Workflow Management Coalition

The Workflow Management Coalition Specification

Workflow Management Coalition
The Workflow Reference Model

Document Number TC00-1003

Document Status - Draft 1.1

12-Jun-96

Author:
David Hollingsworth

Send comments to d.hollingsworth@wsr0104.wins.icl.co.uk

Table of Contents

1. Introduction

1.1. Background

Work Flow Management is a fast evolving technology which is increasingly being exploited by businesses in a variety of industries. Its primary characteristic is the automation of processes involving combinations of human and machine-based activities, particularly those involving interaction with IT applications and tools. Although its most prevalent use is within the office environment in staff intensive operations such as insurance, banking, legal and general administration, etc, it is also applicable to some classes of industrial and manufacturing applications .

Many software vendors have WFM products available today which involve WFM technology and there is a continual introduction of more products into the market. The availability of a wide range of products within the market has allowed individual product vendors to focus on particular functional capabilities and users have adopted particular products to meet specific application needs. However, there are, as yet, no standards defined to enable different WFM products to work together, which is resulting in incompatible "islands" of process automation.

The WFM Coalition is a grouping of companies who have joined together to address the above situation. It has been recognised that all work flow management products have some common characteristics, enabling them potentially to achieve a level of interoperability through the use of common standards for various functions. The WFM Coalition has been established to identify these functional areas and develop appropriate specifications for implementation in workflow products. It is intended that such specifications will enable interoperability between heterogeneous workflow products and improved integration of workflow applications with other IT services such as electronic mail and document management, thereby improving the opportunities for the effective use of workflow technology within the IT market, to the benefit of both vendors and users of such technology.

1.2. Purpose

The purpose of this document is to provide a framework to support the development of the various specifications described above. It provides a common "Reference Model" for workflow management systems identifying their characteristics, terminology and components, enabling the individual specifications to be developed within the context of an overall model for workflow systems. The detailed specifications will be developed as separate documents.

1.3. Scope

This document covers the concepts, terminology, general structure of a workflow management system, its major functional components and the interfaces and information interchange flows between them. It identifies the areas appropriate for standardisation and illustrates the potential interoperability scenarios which may be supported through the use of common standards. It also discusses, where appropriate, the applicability of existing standards to workflow management systems and their integration with other standard IT services. It does not cover wider aspects of business process engineering which lie outside the use of information technology to support the business process.

1.4. Audience

The intended audience of this document is the work flow coalition membership as well as others that are interested in the efforts of the coalition and wish to understand the top level technical architecture which underpins the work of the Coalition. The document is intended for a moderately technical audience but extensive prior knowledge of workflow systems is not assumed.

1.5. How to read this document

Chapter 2 provides a general introduction to the concepts of workflow systems technology, its evolution, the business context and background on the types of systems which may incorporate this type of technology. If you are unfamiliar with workflow technology you should start here; if you are already familiar with workflow management systems, consider starting at Chapter 3.

Chapter 3 discuses the internal structure of workflow systems, the major functional components and the nature of their interactions. It introduces the top level architecture and identifies the various interfaces which may be used to support interoperability between different system components and integration with other major IT infrastructure components.

Chapter 4 provides a general overview of the workflow application programme interface (WAPI), comments on the necessary protocol support for open interworking and discusses the principles of conformance to the specifications. It identifies those aspects of the specifications which are required to support various classes of interoperability. The detailed WAPI specifications are published as separate specification documents (see cross references below).

1.6. Cross References

WFMC SC00 - 1002	WFM Coalition Proposal Information
WFMC SC00 - 1006	WFM Coalition Technical Committee Operations
WFMC TC00 - 1008	Interoperability White Paper
WFMC TC00 - 1009	Client application API descriptions
WFMC TC00 - 1010	Workflow Definition Read/Write Descriptions
WFMC TC00 - 1011	Terminology and Glossary
WFMC TC00 - 1013	Workflow APIs - Naming Conventions

1.7. Revision History

This issue (1.1) is the second major version, incorporating the following changes from the previous version (0.6):

- Incorporation of updated terminology and glossary
- Incorporation of monitoring and metrics interface within the reference model
- Updated material on workflow interoperability (derived from the Coalition work on the Workflow Interoperability White Paper) and its associated interface operations, clarifying the various interoperability scenarios and proposed areas for open interoperability

Revision History (continued)

- Incorporation of comments on the (optional) use of organisational roles within the basic model
- Incorporation of comments clarifying the use of workflow relevant data within the basic model
- Incorporation of minor changes to align with the output of other Coalition Working Groups, particularly the initial API specifications
- Improvements in clarification and consistency in various areas throughout the text, including amended document structure

Version 1.1 incorporates minor editorial changes as a result of the TC meeting in Vienna (10th Nov 94), plus revisions to improve consistency with other Coalition documentation.

2. Workflow Systems Overview

2.1. What is Workflow?

Workflow is concerned with the automation of procedures where documents, information or tasks are passed between participants according to a defined set of rules to achieve, or contribute to, an overall business goal. Whilst workflow may be manually organised, in practice most workflow is normally organised within the context of an IT system to provide computerised support for the procedural automation and it is to this area that the work of the Coalition is directed.

> Definition - Workflow
>
> The computerised facilitation or automation of a business process, in whole or part.

Workflow is often associated with Business Process Re-engineering, which is concerned with the assessment, analysis, modelling, definition and subsequent operational implementation of the core business processes of an organisation (or other business entity). Although not all BPR activities result in workflow implementations, workflow technology is often an appropriate solution as it provides separation of the business procedure logic and its IT operational support, enabling subsequent changes to be incorporated into the procedural rules defining the business process. Conversely, not all workflow implementations necessarily form part of a BPR exercise, for example implementations to automate an existing business procedure.

A Workflow Management System is one which provides procedural automation of a business process by management of the sequence of work activities and the invocation of appropriate human and/or IT resources associated with the various activity steps.

> Definition - Workflow Management System
>
> A system that completely defines, manages and executes "workflows" through the execution of software whose order of execution is driven by a computer representation of the workflow logic.

An individual business process may have a life cycle ranging from minutes to days (or even months), depending upon its complexity and the duration of the various constituent activities. Such systems may be implemented in a variety of ways, use a wide variety of IT and communications infrastructure and operate in an environment ranging from small local workgroup to inter-enterprise. The WFMC Reference Model thus takes a broad view of workflow management, which is intended to accommodate the variety of implementation techniques and operational environments which characterise this technology.

Despite this variety, all WFM systems exhibit certain common characteristics, which provide a basis for developing integration and interoperability capability between different products. The Reference Model describes a common model for the construction of workflow systems and identifies how it may be related to various alternative implementation approaches.

At the highest level, all WFM systems may be characterised as providing support in three functional areas:

- the Build-time functions, concerned with defining, and possibly modelling, the workflow process and its constituent activities
- the Run-time control functions concerned with managing the workflow processes in an operational environment and sequencing the various activities to be handled as part of each process
- the Run-time interactions with human users and IT application tools for processing the various activity steps

Figure 1 illustrates the basic characteristics of WFM systems and the relationships between these main functions.

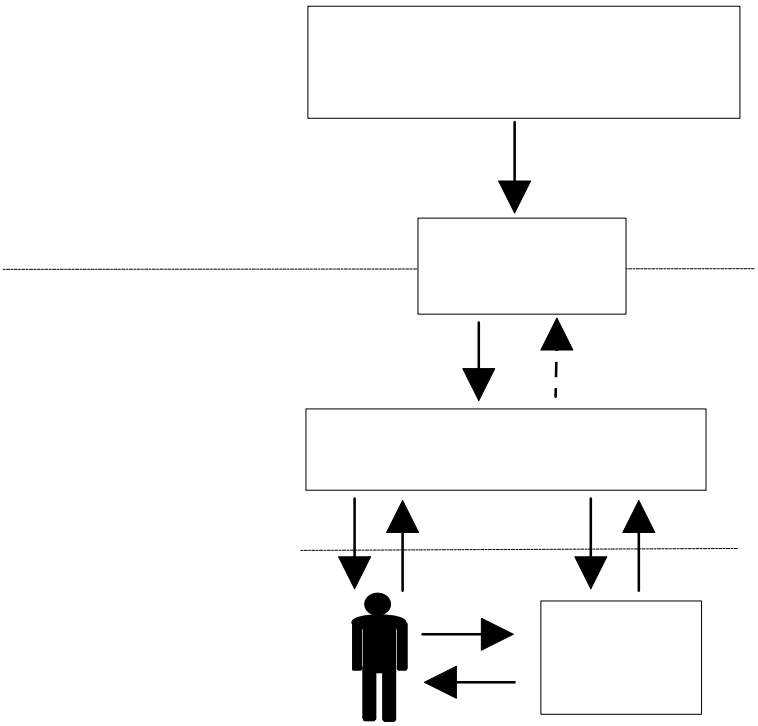

Figure 1- Workflow System Characteristics

2.1.1. Build-time Functions

The Build-time functions are those which result in a computerised definition of a business process. During this phase, a business process is translated from the real world into a formal, computer processable definition by the use of one or more analysis, modelling and system definition techniques. The resulting definition is sometimes called a process model, a process template, process metadata, or a process definition. For purposes of this document, the term 'process definition' will be used.

Definition - Process Definition

The computerised representation of a process that includes the manual definition and workflow definition.

A process definition normally comprises a number of discrete activity steps, with associated computer and/or human operations and rules governing the progression of the process through the various activity steps. The process definition may be expressed in textual or graphical form or in a formal language notation. Some workflow systems may allow dynamic alterations to process definitions from the run-time operational environment, as indicated by the feed-back arrow in the above diagram.

Coalition members do not consider the initial creation of process definitions to be an area of standardisation. Rather, this is considered to be a major distinguishing area between products in the marketplace. However, the result of the Build-time operation, the process definition, is identified as one of the potential areas of standardisation to enable the interchange of process definition data between different build-time tools and run-time products.

2.1.2. Run-time Process Control Functions

At run-time the process definition is interpreted by software which is responsible for creating and controlling operational instances of the process, scheduling the various activities steps within the process and invoking the appropriate human and IT application resources, etc. These run-time process control functions act as the linkage between the process as modelled within the process definition and the process as it is seen in the real world, reflected in the runtime interactions of users and IT application tools. The core component is the basic workflow management control software (or "engine"), responsible for process creation & deletion, control of the activity scheduling within an operational process and interaction with application tools or human resources. This software is often distributed across a number of computer platforms to cope with processes which operate over a wide geographic basis.

2.1.3. Run-time Activity Interactions

Individual activities within a workflow process are typically concerned with human operations, often realised in conjunction with the use of a particular IT tool (for example, form filling), or with information processing operations requiring a particular application program to operate on some defined information (for example, updating an orders database with a new record). Interaction with the process control software is necessary to transfer control between activities, to ascertain the operational status of processes, to invoke application tools and pass the appropriate data, etc. There are several benefits in having a standardised framework for supporting this type of interaction, including the use of a consistent interface to multiple workflow systems and the ability to develop common application tools to work with different workflow products.

2.1.4. Distribution & System Interfaces

The ability to distribute tasks and information between participants is a major distinguishing feature of workflow runtime infrastructure. The distribution function may operate at a variety of levels (workgroup to inter-organisation) depending upon the scope of the workflows; it may use a variety of underlying communications mechanisms (electronic mail, messaging passing, distributed object technology, etc). An alternative top-level view of workflow architecture which emphasises this distribution aspect is shown in figure 2 on the following page.

The workflow enactment service is shown as the core infrastructure function with interfaces to users and applications distributed across the workflow domain. Each of these interfaces is a potential point of integration between the workflow enactment service and other infrastructure or application components.

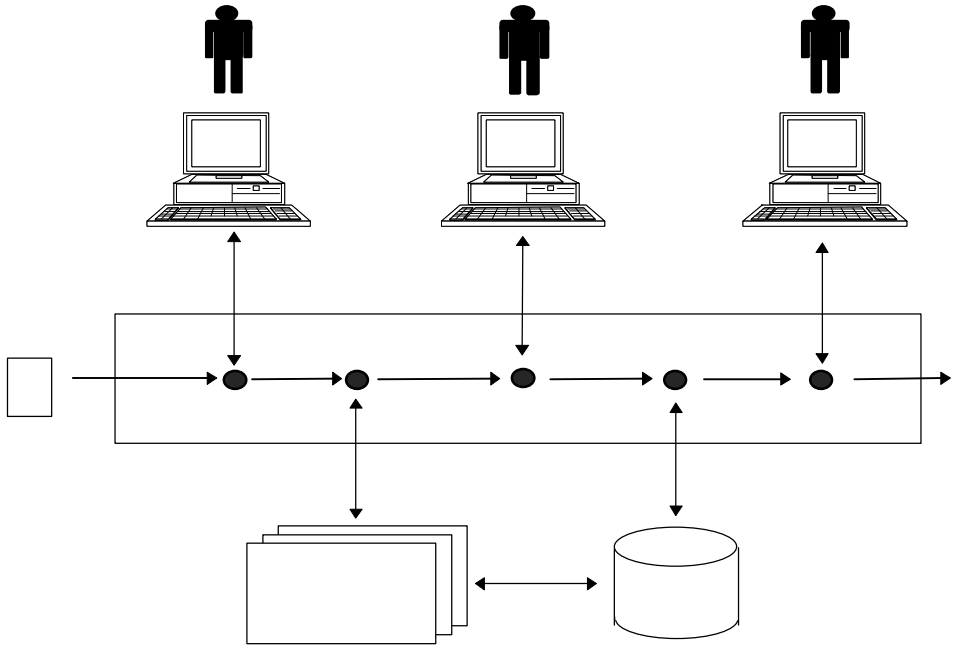

Fig 2 - Distribution within the workflow enactment service

The flow of work may involve the transfer of tasks between different vendors workflow products to enable different parts of the business process to be enacted on different platforms or sub-networks using particular products suited to that stage of the process. In this scenario the flow within the central box passes between two or more workflow products - for example activities 1,2 and 5 may be executed by one workflow system and activities 3 and 4 by a different system, with control passed between them at appropriate points within the overall workflow. Standards to support this transfer of workflow control enable the development of composite workflow applications using several different workflow products operating together as a single logical entity.

The full range of interfaces being defined by the WFM Coalition therefore covers:

- specifications for process definition data and its interchange
- interfaces to support interoperability between different workflow systems
- interfaces to support interaction with a variety of IT application types
- interfaces to support interaction withuser interface desktop functions
- interfaces to provide system monitoring and metric functions to facilitate the management of composite workflow application environments

These are further developed in Section 3.

2.2. The Evolution of Workflow

Many types of product in the IT market have supported aspects of workflow functionality for a number of years, yet it is only comparatively recently that its importance has been recognised in its own right. The evolution of workflow as a technology has thus encompassed a number of different product areas.

2.2.1 Image Processing

Workflow has been closely associated with image systems and many image systems have workflow capability either built-in or supplied in conjunction with a specific workflow product. Many business procedures involve interaction with paper-based information, which may need to be captured as image data as part of an automation process. Once paper based information has been captured electronically as image data, it is often required to be passed between a number of different participants for different purposes within the process, possibly involving interaction with other IT applications, thereby creating a requirement for workflow functionality.

2.2.2 Document Management

Document management technology is concerned with managing the lifecycle of electronic documents. Increasingly, this is including facilities for managing document repositories distributed within an organisation as a shared resource with facilities for routing documents (or even separate parts of documents) to individuals for information access or updating according to their specific roles relating to a specific document. The document may form part of a particular business procedure which requires access to the document by individual staff undertaking separate activities according to a particular sequence according to some procedural rules - i.e. a document-centric form of workflow.

2.2.3 Electronic Mail & Directories

Electronic mail provides powerful facilities for distributing information between individuals within an organisation or between organisations; the use of directory mechanisms not only provides a way of identifying individual participants within an email domain but also potentially recording information about individual user attributes, such as organisation roles or other attributes relating to business procedures. Thus electronic mail systems have themselves been progressing towards workflow functionality through the addition of routing commands to define a sequence of recipients for particular types of mail items in response to some form of identified business procedure.

2.2.4 Groupware Applications

The groupware industry has introduced a wide range of software applications designed to support and improve the interactions between groups of individuals. Initially many of these applications supported improvements in group working via informal processes, accessing group bulletin boards or diary/scheduling applications on an ad-hoc basis. As the scope of such applications has spread towards more formal business focussed group interactions there has been an increasing requirement to provide a more formal and controllable procedural framework to support the use of groupware applications. Workflow technology provides a solution to this type of requirement.

2.2.5 Transaction-based Applications

For many years applications to support certain classes of business procedures ("transactions") have been developed using transaction management facilities within TP monitors and/or Database Management software. From the initial centralised style of working, such application software has increasingly enabled the distribution of transaction based applications across a number of computer platforms. Transaction based applications typically exhibit important characteristics of robustness and support for "atomic" properties of the transaction; however, they do not typically exhibit a separation between the business procedure logic and the invocation of the various application tools which may be required to support individual activities within the business process. Over time, this is leading to a requirement to consolidate workflow capabilities to control the business procedures with the ability to invoke traditional transaction application programs for appropriate parts of the business process, as well as other types of application (document or office based, etc..) for other parts of the business process.

2.2 5 Project Support Software

Software to handle complex IT application project development (eg IPSEs - "Integrated Project Support Environments") has often provided a form of workflow functionality within the project environment, for "transferring" development tasks between individuals and routing information between individuals to support these tasks. In some cases this type of software has been generalised to support a wider, business-oriented view of process and a wider range of application tools - offering a more general workflow capability.

2.2.6 BPR and Structured System Design Tools

Business Process Re-engineering tools have provided IT based support for the activities of analysing, modelling and (re-)defining the core business processes of an organisation and the potential effects of change in such processes or organisational roles and responsibilities associated with such processes. This may include analysis of the process structure and information flows supporting it, the roles of individuals or organisational units within the process and actions taken in response to different events, etc. A natural extension of such tools is to facilitate the implementation of the process with IT support infrastructure to control the flows of work and associated activities within the business process.

2.2.7 Separation of workflow functionality

The market for workflow has evolved from requirements across a spectrum of the IT industry and is likely to continue to do so, with a wide range of products focussed on one or more particular aspects of the overall workflow requirement. Some may be provided in conjunction with other areas of technology, such as image processing or document management, others may be more general purpose. This multiplicity of products will allow wide choice for individual implementation circumstances and is recognised as something to be encouraged. However, it also increases the need for standards within the industry to enable different products to work together and integrate within a consistent overall architecture.

The reference architecture described in this document provides a framework which separates the various functions within a workflow environment and identifies various interface points at which product integration and interworking may be accomplished. It forms the template within which the individual interfaces and interchange specifications are being developed by the Coalition..

2.3. Product Implementation Model

Overview

Despite the variety in workflow products in the market, it has proved feasible to construct a general implementation model of a workflow system which can be matched to most products in the marketplace thereby providing a common basis for developing interoperability scenarios.

This approach identifies the main functional components within a workflow system and the interfaces between them as an abstract model. It is recognised that many different concrete implementation variants of this abstract model will exist and therefore the interfaces specified may be realised across a number of different platform and underlying distribution technologies. Furthermore not all vendors may choose to expose every interface between the functional components within the model; this will be dealt with by the specification of a variety of conformance levels which will identify the particular interworking functions where open interfaces are supported for multivendor integration.

The main functional components of a generic workflow system are illustrated in figure 3.

The generic model has three types of component:

- software components which provide support for various functions within the workflow system (shown in dark fill)
- various types of system definition and control data (shown unfilled) which are used by one or more software components
- applications and application databases (shown in light fill) which are not part of the workflow product, but which may be invoked by it as part of the total workflow system

The roles of the major functional components within this system are described below.

Process Definition Tool

The process definition tool is used to create the process description in a computer processable form. This may be based on a formal process definition language, an object relationship model, or in simpler systems, a script or a set of routing commands to transfer information between participating users. The definition tool may be supplied as part of a specific workflow product or may be part of a business process analysis product, which has other components to handle analysis or modelling of business operations. In this latter case there must be a compatible interchange format to transfer the process definitions to/from the run-time workflow software.

Process Definition

The process definition contains all necessary information about the process to enable it to be executed by the workflow enactment software. This includes information about its starting and completion conditions, constituent activities and rules for navigating between them, user tasks to be undertaken, references to applications which may to be invoked, definition of any workflow relevant data which may need to be referenced, etc.

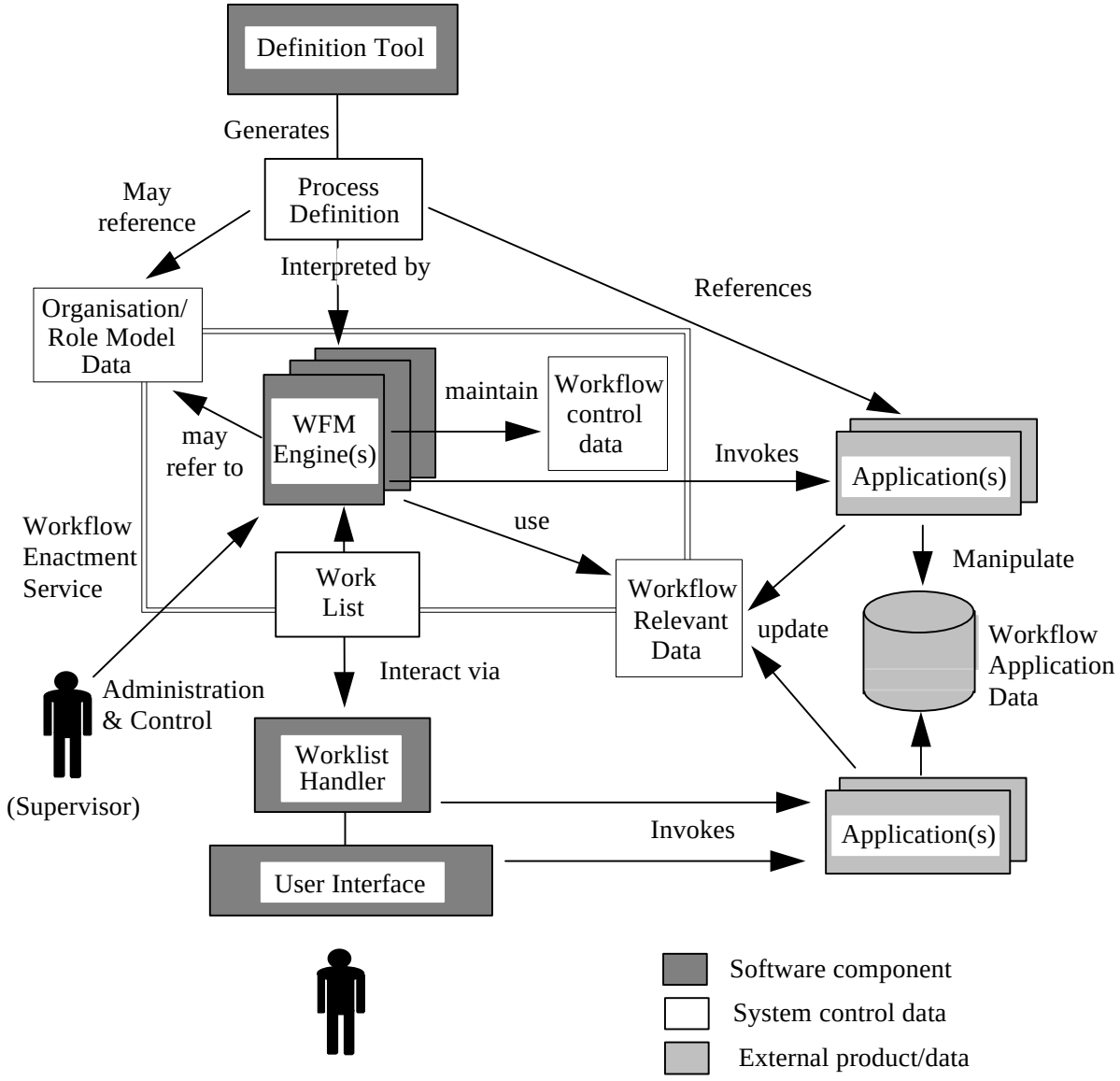

Figure 3 - Generic Workflow Product Structure

The process definition may refer to an Organisation / Role model which contains information concerning organisational structure and roles within the organisation (eg an organisational directory). This enables the process definition to be specified in terms of organisational entities and role functions associated with particular activities or information objects, rather than specific participants. The workflow enactment service then has the responsibility of linking organisational entities or roles with the specific participants within the workflow runtime environment.

Workflow Enactment Service

The workflow enactment software interprets the process description and controls the instantiation of processes and sequencing of activities, adding work items to the user work lists and invoking application tools as necessary. This is done through one or more co-operating workflow management engines, which manage(s) the execution of individual instances of the various processes. The workflow enactment service maintains internal control data either centralised or distributed across a set of

workflow engines; this workflow control data includes the internal state information associated with the various process and activity instances under execution and may also include checkpointing and recovery/restart information used by the workflow engines to co-ordinate and recover from failure conditions.

The process definition, in conjunction with any (run-time) workflow relevant data is used to control the navigation through the various activity steps within the process, providing information about the entry and exit criteria for individual activity steps, parallel or sequential execution options for different activities, user tasks or IT applications associated with each activity, etc. This may require access to organisation / role model data, if the process definition includes constructs relating to these entity types.

The workflow engines also include some form of application tool invocation capability to activate applications necessary to execute particular activities. The generality of such mechanisms may vary greatly, with some simple systems only offering support of a single fixed tool such as a form or document editor, whereas others may provide methods for the invocation of a wider range of tools, both local and remote to the Workflow engine.

Workflow Relevant Data and Application Data

Where process navigation decisions, or other control operations within the workflow engine, are based on data generated or updated by workflow application programs, such data is accessible to the workflow engine and termed workflow relevant data (also known as "case data"); this is the only type of application data accessible to the workflow engine. Workflow application data is manipulated directly (and only) by the invoked applications, although the workflow engines may be responsible for transferring such data between applications (if necessary), as different applications are invoked at different activity points within the workflow process.

Worklists

Where user interactions are necessary within the process execution, the workflow engine(s) place items on to worklists for attention by the worklist handler, which manages the interactions with the workflow participants. This process may be invisible to the workflow participants with the worklist maintained within the workflow software and the user being presented sequentially with the next task to be performed. On other systems the worklist may be visible to the user, who has the responsibility of selecting individual items of work from the list and progressing them independently, with the worklist being used to indicate task completions.

Worklist Handler & User Interface

The worklist handler is a software component which manages the interaction between workflow participants and the workflow enactment service. It is responsible for progressing work requiring user attention and interacts with the workflow enactment software via the worklist. In some systems, this may be little more than a desktop application providing a simple in-tray of work items awaiting user attention. In other systems this may be far more sophisticated, controlling the allocation of work amongst a set of users to provide facilities such as load balancing and work reassignment. In addition to these worklist handling functions, workflow engines typically support a wider range of interactions with client applications, including sign-on and -off of workflow participants, requesting the commencement

of an instance of particular process types, requesting workitems queued for particular participants, etc. Within the reference model the term *workflow client application* is used in preference to "worklist handler" to reflect this wider range of potential usage, which includes process control functions as well as worklist manipulation.

In the diagram the User Interface is shown as a separate software component, responsible for the look and feel of the user dialogue and control of the local interface with the user. In certain systems this may be combined with the Worklist Handler into a single functional entity. It also expected that some client applications will interact with several different workflow services, enabling workitems from such services to be consolidated into a unified task list for presentation to participants via a common user interface.

Invocation of local applications may be necessary to support the user in the particular tasks to be undertaken. This may be done by the Worklist Handler, for example at the time of presenting workitems to the user, or may be the responsibility of the user, using general facilities available at the User Interface software to load appropriate supporting applications. There is a distinction between application invocation at the Worklist Handler/User Interface (which is not directly controlled from the workflow engine and may not be visible to it) and direct application invocation by the workflow enactment software.

Supervisory Operations

Within a workflow system there are a number of supervisory functions which are normally provided; these are typically supported on the basis of supervisory privilege to a particular workstation or user(s). These functions may enable supervisors to alter work allocation rules, to identify participants for specific organisational roles within a process, to track alerts for missed deadlines or other forms of event, to trace the history of a particular process instance, to enquire about work throughput or other statistics, etc. Where distributed workflow engines are used there may need to be specific commands to transfer such control operations or (partial) responses between different workflow engines to provide a single administrative interface.

Exposed and Embedded Interfaces

Whilst the majority of workflow products can be related to the above structure, not all products offer exposed interfaces between the various individual system functional components; some products may implement several functional components together as a single logical entity with the interfaces embedded within the software component and not available for third party product use. The WFM specifications will identify, for each interface, the role of that interface in achieving interoperability, so that individual products can identify conformance against particular interoperability criteria. (For example, a particular product might offer an exposed interface for worklist manipulation but not for process definition interchange.)

2.4 Alternative Implementation Scenarios

The structural model of a generic workflow product identifies a series of software components and interfaces. In a concrete product implementation this structure may be realised in a variety of different ways; this is an important area of product differentiation. Major distinguishing factors between

products include choice of platform and network infrastructure, as well as the inherent functionality of the workflow software itself. This section illustrates how the generic model copes with this variety of implementation approach, whilst retaining visible interfaces to facilitate multi-vendor product interworking.

A full discussion of all potential implementation design issues lies outside the scope of this document. Amongst the main alternatives considered are:

- centralised or distributed workflow enactment service
- worklist handler location(s) and distribution mechanism

Workflow Enactment Software -Alternative Approaches

The workflow enactment software consists of one or more workflow engines, which are responsible for managing all, or part, of the execution of individual process instances. This may be set up as a centralised system with a single workflow engine responsible for managing all process execution or as a distributed system in which several engines cooperate, each managing part of the overall execution.

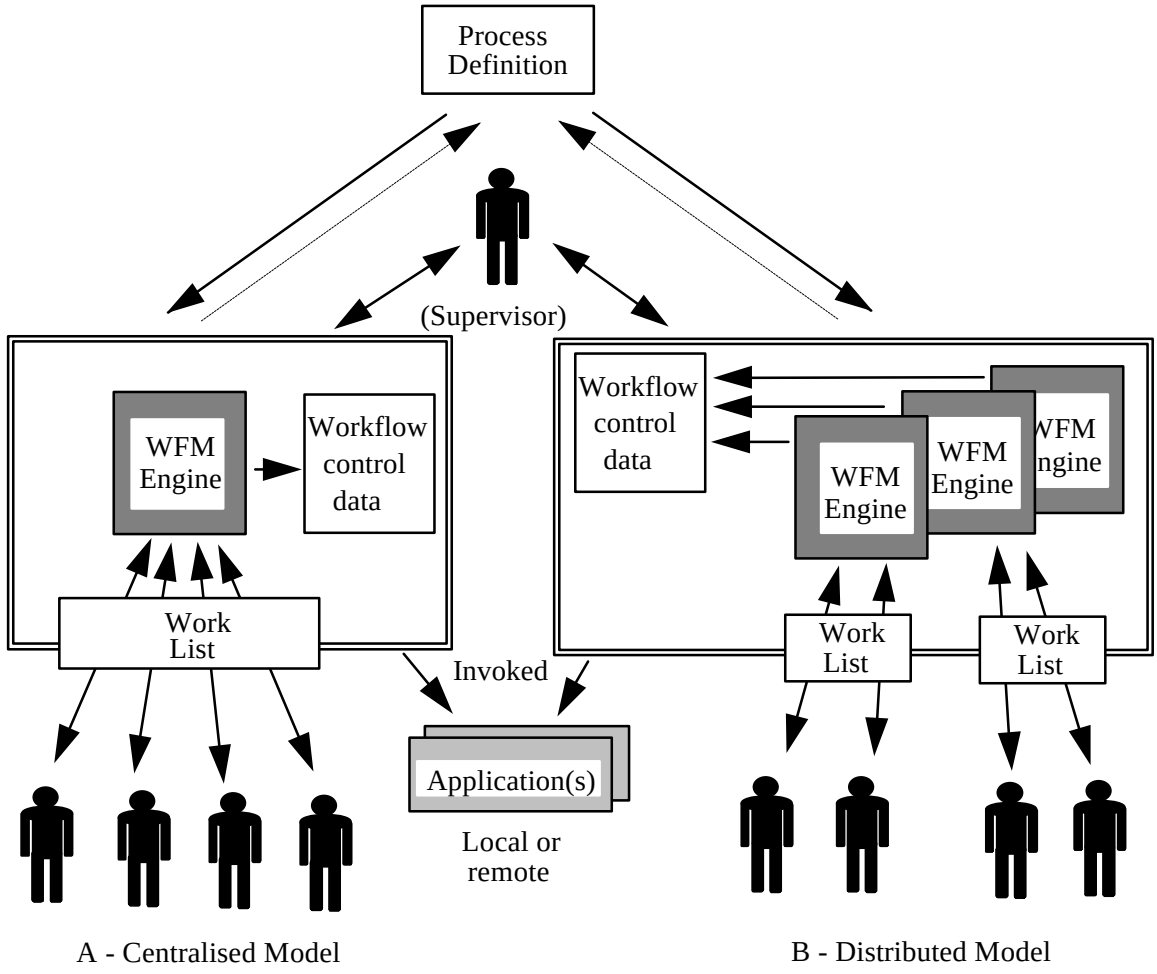

Figure 4 - Standard workflow enactment service boundary

In the above scenario the two workflow services exhibit common properties at the boundary but follow different internal implementation architectures, whose characteristics may be product dependent.

Where several workflow engines cooperate in the execution of a process instance, the control data associated with the process instance must be accessible to the different engines. This workflow control data may be distributed across the engines, located at a master engine or held as a shared filestore resource, or some combination of these. The particular implementation approaches by which this data is made available to the engines is considered to be outside the current scope for standardisation. Similarly, the process definition data may be distributed across all engines or parts transferred to individual engines from some master source during process execution. Interfaces to handle supervisory operations or application invocation may be supported as distributed features or localised to particular engines. The implementation approaches to manage distribution of workflow across multiple engines are thus complex and numerous.

The approach taken by the Coalition is to define a boundary around the workflow enactment service, which exhibits various standard functional attributes accessible via a set of common APIs. The internal mechanisms by which the enactment service delivers this capability are not defined and may include one or more homogenous workflow engines, communicating in a variety of ways.

To support interworking between different products, interfaces are defined for specific co-operative functions between different enactment services so that a composite multi-vendor workflow application may execute parts of a particular process on different enactment services (each comprising one or more specific vendors workflow engines). This is considered a more realistic approach (except perhaps in the long term) than attempting to standardise the internal interfaces and state data of a distributed workflow service.

Workflow Client Applications - Alternative Approaches

In the workflow model interaction occurs between the worklist handler and a particular workflow engine through a well defined interface embracing the concept of a worklist - the queue of work items assigned to a particular user (or, possibly, group of common users) by the workflow enactment service. At the simplest level the worklist is accessible to the workflow engine for the purposes of assigning work items and to the worklist handler (ie the workflow client application) for the purpose of retrieving work items for presentation to the user for processing.

There are various possible product implementations of this worklist interaction model depending upon the nature of the product implementation and, in particular, on the type of infrastructure used to support the distribution of worklist handling.

Four possible approaches are illustrated in the following diagram, one supporting centralised worklist handling and three using a distributed worklist handler function.

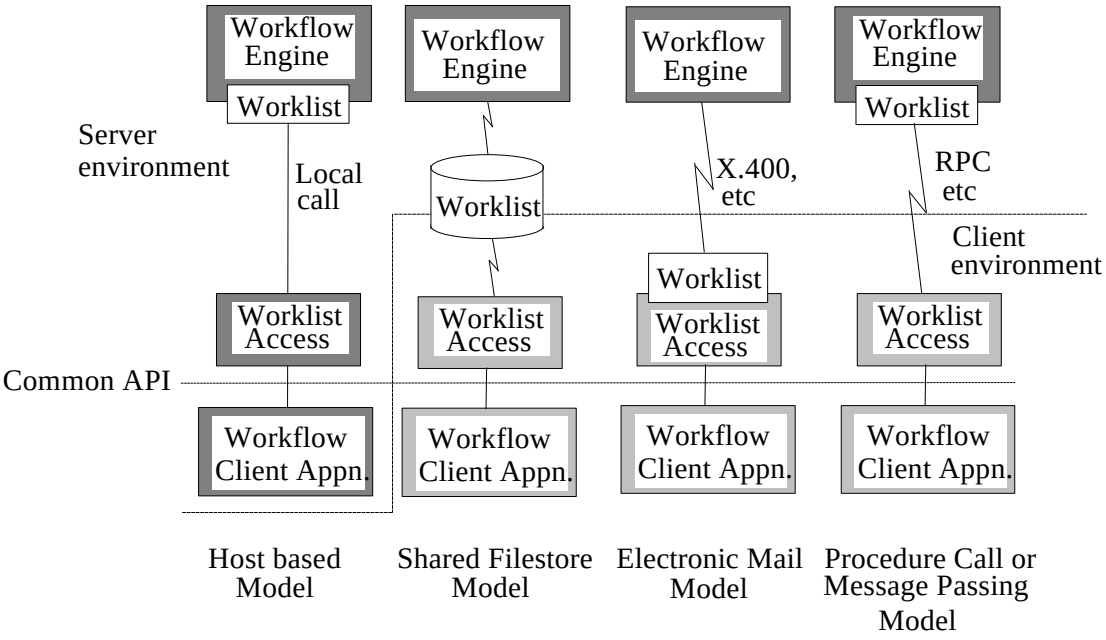

Figure 5 - Alternative client worklist handler implementations

The four example scenarios are as follows:

- Host based Model - the client worklist handler application is host based and communications with the worklist via a local interface at the workflow engine. In this case the user interface function may be driven via a terminal or a remote workstation MMI.
- Shared filestore model - the worklist handler application is implemented as a client function and communication is via a shared filestore, which lies on the boundary between host and client platform environments and is accessible to both.
- Electronic mail model - communication is via electronic mail, which supports the distribution of work items to individual participants for local processing. In this scenario the worklist would normally lie at the client.
- Procedure Call or Message Passing model - communication is via procedure call, or other message passing mechanism. In this scenario the worklist may be physically located on the workflow engine or at the worklist handler according to the particular implementation characteristics.

In each case it is feasible to construct a common API, which supports worklist handler access to the worklist and workflow engine functions, but which is located behind a specific worklist access function appropriate to the product implementation style.

2.5. The Need for Standardisation

The basic rationale to achieve standardisation of important workflow functional interfaces is driven by two major considerations:

- Ongoing support for business re-engineering & operational flexibility
- Integration requirements resulting from product specialisation and market variety

Business re-engineering & operational flexibility

The strategic importance of business process re-engineering and associated workflow implementations will lead to the requirement for sufficient flexibility of product to cope with ongoing business change, indeed this is one of the key motivations behind the use of the technology. This will include cases where several separate business processes have been implemented using different workflow products, and require to be re-engineered into a single composite process involving interaction between existing workflows. These requirements may arise due to reorganisation, legislative changes, changing business objectives, etc. As the use of electronic data interchange develops, these workflows are likely to embrace inter-organisation communications as well as those internal to a single organisation.

In these situations it is extremely likely that different products will be in use within different organisations or departments and the inability of such products to interoperate will cause a significant potential problem in coping with business change. The market projections for the penetration of workflow technology suggest very widespread adoption during the next 5-10 years, leading to the potential incompatibility problems seen in previous generations of information technology unless appropriate interworking standards are developed.

The early availability of such standards with subsequent product implementations will provide a degree of confidence to the market critical to the effective take up of workflow technology.

Specialisation and market variety

There are currently estimated to be in excess of a hundred different workflow (and related) products in the market, focussed on different aspects of functionality and data/application integration. The development of interworking standards will allow application choice of "best of breed" products for individual aspects of a workflow implementation. This may embrace process analysis and definition products from one vendor, coupled with workflow engine software from a different vendor, integrated with a client worklist handling application from a third.

An individual workflow may conveniently be broken down into several sub-processes each enacted on a specialist product suited to the specific data type, platform or network environment related to that particular sub-process. The availability of interworking standards will provide the opportunity to implement composite solutions to business process requirements, linking several such specialist products to meet the precise needs of the process.

Furthermore, many workflow applications require to integrate with other, existing or emerging applications, ranging from desktop office functions to corporate transaction processing / database. The provision of a standard interface to support this will reduce product complexity and the amount of specialist integration skills necessary during implementation.

Members of the Coalition, both vendors and users, recognise the potential importance of standards in all these areas and are co-operating in their definition.

3. Workflow Reference Model

3.1. Overview

The Workflow Reference model has been developed from the generic workflow application structure by identifying the interfaces within this structure which enable products to interoperate at a variety of levels. All workflow systems contain a number of generic components which interact in a defined set of ways; different products will typically exhibit different levels of capability within each of these generic components. To achieve interoperability between workflow products a standardised set of interfaces and data interchange formats between such components is necessary. A number of distinct interoperability scenarios can then be constructed by reference to such interfaces, identifying different levels of functional conformance as appropriate to the range of products in the market.

3.2. The Workflow Model

Figure 6 illustrates the major components and interfaces within the workflow architecture.

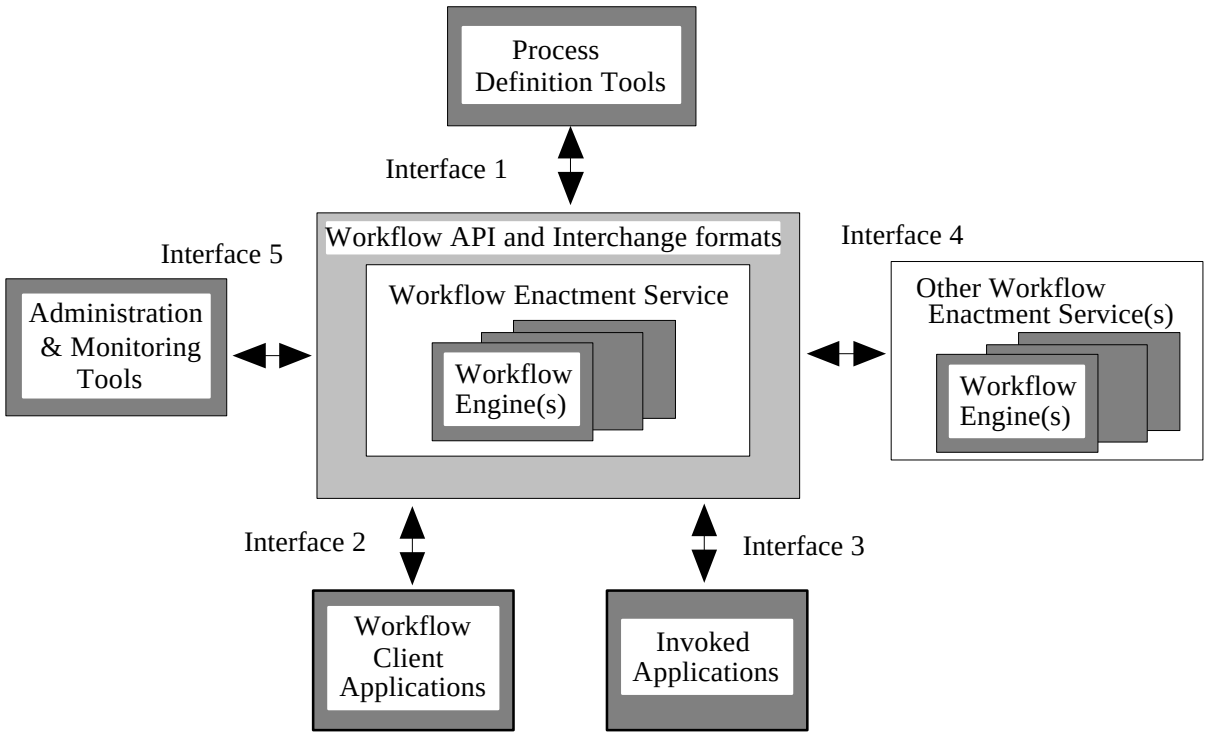

Fig 6 Workflow Reference Model - Components & Interfaces

The architecture identifies the major components and interfaces. These are considered in turn in the following sections. As far as possible, the detail of the individual interfaces (APIs and interchange formats) will be developed as a common core set using additional parameters as necessary to cope with individual requirements of particular interfaces.

The interface around the workflow enactment service is designated WAPI - Workflow APIs and Interchange formats, which may be considered as a set of constructs by which the services of the

workflow system may be accessed and which regulate the interactions between the workflow control software and other system components. Many of the functions within the 5 interface areas are common to two or more interface services hence it is more appropriate to consider WAPI as a unified service interface which is used to support workflow management functions across the 5 functional areas, rather than 5 individual interfaces.

3.3. Workflow Enactment Services

3.3.1. What is a Workflow Enactment Service?

The workflow enactment service provides the run-time environment in which process instantiation and activation occurs, utilising one or more workflow management engines, responsible for interpreting and activating part, or all, of the process definition and interacting with the external resources necessary to process the various activities.

> Definition - Workflow Enactment Service
>
> A software service that may consist of one or more workflow engines in order to create, manage and execute workflow instances. Applications may interface to this service via the workflow application programming interface (WAPI).

In the model adopted, there is a logical separation between this process and activity control logic, which constitutes the workflow enactment service, and the application tools and end user tasks which constitute the processing associated with each activity. This separation provides the opportunity for a wide range of industry standard or user specific application tools to be integrated within a particular workflow application.

Interaction with external resources accessible to the particular enactment service occurs via one of two interfaces:

- The client application interface, through which a workflow engine interacts with a worklist handler, responsible for organising work on behalf of a user resource. It is the responsibility of the worklist handler to select and progress individual work items from the work list. Activation of application tools may be under the control of the worklist handler or the end-user.
- The invoked application interface, which enabl es the workflow engine to directly activate a specific tool to undertake a particular activity. This would typically be a server-based application with no user interface; where a particular activity uses a tool which requires end-user interaction it would normally be invoked via the worklist interface to provide more flexibility for user task scheduling. By using a standard interface for tool invocation, future application tools may be workflow enabled in a standardised manner.

These interfaces are described in sections 3.5 and 3.6 respectively.

Within this section, the workflow enactment service has been discussed as a single logical entity, although physically it may be either centralised or functionally distributed.

In a distributed workflow enactment service, several Workflow engines each control a part of the process enactment and interact with that subset of users and application tools related to the activities

within the process for which they are responsible. Such an enactment service is considered to have common naming and administrative scope, so that process definitions (or subsets) and user/application names may be handled on a consistent basis. Distributed workflow systems make use of specific protocols and interchange formats between Workflow engines to synchronise their operations and exchange process and activity control information. Workflow relevant data may also be transferred between Workflow engines. Within a single homogeneous workflow enactment service, such operations are vendor specific.

Where heterogeneous products are involved, a standardised interchange is necessary between workflow engines. Using interface 4, the enactment service may transfer activities or sub-processes to another (heterogeneous) enactment service for execution. Within the Workflow Reference Model this is termed Workflow Engine Interchange and is considered under section 3.7.
Common administration and monitoring functions may also be required in such a heterogeneous environment; these are considered in section 3.8.

3.3.2. The Workflow Engine

A workflow engine is responsible for part (or all) of the runtime control environment within an enactment service.

Definition - Workflow Engine
A software service or "engine" that provides the run time execution environment for a workflow instance.

Typically such software provides facilities to handle:

- interpretation of the process definition
- control of process instances - creation, activation, suspension, termination, etc
- navigation between process activities, which may involve sequential or parallel operations, deadline scheduling, interpretation of workflow relevant data, etc
- sign-on and sign-off of specific participants
- identification of workitems for user attention and an interface to support user interactions
- maintenance of workflow control data and workflow relevant data, passing workflow relevant data to/from applications or users
- an interface to invoke external applications and link any workflow relevant data
- supervisory actions for control, administration and audit purposes

A workflow engine can control the execution of a set of process, or sub-process, instances with a defined scope - determined by the range of object types, and their attributes, which it can interpret within the process definition(s).

In an enactment service consisting of multiple workflow engines, there is a partitioning of process execution across the constituent engines. This may be by process type, with a particular engine controlling a particular process type in its entirety, by functional distribution, with a particular engine

controlling those parts of a process requiring user or resource allocation within its own control domain, or some other partitioning mechanism.

3.3.3. Homogeneous & Heterogeneous Workflow Enactment Services

An homogeneous workflow enactment service comprises one or more compatible workflow engines which provide the runtime execution environment for workflow processes with a defined set of (product specific) process definition attributes. The mechanisms by which process execution is organised across the various workflow engines and protocols and interchange formats used to support this are product specific and not standardised.

A heterogeneous workflow enactment service comprises two or more homogeneous services, which follow common standards for interoperability at a defined conformance level. It is envisaged that a number of conformance levels will be defined to support increasing levels of common functionality.

These are expected to include (amongst other things):

- A common naming scheme across the heterogeneous domain
- Support for common process definition objects and attributes across the domain
- Support for workflow relevant data transfer across the domain
- Support for process, sub-process or activity transfer between heterogeneous workflow engines
- Support for common administration and monitoring functions within the domain

Support for common workflow control data and its interchange (eg shared process and activity state data) would be necessary to support totally open interworking between heterogeneous products; whilst an interesting standardisation challenge it is considered unattainable in the foreseeable future, hence the emphasis on levels of interoperability governed by defined conformance criteria.

Process and Activity State Transitions

The workflow enactment service may be considered as a state transition machine, where individual process or activity instances change states in response to external events (eg completion of an activity) or to specific control decisions taken by a workflow engine (eg navigation to the next activity step within a process).

A basic state transition scheme for process instances is shown below.

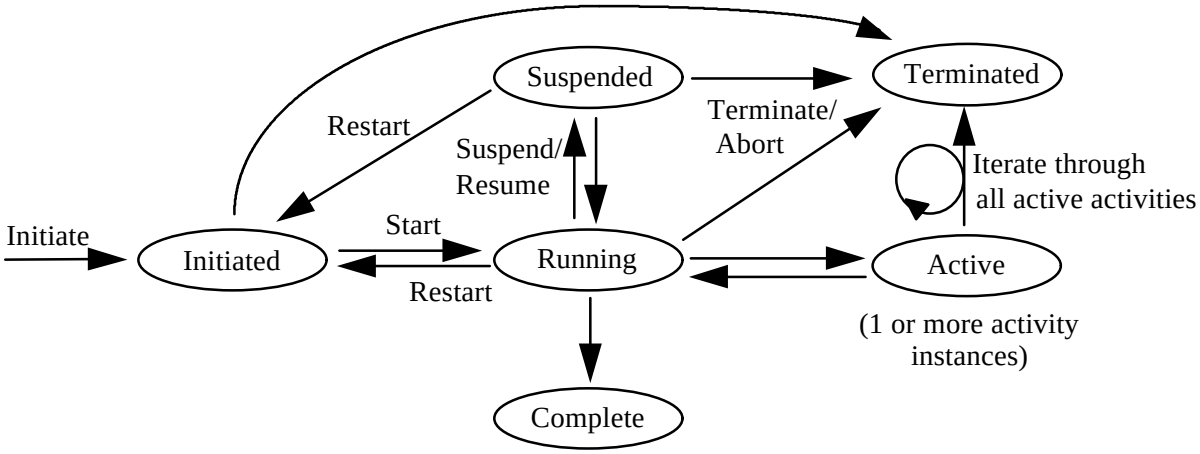

Figure 7 - State transitions for a process instances

Within the above diagram, transition between states (represented by the arrows) take place in response to the particular WAPI commands identified; transition between certain states will also take place as a result of transition conditions within the process definition being met (eg as the result of an external event, or time or data dependent condition, etc). The basic states are:

created - a process instance has been created, including any associated process state date and workflow relevant data, but the process has not (yet) fulfilled the conditions to cause it to start execution

running - the process instance has started execution and any of its activities may be started (once any appropriate activity start conditions have been met)

active - one or more of its activities has been started (ie a workitem has been created and assigned to an appropriate activity instance)

suspended - the process instance is quiescent and no activities are started until the process has returned to the running state (via a resume command)

completed - the process instance has fulfilled the conditions for completion; any internal post-completion operations such as logging audit data or statistics will be performed and the process instance destroyed

terminated - execution of the process instance has been stopped before its normal completion; any internal operations such as error logging or logging recovery data may be performed and the process instance destroyed

Activities may be non-interruptable; ie once a workflow service has started a particular activity within a process instance, it may not be possible to suspend or terminate that activity. This means that suspension / restart / terminate functions cannot be completed until all active activities have completed and the process instance returned to a running state. In addition, it may be required to mark a set of activities as an atomic unit, which are either executed in entirety or the process instance "rolled-back" to a restart point. The potential treatment of interruptable activities and atomic activity units with restart capability will require further consideration and is beyond the initial work of the Coalition.

Ignoring these additional complexities, a simple illustration of the basic states and transitions for an activity instance is thus:

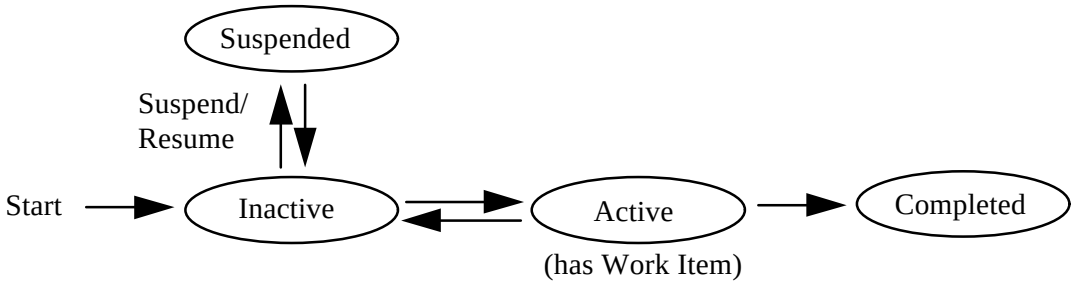

Figure 8 - State transitions for activity instances

The basic states of an activity instance are:

inactive - the activity within the process instance has been created but has not yet been activated (eg because activity entry conditions have not been met) and has no workitem for processing

active - a workitem has been created and assigned to the activity instance for processing

suspended - the activity instance is quiescent (eg as a result of a change_state_of_activity_instance command) and will not be allocated a workitem until returned to the running (inactive) state

completed - execution of the activity instance has completed (and any post-activity transition conditions will be evaluated)

A particular product implementation may, of course, support additional state types or use a different representation of the basic transition diagrams shown above. The reference model does not attempt to prescribe standardised internal behaviour of workflow systems but the state transitions represent the basic underlying concepts which are necessary to scope the effects of the API command set which the Coalition is developing.

3.3.4. Workflow Application Programming Interface & Interchange

The WAPI may be regarded as a set of API calls and interchange functions supported by a workflow enactment service at its boundary for interaction with other resources and applications. Although this architecture refers to 5 "interfaces" within WAPI, a number of the functions within each of these interfaces are common (for example process status calls may be issued from the client application interface or the administration interface). The WAPI is thus being defined as a common core of API calls /interchange formats with specific extensions where necessary to cater individually for each of the five functional areas.

The majority of WAPI functions comprises of APIs calls with defined parameter sets / results codes. Where appropriate it also defines interchange data formats, for example for the exchange of process definitions. The use of WAPI within each of the five functional areas is described within the following sections (3.4 - 3.8).

3.3.5. Workflow Control, Workflow Relevant and Workflow Applications Data

The workflow enactment service maintains internal control data to identify the state of individual process or activity instances and may support other internal status information. This data is not accessible or interchangeable, as such, via the WAPI commands, but some of the information content

may be provided in response to specific commands (eg query process status, give performance metrics, etc). Homogeneous workflow enactment services may exchange such information between workflow engines by specific private dialogue.

Definition - Workflow Control Data

Internal data that is managed by the workflow management system and/or workflow engine.

Workflow Relevant Data is used by a workflow management system to determine particular transition conditions and may affect the choice of the next activity to be executed. Such data is potentially accessible to workflow applications for operations on the data and thus may need to be transferred between activities by the workflow enactment software. When operating in a heterogeneous environment, such data may need to be transferred between workflow engines, where the process execution sequence spans two or more workflow engines; this process may (potentially) require name mapping or data conversion.

Definition - Workflow Relevant Data

Data that is used by a workflow management system to determine the state transition of a workflow process instance.

Manipulation of application data may be required within each activity of a process definition, for example by a particular tool or application, either under the direct control of the application or in conjunction with some form of user interaction. The workflow model must, therefore, cope with any necessary interchange of case data between the various activities. In some circumstances this may also require some form of case data transformation between different tool data formats, for example conversion of a document or spreadsheet from one application format to another. (In some systems this may be a function of the workflow enactment service, in others data conversion may be defined as an activity in its own right within the process definition.)

Definition - Workflow Application Data

Data that is application specific and not accessible by the workflow management system.

Workflow application data is not used by the workflow enactment software and is relevant only to the applications or user tasks executed during the workflow. As with workflow relevant data, it may need to be transferred (and/or transformed) between workflow engines in a heterogeneous enactment service, so as to be made available to the appropriate activities executed on the individual engines.

The relationship between an application and any workflow relevant or application data it needs to manipulate will normally be defined within the process definition. In some cases this may be an implicit relationship (for example in those systems where case data is physically transferred to the next activity as part of the activity navigation within the process), whereas in others (for example access to a shared object store) it may be an explicit relationship defining a specific object name and application access path. Within the reference model the former scenario will be called direct data interchange and the latter indirect data interchange.

3.3.6. Data Interchange

Interchange of workflow relevant and application data is (potentially) required across the WAPI to support interworking within three runtime functions

- worklist handler (interface 2)
- invoked application (interface 3)
- workflow engine interchange (interface 4)

This section covers the general principles of data interchange; this area will require further specification work. The proposed API command set may include specific calls to accept/return workflow relevant data from/to the enactment service across the WAPI; variants of these could be defined for both direct and indirect case data interchange.

The direct interchange of application data is typified by email driven workflow systems in which the data is physically transferred between activities, either application or user-driven. In this situation there is no need to define an explicit relationship between activities and application data; the data is transferred as part of the standard workflow activity navigation and locally linked to the application on invocation. Where there is a requirement to provide data format conversion between activities, the model recognises that a particular application may define, as an attribute, the data type (or types) with which it is associated (this attribute information may be held local to a particular software environment or global to the entire workflow service - for example in a directory). This enables systems which are constructed to use heterogeneous workflow applications to provide data conversion (where necessary) on the basis of attribute types defined for the respective applications. Conventions will need to be adopted (or developed) for transferring and retaining the data type information, for example by the use of X.400 body part object identifiers or the Internet mail MIME mechanism (RFC-1341).

Some types of workflow system (for example, those implemented via a shared document store) do not physically transfer application data between activities. In these systems, data is accessed in situ by the application using an appropriate access path (which may be networked). In this case, the access path naming scheme must be global to all applications which may be invoked within the workflow service and appropriate access permissions must be available and controlled for each active process instance. Data format conversion in this scenario, if necessary, may be modelled as an activity in its own right, using an appropriate application tool (for example a document converter).

Homogenous systems may use private conventions for object names and access permissions, but heterogeneous systems require a common scheme. In this case, either the (common) process definition must include access path references to the application data object storage, or the navigation between activities must include transfer of the necessary access path references for any data objects to be transferred between activities.

Where interworking between heterogeneous workflow products is planned they must either follow the same approach to application data interchange or interwork through a gateway mechanism (section 3.7), which can map between the two approaches and/or handle any differences in object naming and data type conventions by appropriate conversion. Further work is required on the detail in this area, but it is possible that alternative interchange conformance criteria could be identified to cover the two cases.

The way in which application or workflow relevant data interchange is to be handled across the 3 interfaces is for more detailed study; the following notes identify some initial options.

Client applications - workflow relevant data may be embedded in the workitem and extracted from the worklist for presentation to the user or for linkage to a particular application tool (for example by the worklist handler locating it in a particular local directory). Alternatively, the data may be indirectly passed to a specific application via some form of shared object store (for example by the use of a common file for data in transit between applications, or by passing a specific file reference embedded as part of the workitem.)

Invoked applications - the data interchange will depend upon the nature of the application invocation interface (section 3.6) and may require the invocation service to embed the data within a specific application protocol. APIs for reading /writing workflow relevant data are feasible for specific workflow-enabled applications or to construct generalised application agents.

Workflow engine interoperability - considerations are similar to the Client Application interface, although where the different systems support different application data interchange approaches, the use of a gateway function will be necessary to map between the two schemes and, possibly, handle name resolution.

3.4. Process Definition

3.4.1. Process Definition Tools

A variety of different tools may be used to analyse, model, describe and document a business process; such tools may vary from the informal ("pencil and paper") to sophisticated and highly formalised. The workflow model is not concerned with the particular nature of such tools nor how they interact during the build-time process. As noted earlier, such tools may be supplied as part of a workflow product or as a separate, for example, BPR product toolset.

Where a workflow product provides its own process definition tool, the resultant process definitions will normally be held within the workflow product domain and may, or may not, be accessible via a programming interface for reading and writing information. Where separate products are used for defining and executing the process, the process definitions may be transferred between the products as and when required or may be stored in a separate repository, accessible to both products (and possibly other development tools).

The final output from this process modelling and design activity is a process definition which can be interpreted at runtime by the workflow engine(s) within the enactment service. For today's workflow products each individual process definition is typically in a form specialised to the particular workflow management software for which it was designed. The workflow definition interchange interface will enable more flexibility in this area.

The process analysis, modelling and definition tools may include the ability to model processes in the context of an organisation structure (although this is not a mandatory aspect of the workflow reference model). Where an organisation model is incorporated into such tools the process definition will include

organisation related objects such as roles. These are related (typically) to system control data such as role: actor relationships (eg within an organisational directory) which may be referenced during process execution.

3.4.2. Workflow Definition Interchange (Interface 1)

The interface between the modelling and definition tools and the runtime workflow management software is termed the process definition import/export interface. The nature of the interface is an interchange format and API calls, which can support the exchange of process definition information over a variety of physical or electronic interchange media. The interface may support the exchange of a complete process definition or a subset - for example a set of process definition changes or the attributes of a particular activity within the process definition.

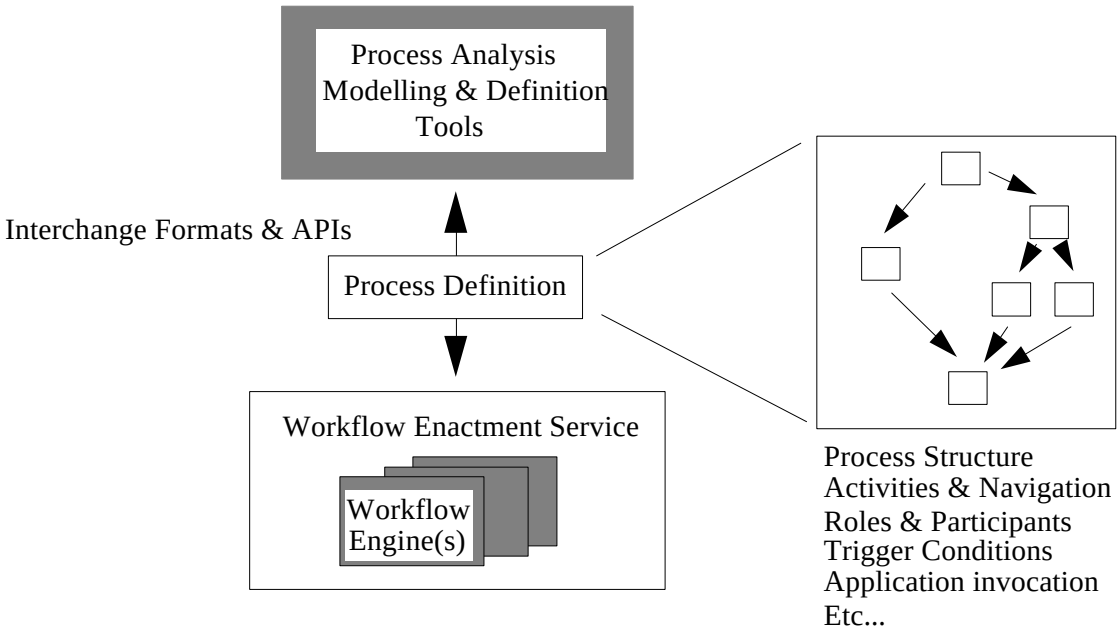

Fig 9 - Process Definition Interchange

There are clear benefits in using a standardised form for this definition.

Firstly, it defines a point of separation between the build-time and runtime environments, enabling a process definition generated by one modelling tool to be used as input to a number of different workflow runtime products. This enables user choice of modelling tools and workflow runtime products to be independent.

Secondly, it offers the potential to export a process definition to several different workflow products which could co-operate to provide a distributed runtime enactment service. (The ability to exchange process definition data is only one aspect of such a distributed service; there are other requirements in terms of runtime interactions between WFM-Engine, which are considered in section 3.8.)

There are two aspects to the Coalition's work in this area:

1. derivation of a meta-model which can be used to express the objects, their relationships and attributes within a process definition and which can form the basis for a set of interchange formats to exchange this information between products
2. API calls (within the WAPI) between workflow systems or between a workflow system and process definition product, providing a common way to access workflow process definitions. Access may be read, read/write or write only and may manipulate the set of standard objects defined within the meta-model or a product-specific set (for example defined in a product type register).

A Basic Meta-Model

The Coalition is developing a meta-model for the process definition, which identifies a basic set of object types appropriate to an initial level for the interchange of relatively simple process definitions. Further object types may be added, either by vendor specific extensions and/or by defining additional conformance levels with added functionality.

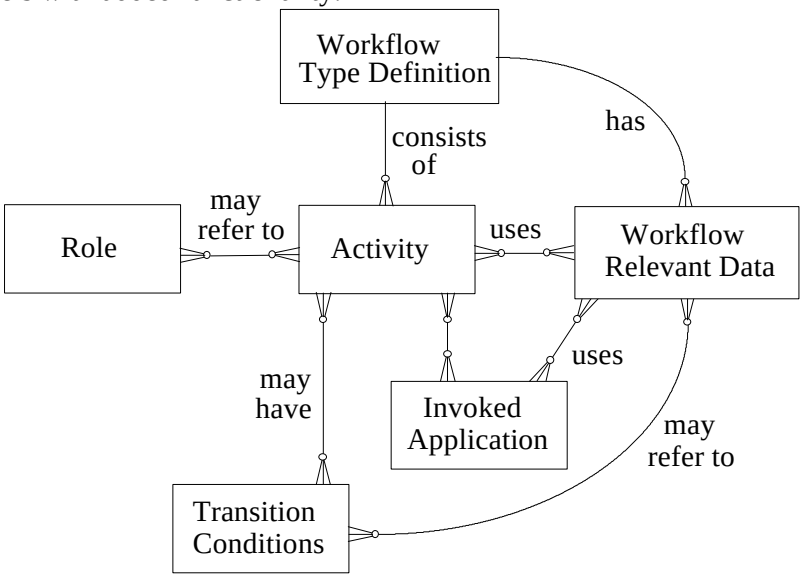

Fig 10 - Basic Process Definition Meta-model

It is envisaged that particular attributes of the following types will be defined:

Workflow Type Definition

- Workflow process name
- Version number
- Process start and termination conditions
- Security, audit or other control data

Activity

- Activity name
- Activity type (subflow, atomic flow, etc)
- Pre- and post- activity conditions

- Other scheduling constraints

Transition Conditions

- Flow or Execution conditions

Workflow relevant data

- Data name and path
- Data types

Role

- Name and organisational entity

Invoked Application

- Generic type or name
- Execution parameters
- Location or access path

In the case of distributed services, an allocation of activities to individual Workflow engines may also need to be made within the process definition, as an additional activity attribute. Process definition aspects affecting security and administration, for example controls over privileged or supervisory activities within the process, also require consideration in the longer term.

In defining interchange formats, it is assumed that a symbolic naming scheme would be supported which could be unambiguously mapped to real names and addresses in the runtime enactment service. This may be handled by dynamic address resolution mechanisms (for example by the use of a directory service) or by other mechanism external to the process definition. There are other industry groups working in related areas such as process modelling and CASE interchange tools; the proposed WFM Coalition approach in this area is to work with other groups to advance the definition of suitable interchange formats.

APIs to access Process Definitions

A set of API commands within WAPI is under development to support access to process definition data. It is expected that such specifications will cover a number of functions of the following general types. Commands are expected to be provided which operate on a list, or on individual objects or attributes.

Session Establishment

- connection / disconnection of sessions between participating systems

Workflow Definition Operations

- retrieval of lists of workflow process definition names from a repository or other source list.

- selection / de-selection of a workflow process definition to provide a session handle for further object level operations
- read/write top level workflowprocess definition object

Workflow Definition Object Operations

- creation, retrieval & deletion of objects within a workflow definition
- retrieval, setting and deletion of object attributes

3.5. Workflow Client Functions

3.5.1. Workflow Client Applications

The worklist handler is the software entity which interacts with the end-user in those activities which require involve human resources. The worklist handler may be supplied as part of a workflow management product or may be written by a user, for example to provide a particular common house style for use with a number of different workflow applications utilising different vendor's products. In other cases, workflow may be integrated into a common desktop environment alongside other office services such as mail and work-in-progress folders to provide a unified task management system for the end-user. There is thus a need for a flexible mechanism of communication between a workflow enactment service and workflow client applications to support the construction of the many different operational systems which are expected to be encountered.

In the workflow model interaction occurs between the client application and the workflow engine through a well defined interface embracing the concept of a worklist - the queue of work items assigned to a particular user (or, possibly, group of common users) by the workflow engine. At the simplest level the worklist is accessible to the workflow engine for the purposes of assigning work items and to the worklist handler for the purpose of retrieving work items for presentation to the user for processing. There are various possible product implementations of this worklist interaction (see section 2.4).

Activation of individual work items from the worklist (for example launching application and linking workflow relevant data) may be under the control of the workflow client application or the end-user. A range of procedures is defined between the workflow client application and the workflow enactment service to enable new items to be added to the worklist, completed activities to be removed from the worklist, activities to be temporarily suspended, etc. These are described in section 3.5.2
Application invocation may also be handled from the worklist handler, either directly or under the control of the end-user. In general it is expected that the range of applications invoked from the worklist handler would be predominantly local to that environment, although it may place an unnecessary constraint on the generality of the model to assume that this will always be the case.

Part of the activity related data associated with the worklist is the necessary information to enable the worklist handler to invoke the appropriate applications(s). Where the application data is strongly typed, an association may be stored at the worklist handler and used for this purpose. In other cases an interchange of the full application name and address information may be necessary between the worklist handler and Workflow engine, in which case the workflow Client Application may also implements some functions from the invoked application interface (i/f 3) to obtain the necessary information.

A worklist may contain items relating to several different active instances of a single process and/or individual items from activations of several different processes. A worklist handler might potentially be interacting with several different Workflow engines and several different enactment services. (According to individual product implementation, separate physical worklists may be maintained for each process type, or the worklist handler may consolidate the various worklists items into a single representation to the end-user.) .

The interface between the client workflow application and Workflow engine must therefore be sufficiently flexible in terms of its use of:

- process and activity identifiers
- resource names and addresses
- data references and data structures
- alternative communications mechanisms

to contain these variations of implementation approach.

3.5.2. Workflow Client Application Interface (Interface 2)

The approach to meet the above requirement is to contain the variety behind a standard set of APIs (the WAPI), which may be used in a consistent manner for access from a workflow application to the Workflow engine and worklist, irrespective of the nature of actual product implementation.

The APIs and its parameters will be mapped onto several alternative communications mechanisms to fit the variety of workflow implementation models. (In the case of email based communications it is also possible, of course, for a worklist handler to directly access the incoming mailbox for incoming work items using any local mailbox access interface, rather than via specific WAPI calls. In this case the worklist handler application will take responsibility for filtering any non-workflow email items and handling them in an appropriate manner. Similarly commands or responses directed at the Workflow engine by the workflow application may be submitted directly to an outgoing mailbox handler. In this scenario a simple level of interoperability is achieved through the use of standardised mail interchange formats, rather than the full WAPI.)

The overall approach to the client application API is shown in figure 11, following.

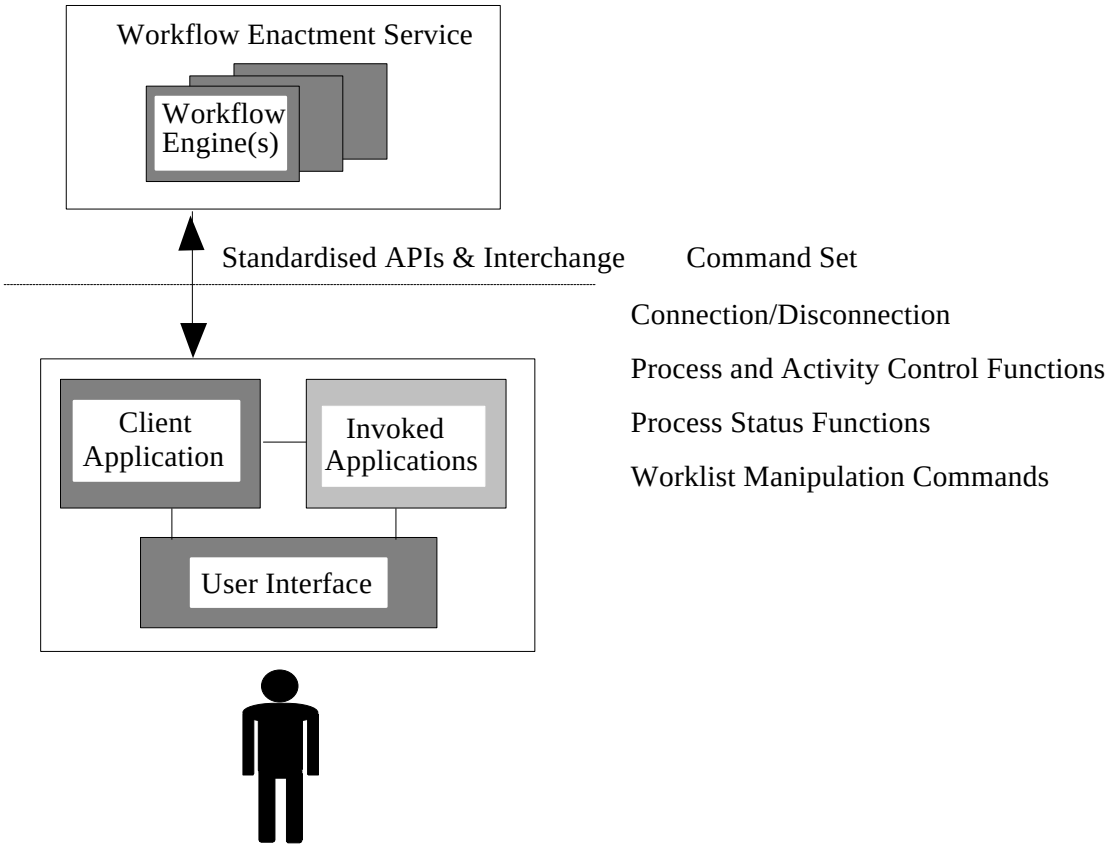

Figure 11 - Client Application Interface

The API specifications are published in a separate Coalition document; the following provides an overview of the intended APIs for client application use, grouped into various functional areas. Commands are provided for operations on individual or collective process or activity instances as well as worklist manipulation.

Session Establishment

- connection / disconnection of sessions between participating systems

Workflow Definition Operations

- retrieval / query functions (with optional selection criteria) on workflow process definition names or attributes

Process Control Functions

- creation / starting / termination of an individual process instance
- suspension / resumption of an individual process instance
- forcing a state change within an individual process instance or activity instance
- assignment or query of an attribute (eg priority) of a process or activity instance

Process Status Functions

- Opening / closing a process or activity instances query, setting optional filter criteria
- Fetching details of process instances or activity instances, filtered as specified
- Fetching details of a specific (individual) process or activity instance

Worklist/Workitem Handling Functions

- Opening / closing a worklist query, setting optional filter criteria
- Fetching worklist items, filtered as specified
- Notification of selection / reassignment / completion of a (specific) workitem
- Assignment or query of a workitem attribute

Process Supervisory Functions

(The following functions operate on all process or activity instances and are deemed to operate in the context of a supervisory privilege level, which may, or may not, be granted to a specific client application or user logged onto such application.)

- changing the operational status of a workflow process definition and/or its extant process instances
- changing the state of all process or activity instances of a specified type
- assigning attribute(s) to all process or activity instances of a specified type
- termination of all process instances

Data Handling Functions

- retrieval / return of workflow relevant or application data

Administration Functions

Support for additional administration functions across the WAPI may be appropriate for certain client applications. A subset of the operations discussed in 3.8.2 may be included in a future conformance level.

Application Invocation

The functions outlined above provide a base level of functionality to support application invocation by the worklist handler function (eg by providing access to process/activity/workitem attributes and workflow relevant data). Some of the proposed commands for the application invocation function (section 3.6.2) may also be relevant to the client application environment.

It is possible that some product implementations may wish to support a subset of the full WAPI; further consideration will be given to identifying conformance levels to cater for the different interoperability requirements arising from the range of workflow products available in the market.

3.6. Invoked Application Functions

3.6.1. Invoked Applications

It may be assumed that any particular WFM implementation will not have sufficient logic to understand how to invoke all potential applications which might exist in an heterogeneous product environment. This would require the logic to cope with invocation across (potentially) all platform and network environments, together with a means of transferring application or workflow relevant data in a common format and encoding (or transforming it to the individual application environments).

However there are many workflow systems which deal with a more restrictive range of applications, particularly those where the data is strongly typed and may be directly associated (for example via a directory) with a particular application tool such as a word processor or spreadsheet. In other cases invocation of an operation by a particular application may be accomplished through a standard interchange mechanism such as the OSI TP protocol or X.400. Some implementations use the concept of a "Application Agent" to contain this variety of method invocation behind a standard interface into the workflow enactment service. There is also the possibility of developing "workflow enabled" application tools which use a standard set of APIs to communicate with the workflow enactment service - to accept application data, signal and respond to activity events, etc. Such APIs may be used directly by an application tool or by a application agent process acting as a front end for interaction with heritage or other applications written without a specific knowledge of workflow.

Some of the possible types of interface for application invocation are identified in the following table.

Interface Type	Workflow Relevant Data Access	Standardisation Candidate
Local Process Call	Local File	No
Shell Script	Local File	POSIX environments?
ORB Call (eg object linking and launch service)	Via reference (call parameters)	Yes
Remote Execution Call	Via reference (call parameters)	Yes
Message Passing (eg X.400)	Embedded or via reference	Yes
Transaction (eg OSI-TP)	Embedded or via reference	Yes

Table 1 - Application Invocation Interfaces

Further discussion will be required on the full range of possible options for application invocation. The initial work of the Coalition is likely to focus on developing a catalogue of interface types, together with a set of APIs for use in future workflow specific applications.

3.6.2. Invoked Applications Interface (Interface 3)

The diagram following shows the scope of this interface, which is intended to be applicable to application agents and (longer term) applications which have been designed to be "workflow enabled" (ie to interact directly with a workflow engine).

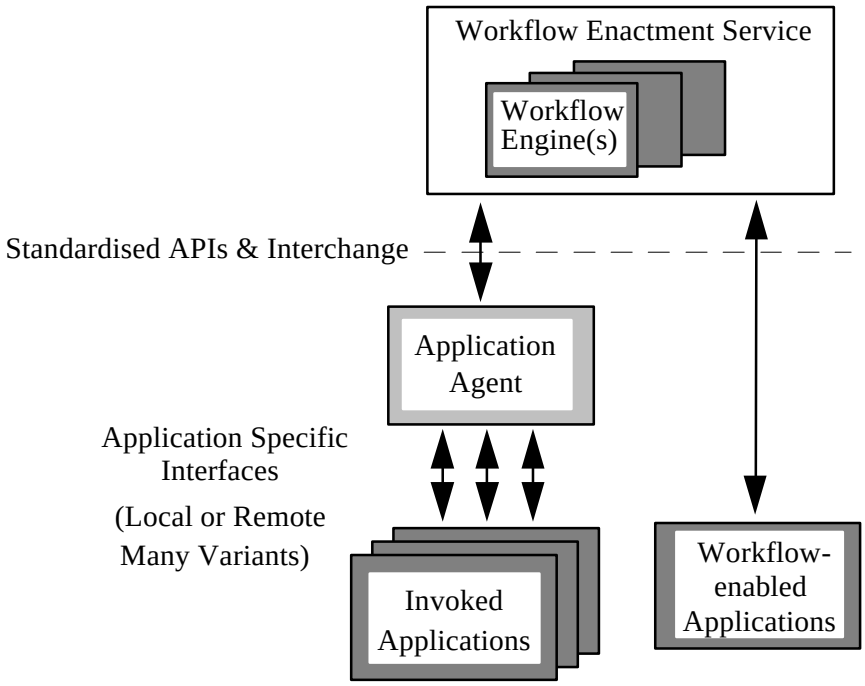

Figure 12 - Invoked Application Interface

In the simple case, application invocation is handled locally to a workflow engine, using information within the process definition to identify the nature of the activity, the type of application to be invoked and any data requirements. The invoked application may be local to the workflow engine, co-resident on the same platform or located on a separate, network accessible platform; the process definition contains sufficient application type and addressing information (specific to the needs of the workflow engine) to invoke the application. In this case the conventions for application naming and addressing are local between the process definition and the workflow engine.

The detailed semantics and syntax of an API set for application invocation are for further study and will be documented as part of the Coalition specification set. Operation is envisaged over a variety of underlying interfaces, including a selection from the above table, some of which may operate synchronously and others asynchronously. The operation of the API is assumed at this stage to be potentially either single- or multi-threaded (in the latter case using an activity id handle for thread discrimination). The following provides an outline of a possible command set applicable to application invocation functions.

Session Establishment

- connection / disconnection of application (or application agent) session

Activity Management Functions

(workflow engine to application)

- Start activity (workflow engine to application)
- Suspend/Resume/Abort activity (where an asynchronous application interface is available)

(application to workflow engine)

- Activity complete notification
- Signal event (eg synchronisation)
- Query activity attributes

Data Handling Functions

- Give workflow relevant data (pre-activity to application, post activity from application)
- Give application data or data address

More complex scenarios, involving interworking between heterogeneous Workflow engines, may require application invocation information to be transferred between Workflow engines, either as part of the run-time interchange or by importing (parts) of the process definition after the process development phase. This is considered under section 3.7 (workflow interoperability).

3.7. Workflow Interoperability

3.7.1. Heterogeneous Workflow Enactmnent Services

A key objective of the coalition is to define standards that will allow workflow systems produced by different vendors to pass work items seamlessly between one another.

Workflow products are diverse in nature ranging from those used for more ad-hoc routing of tasks or data to those aimed at highly regularised production processes. In its drive for interoperability standards the Coalition is determined not to force workflow product vendors to choose between providing a strong product focused on the needs of its customers and giving up those strengths just to provide interoperability.

The work of the Coalition has therefore focussed on developing a variety of interoperability scenarios which can operate at a number of levels from simple task passing through to full workflow application interoperability with complete interchange of process definition, workflow relevant data and a common look and feel. In this area it is expected that relatively simple interoperability scenarios will be supported initially, with the more complex situations requiring further work on interoperability definitions.

Although it is possible to consider the development of very complex interoperability scenarios in which a number of different vendor engines cooperate to deliver a single enactment service, this scenario is unlikely to be realised in the near future as it requires that all engines can interpret a common process

definition and share a common set of workflow control data, in effect maintaining a shared view of process states across the heterogeneous workflow control engines. A more realistic target in the near term is the ability to transfer parts of a process for runtime support on a different enactment service.

Four possible interoperability models has been identified, covering various (increasing) levels of capability. The following sections describes these potential interoperability models; illustrations show circles to indicate tasks or activities co-ordinated by one server and squares to indicate tasks co-ordinated by another server.

3.7.2 Scenario 1 - Connected Discrete (Chained)

This model allows a connection point within process A to connect to another point within process B. Although the illustration shows these connection points at the terminus and starting points of the processes, this is done for illustration purposes only. It is presumed that the connection points can be anywhere within the processes that makes sense for the meat-process created by the connection of the two.

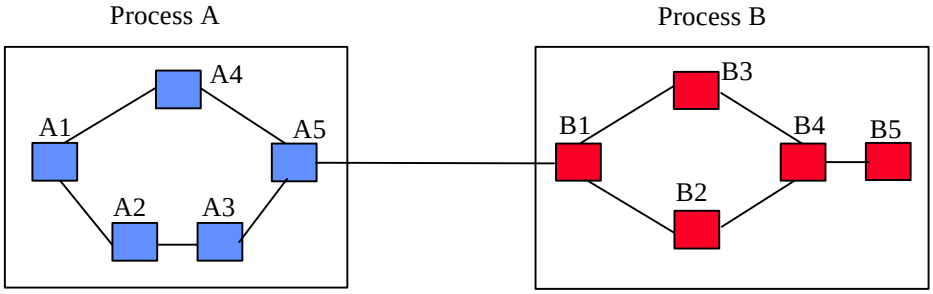

Figure 13 - Chained Services Model

This model supports the transfer of a single item of work (a process instance or activity) between the two workflow environments, which then operates independently in the second environment with no further synchronisation. In implementation terms it may be realised via a gateway application function, handling data format conversion, process and activity name mapping, etc, or may be subsumed into one of the workflow services, for example when a standard API call is used between the two services.

3.7.3 Scenario 2 - Hierarchical (Nested Subprocesses)

This allows a process executed in a particular workflow domain to be completely encapsulated as a single task within a (superior) process executed in a different workflow domain. A hierarchic relationship exists between the superior process and the encapsulated process, which in effect forms a sub-process of the superior. The hierarchic relationship may be continued across several levels, forming a set of nested sub-processes. Recursion within this scenario may, or may not, be permitted by individual product implementations.

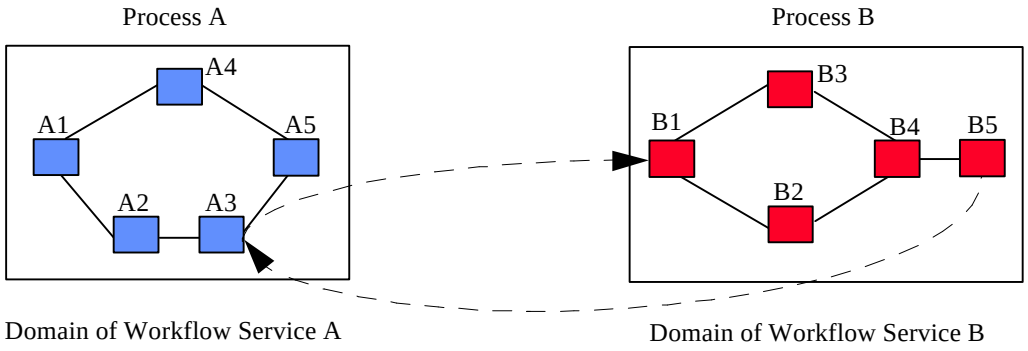

Fig 14 - Nested Subprocesses Model

In the diagram, Workflow Service A has an activity defined (A3) which is enacted as a complete process (B) on Workflow Service B with control returned to Service A on completion. As in scenario 1 earlier, transfer of activity control may be via an applications gateway function or by direct API calls between the two workflow services. The diagram illustrates the simple case with a single entry and exit point in Process B, although activity navigation rules within B may permit other activity flow scenarios, for example process completion conditions enabling the process to be completed prior to activity B5 and control returned to workflow domain A.

3.7.4 Scenario 3 - Connected Indiscrete (Peer-to-Peer)

This model allows a fully mixed environment; the diagram indicates a composite process C, which includes activities which may be executed across multiple workflow services, forming a shared domain. Activities C1, C2 and C5 could be co-ordinated by server A (or even several homogenous servers within a common domain) and activities C3, C4 and C6 co-ordinated by server B.

In this scenario, the process would progress transparently from task to task, without any specific actions by users or administrators, with interactions between the individual workflow engines taking place as necessary.

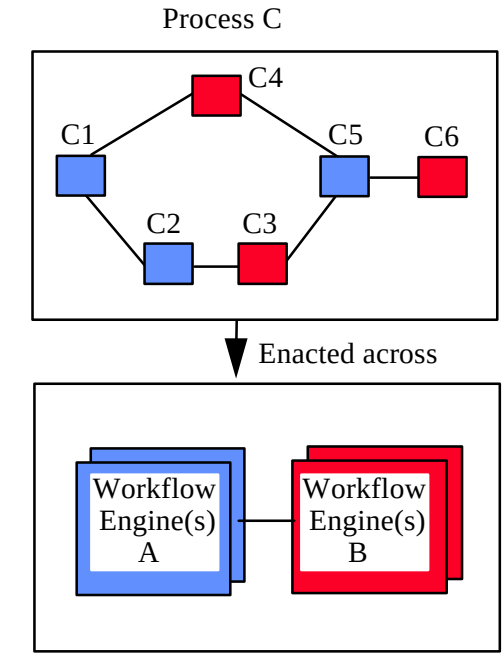

Fig 15 - Peer-Peer Model

This scenario requires that both workflow services support common API sets for communication and that both can interpret a common process definition, either imported to both environments from a common build process or exported between services during the runtime phase. Workflow relevant and application data may also need to be passed between the various heterogeneous engines.

Whilst simply illustrated as an interworking scenario, there are various complexities within the peer-peer model which will require further study. As shown each particular activity is associated with a specific workflow domain, for example predefined within the process definition. Further complexities arise where a specific activity may be executed on either of two independent workflow services or where a particular process instance can be created or terminated independently by either service. Systems administration, security and recovery across co-operating workflow services will also need to be addressed. In the extreme, the two different workflow enactment services may require to share much of the process state data normally maintained internally to each, in effect forming a single heterogeneous service. The Coalition intends to define a number of conformance levels, allowing earlier specifications to cope with simpler scenarios and additional functions to cope with more complex scenarios to be added in the future.

3.7.5 Scenario 4 - Parallel Synchronised

This model allows two processes to operate essentially independently, possibly across separate enactment services, but requires that synchronisation points exist between the two processes. Synchronisation requires that once the processes each reach a predefined point within their respective execution sequences, a common event is generated. This type of mechanism may be used to facilitate functions such as process scheduling across parallel execution threads, checkpointing of recovery data or the transfer of workflow relevant data between different process instances.

In the diagram following synchronisation is shown between activity A3 within process A and activity B4 within process B.

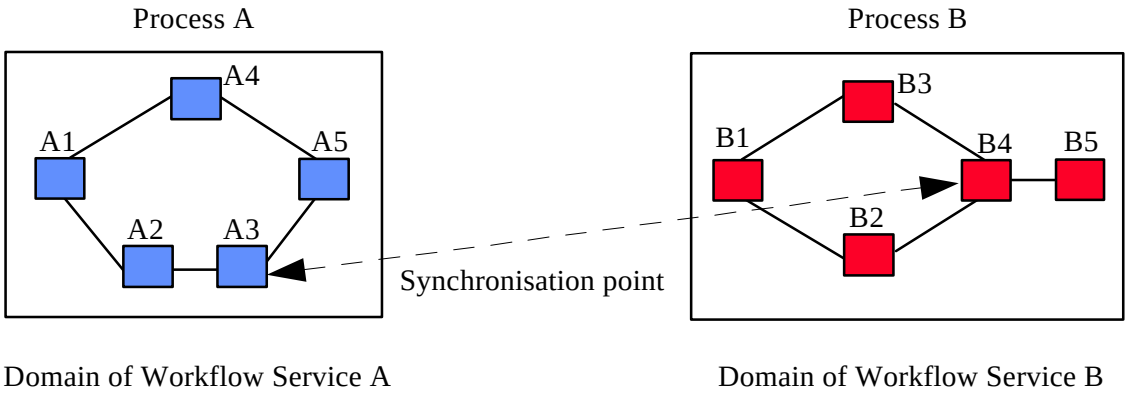

Fig 16 - Parallel Synchronised Model

Matching pairs of work can thus be synchronised at specific points in each process. This requires an event co-ordination and tracking mechanism, in addition to both services being able to recognise tasks from the two process definitions. It is included for completeness but is recognised as lying beyond the scope of the Coalition's current specification activity.

3.7.6 WAPI Interoperability Functions (Interface 4)

The general nature of the information and control flows between heterogeneous workflow systems is shown in diagram 17.

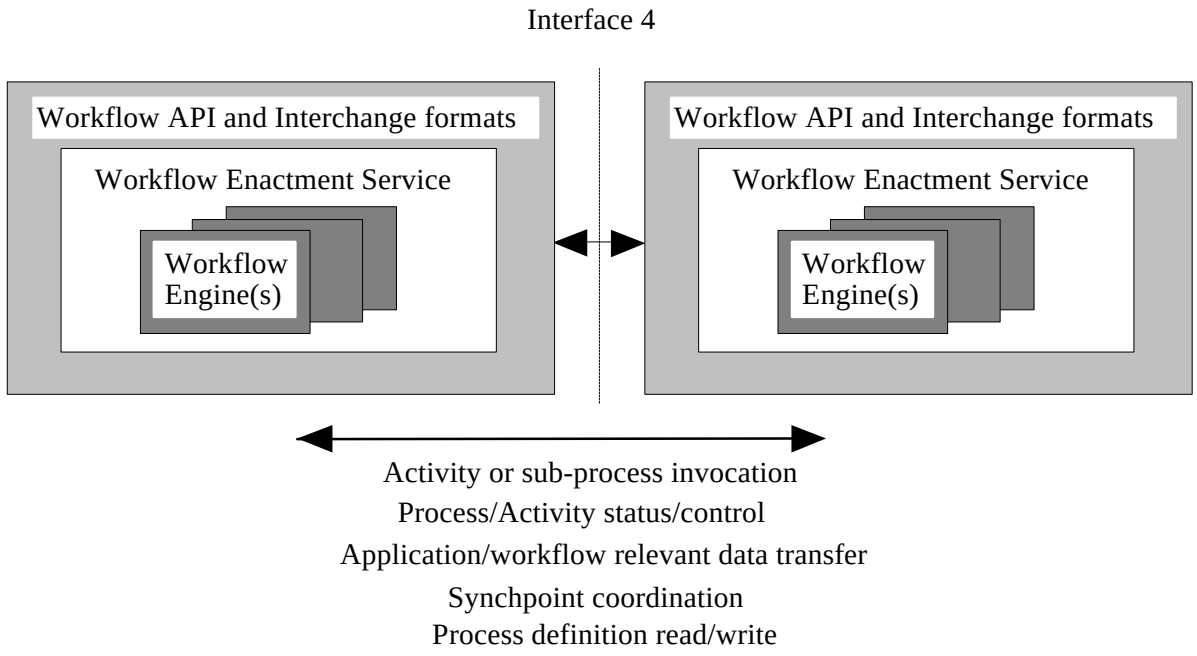

Figure 17 - Workflow interoperability interface

There are two major aspects to the necessary interoperability:

- the extent to which common interpretation of the process definition (or a subset) is necessary and can be achieved
- runtime support for the interchange of various types of control information and to transfer workflow relevant and/or application data between the different enactment services

Use of Process Definitions across Multiple Domains

Where both enactment services can interpret a common process definition, for example generated from a common build tool, this enables both environments to share a single view of the process definition objects and their attributes. This would include activity, application, organisation and role names, navigation conditions, etc. This potentially enables individual workflow engines to transfer execution of activities or sub-processes to heterogeneous workflow engines within the context of a common naming and object model. This approach is particularly applicable to interoperability scenario 3, where several systems are co-operating at peer level, although can also be employed in simpler scenarios.

Where this shared view of a process definition is not feasible, the alternative approach of "exporting" details of a process definition subset as part of the runtime interchange may be possible. The process definition interchange APIs provide a means of requesting object and attribute data from a particular workflow service, thus enabling a workflow engine to obtain process definition data relevant to the execution of an individual activity or sub-process assigned to it in a co-operative enactment environment.

Where process definition interchange by either of the above approaches is infeasible, interoperability is constrained to a gateway approach, in which (typically a subset of) object names and attributes are mapped between the two environments via an application interworking gateway. In this simplest case, the two separate enactment services use their own process definition formats with any mapping between the two handled within the gateway. This approach effectively constrains interworking to the simpler scenarios 1 and 2 or relatively trivial examples of scenario 4.

RunTime Control Interactions

At runtime, the WAPI calls are used to transfer control between workflow services to enact subprocesses or individual activities on a specific service. Where both services support a common level of WAPI calls and a common view of the process definition objects (including naming conventions and any workflow relevant or applications data) this will be done directly between Workflow engines - although this will require agreement on common protocol support for WAPI primitives.

Where this is not the case the WAPI calls can be used to construct a gateway function providing interworking between the two workflow services by mapping the different object and data views between the two environments and (where necessary) supporting different protocol environments into each workflow service. This is illustrated in the following diagram.

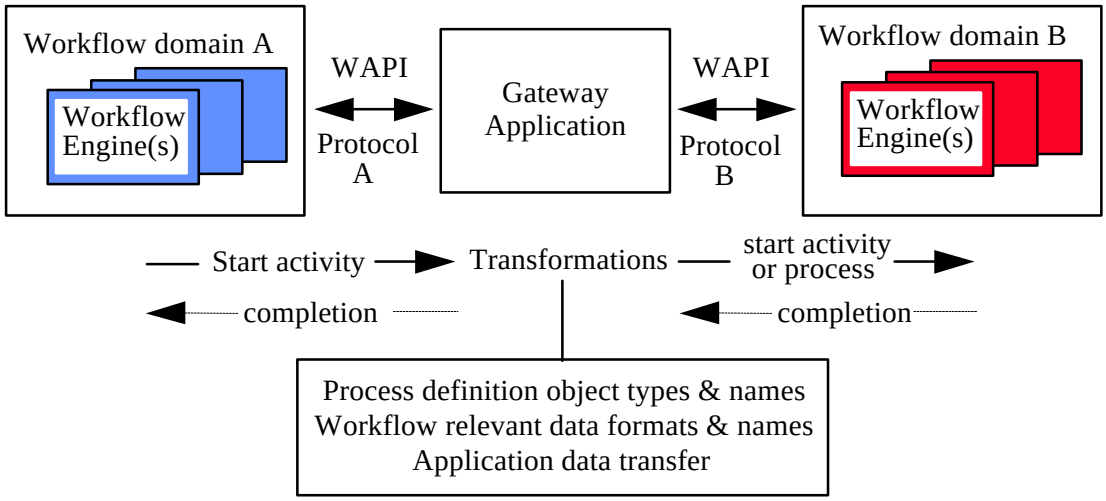

Fig 18 - Gateway operation using WAPI

The diagram illustrates the main principles of gateway operation; depending upon the particular interworking scenario an individual activity from one domain (A) may be mapped to a single activity or a new process / subprocess in the second domain (B).

A large number of WAPI commands are (ultimately) likely to be exploited to support interoperability either by direct call between the two workflow services or via a gateway function. Many of the WAPI commands discussed earlier (sections 3.4.2, 3.5.2 & 3.6.2) are also potentially applicable in workflow interoperability interactions:

- Session establishment
- Operations on workflow definitions and their objects
- Process control and status functions
- Activity management functions
- Data handling operations

A degree of common administration between multiple workflow domains will also be necessary using functions developed for interface 5 (section 3.8.2).

Once activities are being enacted on a separate (subordinate) service, interactions from workflow client applications with the original service (for example query status of activity/process instance, or suspend/resume/terminate process instance) may need "referral" to the subordinate service. Some operations may thus need to be chained across several workflow engines (for example, where different activities within an active process instance are distributed across several machines). Some form of event notification service is also likely to be required to inform the initiating service of activity status changes and completion of activities and/or subprocesses. It is envisaged that a number of additional WAPI operations will be developed, over time, to support these and other functions arising from more complex interworking scenarios.

The range of possible interactions is relatively extensive and complex in terms of state transitions (including, for example aspects such as failure containment and recovery); further study will be

required to develop the necessary conformance levels which could form a practical basis for interoperability between different products.

3.8. Systems Administration

3.8.1. Administration & Monitoring Tools

The final area of proposed specification is a common interface standard for administration and monitoring functions which will allow one vendor's management application to work with another's engine(s). This will provide a common interface which enables several workflow services to share a range of common administration and system monitoring functions.

Although process status commands are defined within the interfaces already described, there is a recognised requirement in some industries for a function to apply overall status monitoring and extract metrics information. The proposed interface is intended to allow a complete view of the status of work flowing through the organisation, regardless of which system it is in; it is also intended to present a comprehensive function set for administration purposes, including specific considerations of security, control and authorisation.

The interface will include specific commands within the WAPI set to manipulate designated administration and monitoring functions. In addition, further review is intended to ascertain to what extent this interface can exploit existing protocol mechanisms such as CMIP and SNMP to set and retrieve management status and statistical information defined in an open MIB (Management Information Base).

3.8.2. Administration & monitoring Interface (Interface 5)

The interface as illustrated shows an independent management application interacting with different workflow domains, although alternative implementation scenarios are also feasible; for example the management application may be an integral part of one enactment service, although capable of managing various functions across additional (heterogeneous) workflow domains.

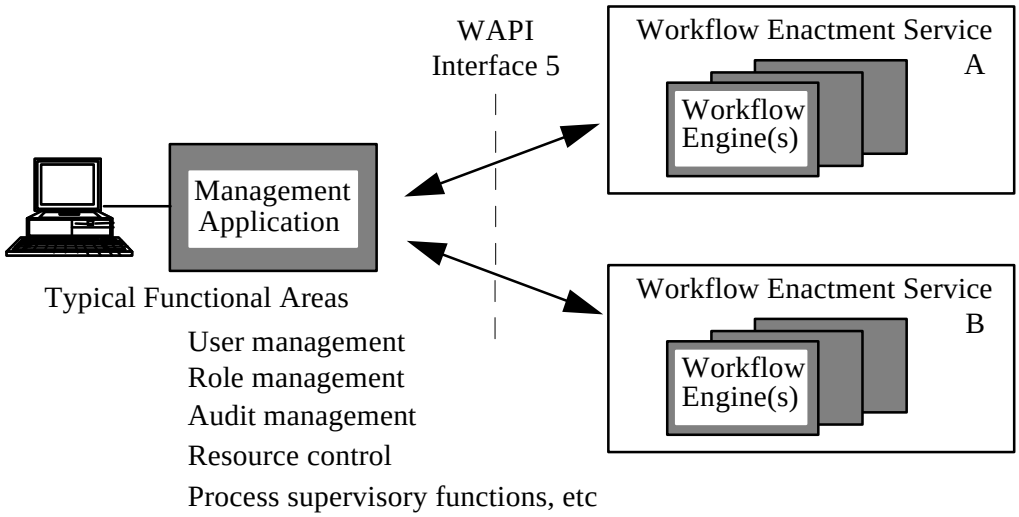

Fig 19 - Systems Administration & Monitoring Interface

It is also feasible for the management application to take on other management functions, beyond those shown. For example, it may also manage workflow process definitions, acting as a repository and distributing process definitions to the various workflow domains via operations within interface 1. The detail of this interface is for further study, but it is envisaged to include the following types of operation (some of which are common to other interface areas):

User Management operations

- establishment / deletion / suspension / amendment of privileges of users or workgroups

Role Management operations

- define / delete / amend role:participant relationships
- set or unset role attributes

Audit Management operations

- query / print / start new / delete audit trail or event log, etc

Resource Control operations

- set / unset / modify process or activity concurrency levels
- interrogate resource control data (counts, thresholds, usage parameters, etc)

Process Supervisory Functions

- changing the operational status of a workflow process definition and/or its extant process instances
- enabling or disabling particular versions of a process definition
- changing the state of all process or activity instances of a specified type
- assigning attribute(s) to all process or activity instances of a specified type
- termination of all process instances

Process Status Functions

- Opening / closing a process or activity instances query, setting optional filter criteria
- Fetching details of process instances or activity instances, filtered as specified
- Fetching details of a specific (individual) process or activity instance

4. WAPI Structure, Protocols and Conformance

4.1 WAPI - Functional Overview of APIs

The WAPI is envisaged as a common set of API calls and related interchange formats which may be grouped together as required to support each of the five functional interface areas. Operations already identified across these 5 interface areas (and discussed in section 3) include those in the following groups:

API Calls

- Session establishment
- Operations on workflow definitions and their objects
- Process control functions
- Process control supervisory functions
- Process status functions
- Activity management functions
- Data handling operations
- Worklist/Workitem Handling Functions
- User Management operations
- Role Management operations
- Audit Management operations
- Resource Control operations

Data Interchange Functions

Interchange formats are expected to be defined to cover:

- Process definition transfer
- Workflow relevant data transfer

API Call Structure and Naming

API calls will be defined initially in terms of their logical operations, the datatypes on which they may operate (ie as call parameters) and the supporting data structures referenced from such parameters. Language bindings are expected to be developed, initially for the C language and subsequently for other important development environments (both C++ and IDL are candidates for further study). Naming conventions are being specified for the call functions themselves, plus the supporting datatype definitions, parameter types and data structures (see document reference WFMC_TC00-1013 for API naming convention information).

4.2 WAPI Protocol Support

The WAPI calls will be able to function in two types of interconnection scenario:

1. Where an exposed WAPI interfac e is provided as a boundary function to a workflow enactment service (eg as vendor stub routines embedded in a client application or application agent), vendor specific mappings may be used to encode the call and associated parameters to the particular vendor protocol environment used to communicate with the workflow engines.

2. Where direct interworking between different products is provided (eg interoperability between different workflow engines), open (common) protocol support will be necessary. This will require a standardised mapping from WAPI calls onto one or, more likely, several interworking protocols.

The expected areas of standardisation relating to these two scenarios are as illustrated in the following diagram.

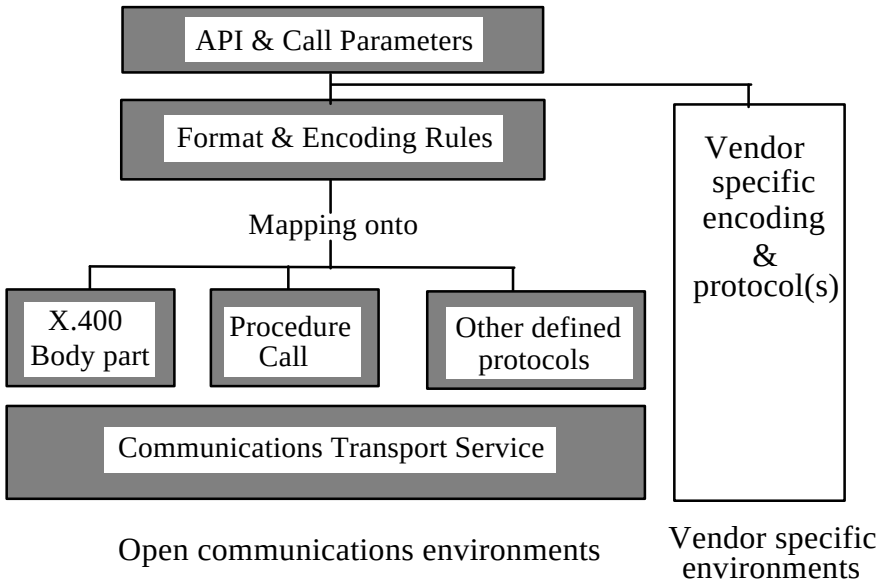

Fig 20 - WAPI protocol support

The details of protocol usage within WAPI are for further study, but it is expected that WAPI mappings will be developed onto important communications environments, ie those widely used by workflow products available in the market. Initial support for client application integration and workflow interoperability via an application gateway can be achieved using approach 1 (with vendor specific protocols); however, this approach has some inherent limitations and the development of appropriate protocol usage specifications is a clear requirement in the medium term.

Implementations would be expected to identify the particular communications environment(s) which are supported, along with the specific API command set options being implemented for the particular interchange function. This subject will be further considered as conformance rules are developed (see following section).

