work item that represents an activity or process notification.

O

organization. An administrative unit of an enterprise. Organization is one of the criteria that can be used to dynamically assign activities to people. See *child organization* and *parent organization*.

output container. Storage for data produced by an activity or process for use by other activities or for evaluation of conditions. See also *sink*.

P

parent organization. An organization within the hierarchy of administrative units of an enterprise that has one or more child organizations. A child

is one level below its parent in the hierarchy. Contrast with child *child organization*.

parent process. A process instance that contains the process activity which started the process as a subprocess.

pattern activity. A single and simple activity in a bundle activity from which multiple instances, called pattern activity instances, are created at run time.

person (pl. people). A member of staff in an enterprise who has been defined in the MQ Workflow database.

predefined data structure member. A data structure member predefined by MQ Workflow and used for communication between user applications and MQ Workflow Runtime.

process. Synonymously used for a process model and a process instance. The actual meaning is typically derived from the context.

process activity. An activity that is part of a process model. When a process activity is executed, an instance of the process model is created and executed.

process administrator. A person who is the administrator for a particular process instance. The administrator is authorized to perform all operations on a process instance. The administrator is also the target for staff resolution and notification.

process category. An attribute that a process modeler can specify for a process model to limit the set of users who are authorized to perform functions on the appropriate process instances.

process definition. Synonym for *process model*.

process diagram. A graphical representation of a process that shows the properties of a process model.

process instance. An instance of a process to be executed in MQ Workflow Runtime.

process instance list. A set of process instances that are selected and sorted according to user-defined criteria.

process instance monitor. An MQ Workflow client component that shows the state of a particular process instance graphically.

process management. The MQ Workflow Runtime tasks associated with process instances. These consist of creating, starting, suspending, resuming, terminating, restarting, and deleting process instances.

process model. A set of processes represented in a process model. The processes are represented in graphical form in the process diagram. The process model contains the definitions for staff, programs, and data structures associated with the activities of the process. After having imported and translated the process model into a process template, the process template can be executed over and over again. *Workflow model* and *process definition* are synonyms.

process monitor API. An application programming interface that allows applications to implement the functions of a process instance monitor.

process-relevant data. Data that is used to control the sequence of activities in a process instance.

process status. The status of a process instance.

process template. A fixed form of a process model from which process instances can be created. It is the imported and translated form in MQ Workflow Runtime. See also *process instance*.

process template list. A set of process templates that have been selected and sorted according to user-defined criteria.

program. A computer-based application that serves as the implementation of a program activity or as a support tool. Program activities reference executable programs using the logical

names associated with the programs in MQ Workflow program registrations. See also *program registration*.

program activity. An activity that is executed by a registered program. Starting this activity invokes the program. Contrast with *process activity*.

program execution agent. The MQ Workflow component that manages the implementations of program activities, such as .EXE and .DLL files.

program registration. Registering a program in MQ Workflow so that sufficient information is available for managing the program when it is executed by MQ Workflow.

R

role. A responsibility that is defined for staff members. Role is one of the criteria that can be used to dynamically assign activities to people.

S

scheduling server. The MQ Workflow component that schedules actions based on time events, such as resuming suspended work items, or detecting overdue processes.

server. The servers that make up an MQ Workflow system are called Execution Server, Administration Server, Scheduling Server, and Cleanup Server.

sink. The symbol that represents the output container of a process or a block activity.

source. The symbol that represents the input container of a process or a block activity.

specific resource assignment. A method of assigning resources to processes or activities by specifying their user IDs.

standard client. The MQ Workflow component, which enables creation and control of process instances, working with worklists and work items, and manipulation of personal data of the logged-on user.

start activity. An activity that has no incoming control connector.

start condition. The condition that determines whether an activity with incoming control connectors can start after all of the incoming control connectors are evaluated.

subprocess. A process instance that is started by a process activity.

substitute. The person to whom an activity is automatically transferred when the person to whom the activity was originally assigned is declared as absent.

support tool. A program that end users can start from their worklists in the MQ Workflow MQ Workflow Client to help complete an activity.

symbolic reference. A reference to a specific data item, the process name, or activity name in the description text of activities or in the command-line parameters of program registrations. Symbolic references are expressed as pairs of percent signs (%) that enclose the fully-qualified name of a data item, or either of the keywords _PROCESS or _ACTIVITY.

system. The smallest MQ Workflow unit within an MQ Workflow domain. It consists of a set of the MQ Workflow servers.

system group. A set of MQ Workflow systems that share the same database.

system administrator. (1) A predefined role that conveys all authorizations and that can be assigned to exactly one person in an MQ Workflow system. (2) The person at a computer installation who designs, controls, and manages the use of the computer system.

T

top-level process. A process instance that is not a subprocess and is started from a user's process instance list or from an application program.

transition condition. A logical expression associated with a conditional control connector. If specified, it must be true for control to flow along the associated control connector. See also *control connector*.

translate. The action that converts a process model into a Runtime process template.

U

user ID. An alphanumeric string that uniquely identifies an MQ Workflow user.

V

verify. The action that checks a process model for completeness.

W

workflow. The sequence of activities performed in accordance with the business processes of an enterprise.

Workflow Management Coalition (WfMC). A non-profit organization of vendors and users of workflow management systems. The Coalition's mission is to promote workflow standards for workflow management systems to allow interoperability between different implementations.

workflow model. Synonym for *process model*.

work item. Representation of work to be done in the context of an activity in a process instance.

work item set of a user. All work items assigned to a user.

worklist. A list of work items assigned to a user and retrieved from a workflow management system.

worklist view. List of work items and notifications selected from a work item set of a user according to filter criteria which are an attribute of a worklist. It can be sorted according to sort criteria if specified for this worklist.

Bibliography

To order any of the following publications, contact your IBM representative or IBM branch office.

MQ Workflow publications

This section lists the publications included in the MQSeries Workflow library.

- *IBM MQSeries Workflow: Concepts and Architecture*, GH12-6285, explains the basic concepts of MQ Workflow. It also describes the architecture of MQ Workflow and how the components fit together.
- *IBM MQSeries Workflow: Getting Started with Buildtime*, SH12-6286, describes how to use Buildtime of MQ Workflow.
- *IBM MQSeries Workflow: Getting Started with Runtime*, SH12-6287, describes how to get started with the MQ Workflow Client.
- *IBM MQSeries Workflow: Programming Guide*, SH12-6291, explains the application programming interfaces (APIs).
- *IBM MQSeries Workflow: Installation Guide*, SH12-6288, contains information and procedures for installing and customizing MQ Workflow.
- *IBM MQSeries Workflow: Administration Guide*, SH12-6289, explains how to administer an MQ Workflow system.

Related publications

- *Frank Leymann, Dieter Roller, Production Workflow: Concepts and Techniques (New Jersey: Prentice Hall PTR, 1999)*
- *Frank Leymann, Dieter Roller, "Workflow-based Applications", IBM Systems Journal 36, no. 1 (1997): 102–123*, you can also refer to the Internet: http://www.almaden.ibm.com/journal/sj361/leymann.html
- *Workflow Handbook 1997, published in association with WfMC*, edited by Peter Lawrence

Readers' Comments — We'd Like to Hear from You

IBM MQSeries Workflow
Concepts and Architecture
Version 3.3

Publication No. GH12-6285-03

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name _____ Address _____

Company or Organization _____

Phone No. _____

Readers' Comments — We'd Like to Hear from You
GH12-6285-03

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape

PLACE
POSTAGE
STAMP
HERE

IBM Deutschland Entwicklung GmbH
Information Development, Dept. 0446
Schoenaicher Strasse 200
71032 Boeblingen
Germany

Fold and Tape **Please do not staple** Fold and Tape

Cut or Fold
Along Line

Part Number: CT896IE
Program Number: 5697-FM3

Printed in Denmark by IBM Danmark A/S

(1P) P/N: CT896IE

GH12-6285-03