The Workflow Management Coalition Specification

Interworkflow Application Model: The Design of Cross-Organizational Workflow Processes and Distributed Operations Management

Groupware Technical Sub Committee1
The Japanese Standards Association

Submitted to WfMC as:

Document WFMC-TC-2102
Feb 97

Table of Contents

1. Introduction

Workflow management systems are being deployed in enterprises as a new kind of information infrastructure, helping to streamline and automate office work processes. A key feature of these systems is that they automate the workflow itself. Such a system can predefine the routine work procedures with the relevant information, the role of the participants at each step of the work, and application software required to process each step. Then each new case is automatically assigned to the participants in proper sequence, information is delivered to the people needing it, and applications are launched as needed. The system automatically performs this management, thereby helping to eliminate human error and streamlining information flow, so that the work can be performed efficiently.

Workflow systems like this are today being used in a variety of fields. They range from relatively small-scale tasks such as handling business trip paperwork, supply orders and other in-house forms, to larger jobs such as managing the progress from order receipt to delivery, or even larger workflows on the scale of tracking financial transactions in the banking industry.

As workflow management systems establish a place in companies, it is only natural that consideration be given to coordination of different workflows within an enterprise or even across different companies and industries. Today workflow management systems are mainly introduced with the aim of automating workflow inside a given department or company. The next logical step is to interconnect the systems of different departments or to extend their reach beyond the walls of one company, applying workflow management to a more global coordination of tasks and to a broader sharing of work and information. The result will be a smoother flow of work and information not just within one group but across organizations, for an overall improvement in efficiency. This concept could be applied, for example, to the procurement work of one company by connecting its procurement workflow to the workflows of the companies from whom products are procured, and to the workflow involved in managing order-taking and delivery. The need for this kind of coordination across workflows has been pointed out variously, and in fact some efforts are already being undertaken in this direction.

Against this backdrop, the Japan Standards Association (JSA) last year began studying the support functions necessary for realizing interaction among workflows. The study is looking at the mechanism for interconnecting workflows across departments and across enterprises, for the purpose of devising interworkflow support mechanisms achieving coordination among different workflows. This interworkflow support is to be implemented in the following three ways.

1. Support for analyzing and modeling interworkflow across departments and companies.

In order to link workflows across organizations, agreement must first be reached among the affected organizations as to how the work is to be divided among them, how information is to be shared, and in what sequence the work is to be carried out. (In our studies we use the term “interworkflow” for the processes resulting when different workflows are linked.) The support here is for shaping such an agreement, modeling, making clear definitions and analyzing definitions (using simulations and detecting bottlenecks), and so on. The support functions are aimed at achieving a workflow that most efficiently divides the work among each organization, and defining that workflow clearly and precisely.

2. Support for designing workflow in each organization.

Based on the definition of the interworkflow with other organizations, the workflow within each organization must be devised. These support functions are aimed at structuring individual workflows quickly and easily.

3. Support for workflow operations management in a distributed environment.

The third kind of support is for controlling the operations between each of the individually designed workflows so that each task in the interworkflow can be carried out efficiently while exchanging information among them as needed.

The object in realizing these support functions is to provide a total interworkflow support mechanism from beginning to end, extending to the instituting of the interworkflow and its operations management.

In this paper, section 2 looks at a specific case study where interworkflow is required, and presents an interworkflow model based on it. Section 3 outlines the studies carried out in the last fiscal year toward realizing interworkflow support, while section 4 considers the ideal form of interworkflow support mechanisms based on those studies. Section 5 discusses the interworkflow modeling undertaken in this fiscal year and the support mechanisms for workflow design. Section 6 summarizes standardization trends by the Workflow Management Coalition (WfMC) in areas that relate closely to interworkflow support. Finally, section 7 presents a summary of this paper.

2. Interworkflow model

In order to study the linking of workflow management systems among multiple organizations, we need to create a model of workflow within and across organizations. To this end we examine an actual case study, and model the workflow across the organizations involved.

- The work in the case study

As our case study we chose the work of product procurement by NTT, centering around the International Procurement Office. The reasons for selecting international procurement work are, (1) it involves coordination both across different departments in NTT and with an indeterminate number of supplier companies; and (2) it involves mostly routine work and is therefore suitable for application of a workflow management system.

The work of international procurement involves the following steps.

1. A development department prepares a draft of the procurement specification for the required product.

2. After completing the draft procurement specification, the development department requests the International Procurement Office to procure the product. At the same time it sends the draft procurement specification (word processor file) to the International Procurement Office.

3. The International Procurement Office receives the procurement request. At this point the work of the International Procurement Office begins.

4. The International Procurement Office checks the procurement request for completeness, commenting and revising as necessary. The comments are made on a word processor, and the file is sent to the development department. This procedure is repeated as often as necessary until the procurement request is completed. During this time data are exchanged several times between the International Procurement Office and development department.

5. After the procurement specification is completed, the procurement offer is publicized in Japan and worldwide.

6. When suppliers request the procurement specification, the International Procurement Office sells it to them. It also handles questions and answers regarding application for the procurement. This process continues from the time the procurement offer is made public until the International Procurement Office work is completed.

7. Supplier selection is made primarily by the development department, and the results are notified by the International Procurement Office to each supplier.

8. When suppliers are notified of the selection results, they perform post-processing depending on the result.

9. The procurement procedures following supplier selection are carried out by the Procurement and Supply Department, to which the International Procurement Office hands over

its work. At this point the work of the International Procurement Office is completed.

10. After the procurement procedures are completed, the department that requested the materials handles installation and maintenance.

The workflow for international procurement is shown in Figure 1.

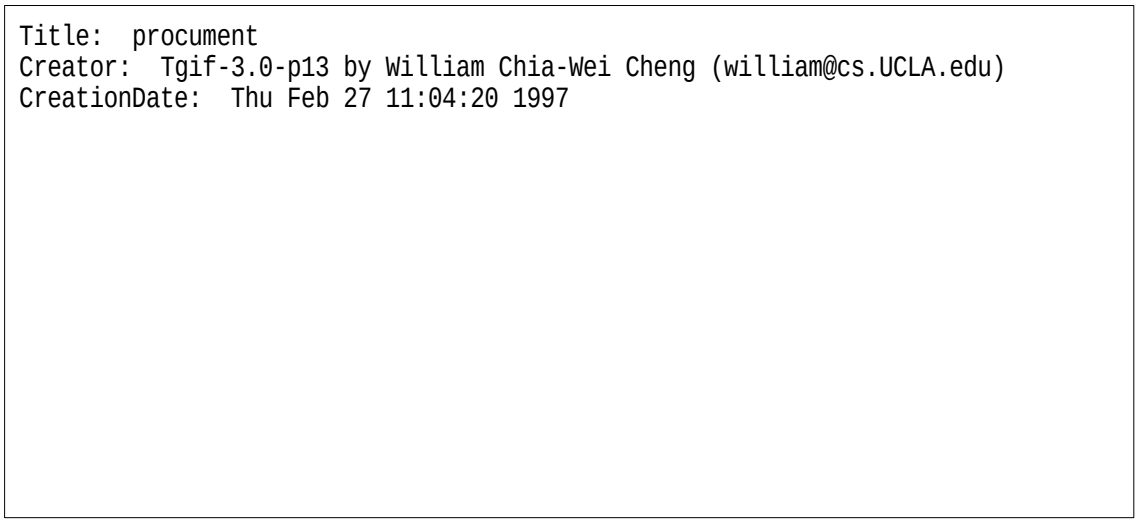


Figure 1. Procurement work by NTT

- Modeling workflow across organizations

Based on the above case study, a model is created of work linkages between organizations. The modeling focuses on three points, (1) the start and end of a work linkage, (2) structures for controlling the workflow, and (3) exchanged data.

- The start and end of a work linkage

For the work in the case study, the start and end of work linkages are classified into three types, (1) cooperative type, (2) hand-over type, and (3) open offer type. The cooperative and hand-over linkages are one-on-one, while the open offer type is a linkage between one entity and an indeterminate number of other entities.

1. Cooperative type (one-on-one linkage)

One organization has work done by another organization and receives the results of that work. The organization receiving the request for the first time creates a new workflow (instance). Sending of data takes place multiple times until the linkage ends. The data are sent from the activity (logical steps making up the job) of one organization to the activity of the other organization.

This kind of work linkage can be seen in international procurement work in the interaction between the developing department and the International Procurement Office.

2. Hand-over type (one-on-one linkage)

Work is handed over from one organization to another, at which point the linkage ends. The organization receiving the request for the first time creates a new workflow (instance). Data sending from the requesting organization to the other organization occurs only at the time the work is handed over. This type can be considered a special form of the cooperative type.

An example of this type of work linkage is the passing of a case from the International Procurement Office to the Procurement and Supply Department.

3. Open offer type (one-on-many linkage)

In this type of linkage, requests to perform work are made from the workflows (instances) of plural organizations in response to a workflow (instance) already executed by another organization. This is a kind of derivative of “Cooperative type”, though, at the organization receiving the requests, at the organization receiving the requests, the names and number of organizations making the request is not determined at the time the workflow is defined.

Sending of data takes place multiple times until the linkage ends. The data are sent from the activity of one organization to the activity of another organization.

This type of work linkage takes place in international procurement between the International Procurement Office and suppliers.

```
Title:  linkage
Creator:  Tgif-3.0-p13 by William Chia-Wei Cheng (william@cs.UCLA.edu)
CreationDate:  Thu Feb 27 14:53:51 1997
```

(a) Cooperative type (b) Hand-over type (c) Open offer type

Figure 2. Types of work linkages

- Structures for controlling workflow

The structures by which workflow is controlled in an organization after a work linkage is started are (1) sequential, (2) conditional branch, and (3) loop. The features of each type of structuring are noted below.

1. Sequential

In sequential control, after one organization completes an activity and passes control to another organization, the work is continued without any condition decision. In order to pass control to the other organization, the passing of control needs to be made explicit to the other organization. Even after control is passed to the other organization, the first organization may continue working.

This kind of control passing takes place between the International Procurement Office and suppliers, for example, in the direction from procurement specification purchase to procurement

specification sale.

2. Conditional branch

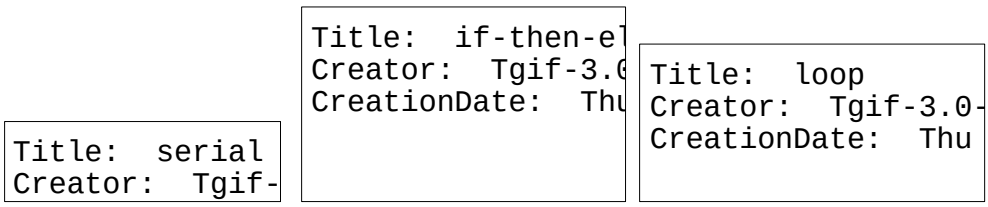
In the conditional branch type, after one organization completes an activity and passes control to another organization, some kind of condition decision takes place and a branch occurs. As values for the condition decision, the exchanged data explained in section 2.2.3 can be used, for example.

This kind of control passing is to be seen in the branch made by suppliers who receive the selection results, based on those results.

3. Loop

In looped control, after one organization completes an activity and passes control to another organization, some kind of condition decision is made and a loop occurs. The exchanged data explained below, for example, can be used in the condition decision.

An example of this type of control passing is in the preparation of a procurement specification, which gets passed between the development department and International Procurement Office for comments and revisions.



(a) Sequential (b) Conditional branch (c)Loop

Figure 3. Control structures

- Exchanged data

The data exchanged among organizations are structured data. These data are called messages.

A message has multiple fields. Each field consists of an ID and a value paired with it. The message itself may be given a name. The fields that must be dealt with in interworkflow include numerical values, character strings, and files, etc.

The field values and message names can be handled by the application programs in the organization receiving the message.

For example, the application sent by suppliers to the International Procurement Office are messages consisting of three fields, the bidder (character string), contact information (character strings), and proposal (file).

- Summary

In order to study the interaction of workflow management systems in different organizations, we devised a model using the NTT International Procurement Office as a case study, and focused on (1) the start and end of a work linkage, (2) the structures for controlling the workflow, and (3) the data exchanged in the workflow.

In the following sections we discuss the implementation of this model and the notation method.

3. Interworkflow support studies to date

As of 1995, the only choices for linking the workflow management systems in different organizations were to use the same system in every organization or to manage the interworkflow manually. In order to realize the work linkage those described in the previous section, NTT designed a workflow management system connection control facility and implemented a prototype system.

This section presents an overview of the connection control facility, then evaluates the prototype system and indicates the remaining issues that were discovered in that study.

- Overview of the workflow management system connection control facility

The connection control facility is positioned between the workflow management system layer and the communication layer (see Figure 4). In order to implement linkages among different workflow management systems, this control facility performs the following functions. (1) It manages the correspondence between instances on other workflow management systems and those on the local workflow management system. (2) It requests processing tasks to other workflow management systems. (3) It accepts processing requests from other workflow management systems and executes them on the local system.

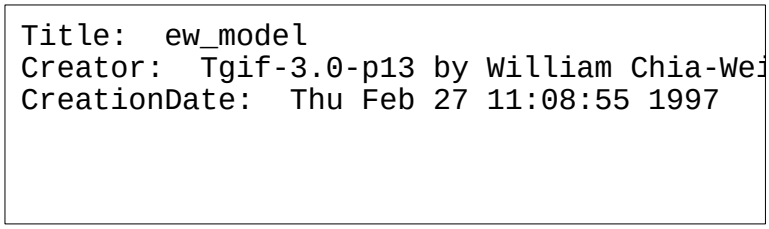


Figure 4. Position of the connection control facility

The followings were defined as commands provided by the protocol used between connection control facilities. (1)commands for instance creation request/response (2) command for data transfer between instances, (3) commands for requesting the canceling of a linkage between instances and for responding, and (4) commands for instance status reference request/response. Since the purpose here was to confirm the feasibility of linkages among different workflow management systems, we defined the bare minimum of functions, without taking into account unanticipated events or security needs.

These functions were then implemented for three kinds of commercial workflow management systems. Email protocol was used as the communication protocol between connection control facilities. No modifications were made to these commercial systems themselves. The connection control facility was implemented by means of functions for linking to the other programs of each system. For this reason the interfaces between the workflow management system and connection control facility, and the workflow definition methods, were different for each system.

- Evaluation

Connection tests were carried out to determine the effectiveness of the prototype system. The results

showed that we were able to achieve the initial objective of interaction between different workflow management systems used in different organizations. The connection tests did, however, reveal the need for more work in the following areas.

1. Workflow definition methods

In the prototype system, the division of work among organizations and the procedure for data transfer between organizations were first defined on paper, after which workflow definition was performed separately in each organization on each workflow system. This approach led to problems such as inconsistency of definitions on different systems (errors in data destinations, etc.), or the workflow defined on one system not being usable on another.

The way to solve this deficiency is seen to be by using a definition language common to all workflow management systems and defining the work as a whole, across multiple organizations.

2. Error handling

In the prototype system, interaction among workflows was mapped to commands focusing on data transfer. These commands were adequate so long as the work was carried out normally. However, if someone cancels a job, for example, or work has to be partially redone, it is conceivable that these commands will no longer suffice.

To make the system better applicable to real work situations, besides commands for data transfer there is a need for ways of managing work across multiple organizations, and especially management methods that can handle errors arising in the course of the work.

3. Security

Since email is used as the transport protocol between workflow management systems, it would be easy for someone to falsify data or crack the system in other ways, such as diverting data to someone else while it was being transferred. In the practical systems, we need to implement security mechanisms for authenticating users and for encrypting transferred data.

In addition, we have to provide the ways of controlling the scope and timing of information that gets disclosed even to rightful users. These controls are particularly important in the type of work where information changes as the work is carried out, and where information is provided on demand.

4. Interworkflow support functions

In this section we discuss the kinds of functions that should be provided for interworkflow support. Here the three kinds of functions are taken up, support for analysis and modeling of interworkflow among organizations, support for devising a workflow in each organization, and support for workflow operations management in a distributed environment.

- Support for analysis and modeling of interworkflow among organizations

Linking workflows across organizations requires agreement among each organization as to how the workflow is to be divided among them and what kind of information is to be shared. If, for example, the procurement workflow in one company is to be coordinated with the ordering and delivery management workflow in the supplier companies from which it procures materials, then the procuring company must make known to the supplier companies, or reach agreement on, such matters as the types, format and timing of information they need to submit in order to participate in the procurement, the kinds of actions the suppliers must take in connection with the procurement, and the procedures involved.

Three steps are required for forming such an agreement. The first step is that of modeling the interworkflow, namely, determining the division of the workflow in each organization, deciding how information is to be shared, and defining the work procedures. This step takes an overview of the work and information flow in each organization and studies the best form for the work to take. The second step is to analyze the contents of the model and use simulations or other means to study the desired form. The results of this step are fed back to the modeling step. The third step is to check the results of the modeling carefully so as achieve strict definitions, eliminating error and ambiguity.

We believe support functions like the followings need to be provided to assist these three steps.

- A formal language for writing interworkflow definitions.
- A graphical interworkflow modeling tool
- Analytical tools for simulating interworkflow, using animation and the like.

A support environment that integrates these functions seamlessly is essential. In particular, the formal definition language is a foundation for realizing the other two functions, making this function especially important.

- Support for workflow design in each organization

The interworkflow defined by the three steps described above defines an inter-linking process to be carried out among the workflows of a number of organizations. For actual execution on a system as a workflow, not only must the linking with other workflows be defined, but specific definitions must be made as to how the workers are to carry out the activities in each workflow. Moreover, the workflow designed in this way needs to be linked properly with the other workflows in accord with the interworkflow definition.

For designing workflow based on this kind of interworkflow definition, assist functions are essential to facilitate the design process and ensure it is done accurately. No matter how carefully the interworkflow is defined, it will be of little worth if errors are made in workflow definition at the stage of designing workflow in each organization. In fact, the lack of such a support function in last year's research resulted in a number or problems arising, such as inconsistencies among different workflow definitions and misdirected data between organizations.

To solve such problems, the following support functions need to be provided.

- A mechanism for semi-automatic generation of corresponding workflow definitions from the interworkflow definitions, in order to simplify the workflow design process.
- A mechanism for checking whether a workflow definition is consistent with the interworkflow definition.

These support functions are based on the earlier mentioned formal language for describing interworkflow, and it is necessary to realize a consistent support for implementing a function for conversion from the descriptions and for testing the appropriateness of the descriptions.

- Support for workflow management in a distributed environment

Managing the workflow operations of each organization in the interworkflow must be done in a distributed environment. Since the interworkflow defines the process of linking among different workflows, naturally the operations management takes place across different workflows. Moreover, those workflows are not constructed on the same management system, but is distributed across the systems of each organization. The interworkflow for international procurement, for example, includes the procurement workflow and also the ordering/delivery management workflow of each supplier company; and these workflows are likely to be distributed across the procuring company's system and the systems of each of the supplier companies.

The following support functions are necessary for implementing operations management of workflows in a distributed environment.

- A mechanism for controlling execution of workflows designed on other workflow management systems.
- A mechanism for exchanging information with workflows running on other workflow management systems.
- A mechanism for monitoring the status of workflows running on other workflow management systems.

In last year's study, the main focus was a management mechanism for realizing linkage among these kinds of distributed workflows. In fact, however, the management mechanism studied then was limited to the exchange of information among workflows in the form of files, and is unable to implement a closer linkage. For example, if all or part of a workflow is canceled in the course of interaction, or if part of a workflow must be redone, the problem of how to reflect that in other related workflows has to be resolved. An operations management mechanism must be realized that can implement a finer degree of linked operation.

5. Support for interworkflow modeling and workflow design

As noted earlier, a formal language infrastructure is needed for describing interworkflow in order to perform the necessary interworkflow modeling. In addition, for supporting workflow design based on the interworkflow, support functions are needed that can semi-automatically generate workflow process definitions from the interworkflow definitions.

This section discusses the studies to date on these two points. First we present our basic thinking on the design of a formal description language and outline our research to date on an interworkflow definition language. Then we outline a method for semi-automatic conversion of the descriptions in this language to workflow process definitions.

- Design of a language for describing interworkflow

As the programming model of the language for describing interworkflow, we chose the actor model as our base. The actor model is a model in which computing take place by means of interactions among plural actors acting concurrently.[3] In this model, each of the actors is treated as existing independently of all the other actors, and each of their computer resources are considered in isolation from the others. Interaction among actors takes place with message passing being the only means of conveying information.

Interworkflow, as noted earlier, is the process by which interaction among workflows takes place. If we think of a workflow as an actor in the above model, each workflow has its own resources for workflow operations, and the workflows are executed in parallel. Moreover, messages are exchanged between workflows, and linkage among them is realized by interacting to initiate a workflow execution as needed. Figure 5 shows an example of such workflows. In the linkage illustrated here, workflow A initiates the execution of workflow B, after which it sends message A to the workflow it initiated earlier. Workflow B receives message A from the workflow by which it was initiated, then sends message B. When workflow A receives that message, it sends message C.

There are two advantages to adopting such a mode. One is that it allows for the independence of organizations. Each organization has the ability or the right to decide things on its own, without unnecessary interference from other organizations. This includes decisions as to how it deploys its workers and allocates work, as well as how it designs its workflow. The right to decide such matters on its own is what we mean by the independence of organizatios.

Because an organization responsible for part of the interworkflow is also involved in the workflows of other organizations, the interaction with other workflows must be specified precisely. At the same time, the specific ways in which an organization processes its own workflow should be left to the organization to decide on its own. The actor model treats the inside of each actor as a black box, dealing only with the interaction with other actors that can be seen from the outside. Similarly, the internals of a workflow may be treated as a black box in our model, making only the linkage with others visible, so that an interworkflow can be defined without violating the independence of each organization.

Another advantage of this model is its applicability to the multifaceted nature of linkage relations. An organization has such relations with many different kinds of organizations. Conceivably, which

organizations a given organization links with may appear differently depending on the organization. In our example of procurement work, the company supplying a product may actually be a sales agency representing the manufacturer of the product, whose only role is as an intermediary between the manufacturer and the procuring company. In this case the procuring company in forming its linkage need only be aware that this is the place from which it purchases the product, regardless of whether it is an agency or not. The supplier of the product, however, must be aware of its links not just with the procuring company but also with the manufacturer. This is what is meant by the multifaceted nature of linkage relations, namely, that the relations are different depending on the perspective.

By treating the internals of each actor as a black box, the actor model is able to represent multifaceted relations of this sort. In a situation, for example, where two actors A and B interact, it is possible to add another actor C and commission C to do part of actor B's processing without making any changes in the interaction between actors A and B. In this example, actors A, B, and C can be taken as representing the procuring company, sales agency, and manufacturing company, respectively, in order to apply the model to the case above. In this way the actor model can explain multifaceted relationships without contradiction.

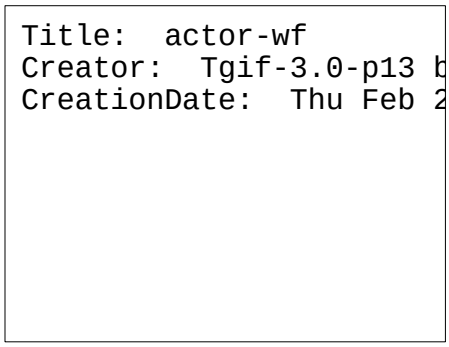


Figure 5. Woflow linkage based on the actor model

- Overview of the language for interworkflow description

The interworkflow description language is made up of four language elements: process, thread, action, and message. Here "process" is something that represents the behavior of the workflow corresponding to that process. It is expressed as a prescription for the order in which the actions take place in its interaction with other workflows. A process instance is the entity that carries out the actual actions in accord with the defined behavior. An action represents the internal processing of a workflow or interaction with another workflow, and is the smallest element of a process. A message represents the structure and type of data exchanged in data transfer between workflows. It is handled by means of a message exchange action. A thread is a special subprocess contained in a process, and is used, for example, when an instance of a given process must exchange messages with an indeterminate number of instances of another process. A thread is one kind of process, and takes the form of an instance generated when it is executed. Each of these elements is described below, giving an overview of the interworkflow description language.

- Action

An action is the smallest element in a process. There are basically two types of actions. One type is an external action, representing an interaction that takes place with another process. There are three kinds of external actions, that of creating an instance of another specific process or thread, that of sending a message to a specific instance, and that of receiving a message from a specific instance.

- An instance create action generates an instance of a designated process or thread. As a result, a workflow instance is created corresponding to that process or thread, and the workflow processing begins.
- A message send action sends a designated message to a designated instance.
- A message receive action waits for the arrival of a designated message from a designated instance, then receives the message contents.

The other type of action is an internal action, representing internal processing in a process. This action has no effect on other processes, representing only the need to perform work having a certain name.

- Message

A message represents the data exchanged by the above message send and message receive actions. Each message is assigned a unique identifier, which is used by those actions to identify the messages to be sent and received.

A message consists of one or more fields, each with an identifier. When a message send action occurs, internal data values of the corresponding workflow are entered in the proper field, then the message is sent. When a receive action occurs, the values of each field of the received message are put into the internal data of the corresponding workflow.

- Process

A process represents the behavior of a workflow corresponding to that process, and is expressed as a series of actions arranged in the order they are to be executed. The designation of execution order may be as a simple sequential order, or two or more blocks may be processed concurrently, or of two or more blocks only one may be processed based on specified conditions, or a block may be looped until certain conditions are fulfilled. Any of these types of control structures are possible, and they may be used to express the interaction with other workflows.

The example given in Figure 6 shows a fairly complex case involving three processes. The vertical bold lines are the individual processes, which are executed in the downward direction. is an action that initiates the execution of another process, is a message send or receive action, and is an action that performs processing internal to a process. Process A initiates the execution of an instance of B, sends two messages to B, then receives a message from B, after which it terminates. During this time it performs two internal processing actions. Process B immediately after being started invokes C, and after receiving a message from B, branches into two processing blocks, both of which it processes in parallel. As a result of this branch, the messages from processes A and C are processed concurrently. This notation lets us easily express the types of interaction of each process with the other processes, as well as the order in which the interaction is to be carried out.

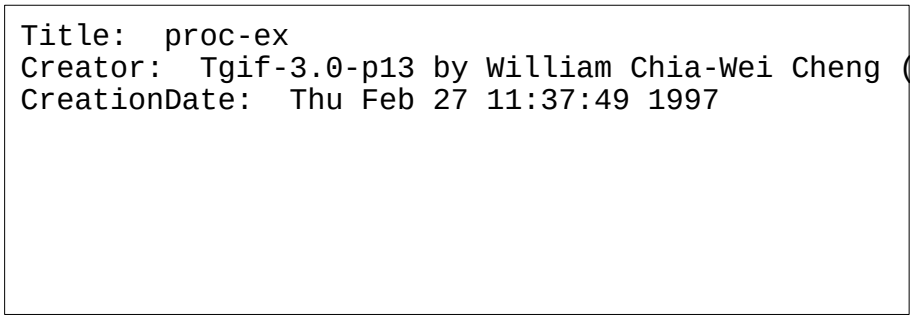


Figure 6. Examples of processes

- Thread

A thread is a special subprocess contained in a process, which is expressed in the same way as a process except for the fact that it cannot contain another thread.

A thread is used in such cases as when a given process must exchange messages with an indeterminate number of instances of another process. This can be illustrated by looking at the interworkflow in a public tender offer. Here the process that makes the tender offer establishes a certain time frame, and during that time a number of bidding processes tender bids, each of which the bid offerer examines. The actual number of bidding processes is unknown to the offering side when the offer is made. Also, each of the bids must be processed concurrently by the bid offerer. If a tender like this were to be processed using a looped control structure, the processing could be expressed as shown in Figure 7. This has several disadvantages, since each of the bid examinations would end up being processed sequentially, with one examination process unable to begin until the previous one had been completed. It would also be difficult to express the processing for comparing each of the intermediate results in the course of the examining.

If we instead use threads to represent the processing, first of all the bid examination processing is defined as threads of the tender offering process. Then at the start of bid acceptance at the offering process, the acceptance of threads is declared. Thereafter the bidding side requests the start of examination processing threads by the offering side. Each time the offering side accepts the start of a thread, it spawns a new examination processing thread. All this is illustrated in Figure 8. When the bidding period ends, the offering side declares the completion of thread acceptance, as a result of which no more bid acceptance threads are spawned. Handling the bid examination processing in this way means each examination can be processed concurrently. Moreover, the tender offering process and each of its threads can be processed concurrently, allowing the intermediate results of the examination processing to be obtained at any time from the tender offering process. The introduction of threads makes this method applicable to a wider range of interworkflow, such as tender offers, that would otherwise be hard to express.

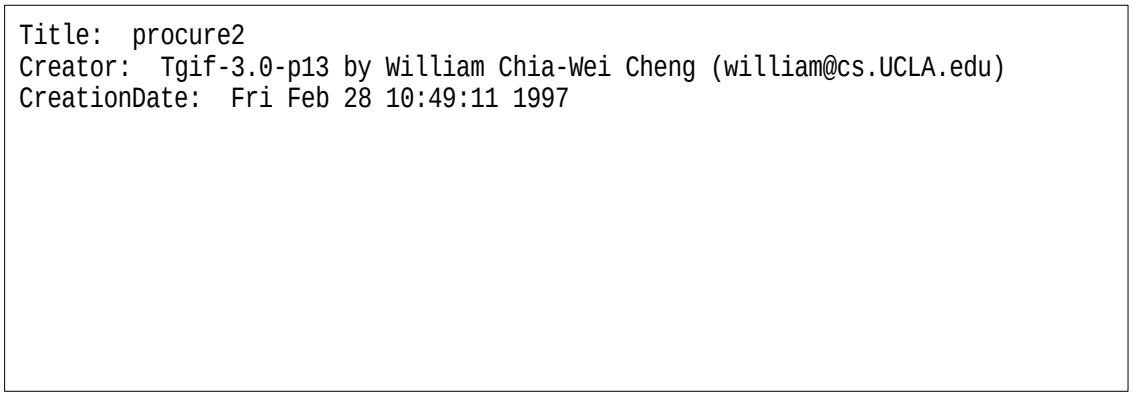


Figure 7. Processing a public tender offer by looping

Figure 8. Processing a public tender offer by thread

- Notation and examples

In the examples shown in Figure 6 and following, processes are illustrated by means of diagrams for explanatory purposes. The actual interworkflow description language, however, expresses processes in text format. The process descriptions are divided into two main parts, a header and a body. The header defines messages and contains the process declaration, whereas the body actually defines the process behavior.

Here we use a sample problem to explain how a process is described using the interworkflow language. The example is of a procurement interworkflow by means of a public tender offer. The interworkflow includes two processes, a procurer-side process and a supplier-side process, with the procurer process containing bid examination threads. The interworkflow performs its processing by means of procedures like those illustrated in Figure 9, which uses the same notation as in the previous figures.

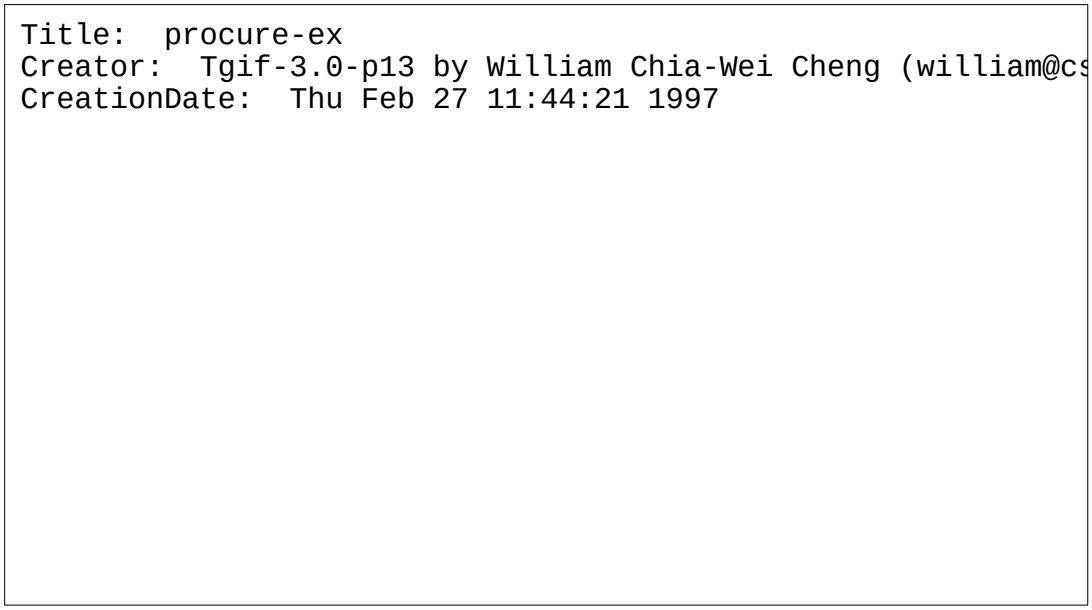


Figure 9. Interworkflow for procurement by public tender offer

When our description language is used to describe this interworkflow, it looks like Figures 10, 11 and 12. Figure 10 shows the header part of the workflow, which defines the messages used and gives the process declaration. Figure 11 shows the procurer process, while Figure 12 shows the supplier process. The parts of a line after // are comments.

The message definitions in the header part shown in Figure 10 define three messages, for the tender proposal, adoption notice and rejection notice. The tender proposal message, for example, is defined as consisting of two fields with character strings as values (bidder's name and contact information) and one field with file contents as its value (the bid description). The contents of the other messages are defined similarly.

Then in the header part, two processes are declared, a procurement process and a bidding process. A process declaration gives the name of the process and also declares the threads contained in that process, declares linked processes indicating which other processes the process interacts with, and declares what kind of linkage relations exist between the linked processes. There are two kinds of linkage relations, CREATOR and INSTANCE. CREATOR means that an instance of the process is created by that linkage process. INSTANCE represents any other case, either that an instance of that linked process is newly created and started in the process, or else that a rendezvous takes place with the instance of a process created somewhere else. In the case of the procurement process, the linked processes declared are the bidding process and its thread, the bid examination process.

A thread declaration is largely similar to a process declaration, declaring the thread name and the thread's linked process. In the foregoing example of a bid examination thread, it means this thread is spawned by the bidding process.

```
message tender proposal {
    string bidder;
    string contact information;
    file bid description;
}
message adoption notice {
    string adoption reason;
    file contract conditions;
}
message rejection notice {
    string rejection reason;
}

process procurement process def {
    thread bid examination def {
            process bidding process : CREATOR;
    }
}
process bidding process def {
    process procurement process : INSTANCE;
    thread procurement process, bid examination : INSTANCE;
}
```

Figure 10. Interworkflow description for procurement by public tender (header part)

The body of each process defines the actual behavior of the process. For example, in Figure 12 the bidding process behavior is defined. First the internal processing for bid preparation takes place, in "**act** [preparation for bidding].” As part of this processing, the retrieve clause searches for instances of procurement processes that have already started executing, and the bidding process decides the instance that is to do the bidding. As a result, a rendezvous takes place with the procurement process whose execution has already started independently. Then, by means of "**new** procurement process, bid examination," an instance is created of the bid examination thread related to an instance of the procurement process. Thereafter, in " bid proposal >> procurement process, bid examination" a bid proposal message is sent. Based on the **alt** conditional branch, either contract proceedings are instituted (if an adoption notice is received) or the bid fails (if a rejection notice is received).

```
process procurement process body { // defines procurement process behavior
    thread bid examination body { // defines bid examination thread behavior
        bid proposal << bidding process; // receive bid proposal from bidding process
        act [bid proposal check];

        sync selection complete; // wait for completion of selection work

        switch { // conditional branch
            [adopted] { if bid is adopted
                Adoption notice >> bidding process; // send adoption notice
            }
            [rejected] { if bid is rejected
                Rejection notice >> bidding process; // send rejection notice
            }
        }
    }
    act [preparation for acceptance];  // internal processing
    open bid examination; start bid acceptance
    act [wait for completion];
    close bid examination;  // end bid acceptance
    act [proposal comparison and selection] wait bid examination, selection completion; // selection
        work synchronized with bid examination thread
}
```

Figure 11. Interworkflow description for procurement by public tender (procurement process body)

```
process bidding process body { // defines bidding process behavior
    act [preparation for bidding] retrieve procurement process; // internal processing
    new procurement process, bid examination; // spawn bid examination thread
    bid proposal >> procurement process, bid examination; // send proposal to thread

    alt { // conditional branch
        adoption notice << procurement process, bid examination; // when adoption notice is
        received
            act [contract proceedings];
        }
        rejection notice << procurement process, bid examination; // when rejection notice is
        received
            act [bid fails];
        }
    }
```

}Figure 12. Interworkflow description for procurement by public tender (bidding process body)

In Figure 11, the following definitions are made. In the thread statement at the beginning, the behavior of a bid examination thread is defined, and following it the behavior of the process itself is defined. After preparations are made for acceptance as internal processing of this process, the "**open** bid examination" statement declares the start of bid examination thread creation in order to start the accepting of bids. After waiting for the end of bidding, a "**close** bid examination" statement declares the stopping of bid examination thread creation. Meanwhile, the instances of bid examination threads created during this time by bidding process instances receive bid proposal messages from the instances that created the threads, and as internal processing the proposals are checked. In proposal comparison and selection, the proposals sent by all the thread instances are compared, and the winning bidder is decided. At this time the "**wait** bid examination, selection completion" clause declares a wait until all the thread instances have arrived at the "**sync** selection complete" execution. When the proposal comparison and selection are complete, the thread instance resumes execution, a **switch** conditional branch takes place, and an adoption message is sent only to the thread corresponding to the instance of the bidding process that was adopted, while rejection notices are sent to the other bidding process instances.

- Conversion to workflow process definition

Each process described using this interworkflow description language was actually converted to the corresponding workflow process definitions, which were then executed on a workflow management system.

The advantage of performing this conversion processing is that the independence of each organization can thus be respected. Workflow design takes place under a variety of constraints and circumstances peculiar to each organization. For this reason the actual contents of definitions may vary considerably even in the case of workflows for the same purpose. Even a simple workflow like that for travel expense handling, for example, may define procedures unique to an organization due to constraints relating to human resources or company rules. This can be applied also to an interworkflow. Looking at our procurement example, the interaction between the supplier companies and the procuring company must take place in accord with exactly the same rules. Yet, the specific workflow definitions in each company may well differ, so individual companies cannot be forced to follow a single set of workflow definitions.

For this reason we adopt the approach of converting from the highly abstract interworkflow descriptions to the actual definitions of each workflow process, and build the workflow on this basis. In this way the minimum necessary definitions for interaction with other workflows are generated automatically, while the rest may be changed or added to freely by the workflow designers as needed.

The conversion from descriptions in interworkflow description language to workflow process definitions can be done easily as follows. First, from a given interworkflow description a process to be converted to a workflow process definition is designated, and the behavior definition of that process is extracted. Then a workflow process definition is created with the same name as that process. Next, workflow process activities are created corresponding to each of the actions included in that definition, and are added to the workflow process definition. The workflow process activities are linked in the order determined by the control structure in the process behavior definition. An example is shown in Figure 13, where a process from Figure 12 is converted to a workflow by this kind of replacement.

After this conversion is performed, the workflow designer completes the workflow design by adding as necessary the specific contents of activity processing for each of the workflow process definitions.

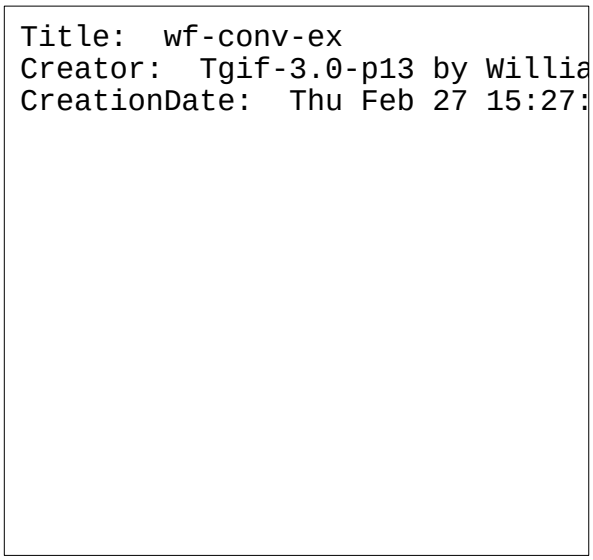


Figure 13. Conversion from interworkflow description to workflow (bidding process in procurement interworkflow)

- Summary

In this section we outlined the design policies of an interworkflow description language and looked at the language itself, then we described a method for semi-automatic conversion from the abstract descriptions in this language to actual workflow definitions.

The interworkflow description language specifications have been kept as simple as possible, and have been designed to facilitate the creation of interworkflow modeling tools. The actor model adopted for this language has a close affinity with the process logarithms of CSP (Communicating Sequential Processes[2]) and the like, and on this foundation it should be possible to build simulation tools and quantitative analysis tools quite easily. The implementation of these is left for further study.

6. Standardization trends

In this section we outline the moves toward workflow interface standardization in the Workflow Management Coalition (WfMC). Then we present an overview of the workflow interoperability interface and workflow process definition language, which are closely related to interworkflow, and note the applicability of each to interworkflow.

- Standardization efforts in the WfMC

The WfMC is promoting standards for the five types of interfaces of a workflow management system. The objectives of these five interfaces are as follows.

1. Interface 1: Process Definition Language Interface

A standard interface (Workflow Process Definition Language) is defined between process definition tools and the workflow engine. Process definitions can be interchanged also among workflow engines.

2. Interface 2: Workflow Client Application Interface

A standard interface is defined between the workflow engine and workflow client. Users benefit from being able to use the same workflow clients with different workflow engines.

3. Interface 3: Invoked Application Interface

A standard interface is defined for the applications that invoke the workflow engine.

4. Interface 4: Workflow Interoperability Interface

A communication interface between workflow engines is defined, enabling interoperability among a variety of workflow management systems existing in different organizations.

5. Interface 5: Administration & Monitoring Tools Interface

The administrative data to be recorded by the workflow engine, and the interface between the workflow engine and monitoring tools are defined.

Here we discuss the Workflow Interoperability Interface (interface 4)[6] and the Process Definition Language (Interface 1)[8], which are closely related to interworkflow.

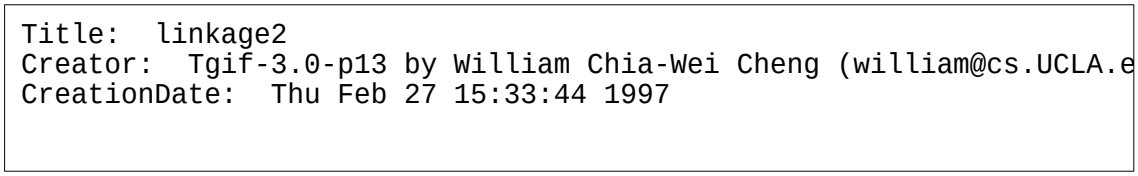
- Workflow Interoperability Interface

The WfMC is studying a workflow interoperability interface specifying the communication interfaces used with the workflow engine when a workflow is executed. The workflow interaction models and communication protocol assumed by this interface are outlined below.

- Workflow interaction models

The WfMC defines two kinds of workflow interaction models, (1) chained processes and (2) nested

subprocesses. With chained processes, one organization makes a work request to another organization, and the linkage ends when the request is completed (Figure 14 (a)). With nested subprocesses, an organization requests work by another organization and receives the results of the work activity (Figure 14 (b)).



(a)Chained processes (b)Nested subprocesses

Figure 14. Interaction models

In the WfMC workflow interaction models, essentially the transmission of data between instances occurs twice, at the start and termination of the instance, and there is no data transfer until the instance terminates (this is allowed by the communication protocol, but apparently no provision is made for this type of use). Moreover, data transfer takes place asynchronously between the sending and receiving instances.

- Communication protocol

The communication procedures for realizing the workflow interaction models defined by the WfMC can be classified into the three phases of (1) pre-processing, (2) subprocess execution, and (3) post-processing. In chained processes, however, only the first two phases occur, not post-processing. The three phases are summarized below. The flow is illustrated in Figure 15.

1. Pre-processing

In order to execute a subprocess on another work engine, (1) subprocess creation, (2) workflow relevant data transfer, and (3) subprocess starting are performed.

2. Subprocess execution

The subprocess is executed in accord with its definition. During this time, essentially there is no communication between the subprocess and the process that requested its execution.

3. Post-processing

After the subprocess terminates, (1) subprocess completion notice, (2) workflow relevant data transfer, and (3) subprocess deletion are performed.

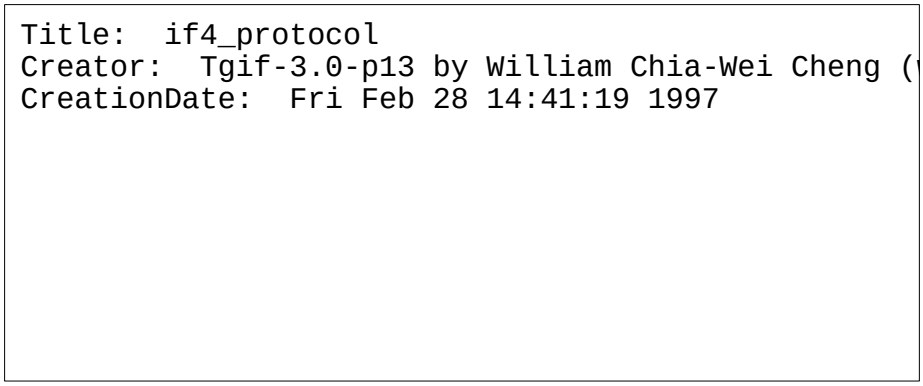


Figure 15. Protocols

- Applicability to interworkflow

Here we discuss the applicability of the workflow interoperability interface defined by the WfMC for use as a runtime interface with the interworkflow description language proposed in this paper.

The functions required by a runtime interface for our proposed interworkflow language are (1) instance creation, (2) message sending and receiving, and (3) thread spawning. Of these, only instance creation is both defined for the WfMC workflow interoperability interface and has an equivalent meaning to the function in our proposal.

The message sending and receiving function specified for interworkflow can be realized by combining the WfMC-defined message transmission with synchronization protocol. In the message sending and receiving specified for interworkflow, the receiver employs a message receive action to receive a message sent by the sender using a message send action. Accordingly, from the standpoint of the receiving side, necessary data can be obtained at the necessary time. In the message sending and receiving specified by the WfMC, the sender arbitrarily sends data at a given point in time, and the receiver cannot choose when to receive it; or else the receiver arbitrarily gets data at a certain point in time, and the sender cannot choose when to send it. The WfMC interface does, however, define a method for synchronization, so that by synchronizing the actions on the sending and receiving sides when the receiver gets data, it should be possible to implement the message sending and receiving specified in our interworkflow.

As for thread spawning, the concept of threads is not defined in the WfMC interface, so this function cannot be implemented using the WfMC workflow interoperability interface. Since not all work requires threads, however, this is not necessarily a fatal deficiency.

- Process definition language

The WfMC is studying a Workflow Process Definition Language (WPDL) as a format for process definition interchange between a process definition tool and work engine, or between different work engines.

WPDL adopts a state transition model as its programming model. Accordingly, it defines a workflow by subdividing work into workflow process activities and links them with arcs to decide their

sequential relationships.

Below we outline the language elements and then discuss how this language can be applied to interworkflow.

- Language elements

WPDL consists of workflow process activities, state transition data, workflow participant definitions, workflow application definitions, and workflow relevant data. Each of these elements is explained below.

1. Workflow process activity

A workflow process activity is a unit by which work is executed. A workflow activity may be either an atomic activity, which is the smallest unit of work, or a subprocess, which is a set of atomic activities. A workflow activity corresponds to a node in the state transition model.

2. State transition information

Transition information represents the order of workflow activities, and corresponds to the arcs in a state transition model. It describes the conditions necessary for making a state transition.

3. Workflow participant definition

This defines the people, groups, organizations or computer programs that perform the actual workflow activity instances.

4. Workflow application definition

This defines the application program invoked when a workflow activity is executed.

5. Workflow relevant data

This represents data necessary when a workflow is executed.

- Method of describing workflow across organizations

Here we discuss the method for using WPDL to define work that is performed by means of linkages among organizations. WPDL was originally devised as a format for workflow process definition interchange among different work engines. As seen, however, in the workflow participant definition element, it can also be used to define workflow performed across multiple organizations. Essentially, then, WPDL can be applied to interworkflow by designating workflow participants mapped to workflow process activities.

The current WfMC workflow interoperability model definition and execution method are noted here. If the invoking side is the parent process and the invoked side is a child process, work extending across organizations can be defined as follows.

1. Define the work (activity) at the organization level.

2. Define activities at the child process level (not atomic activities) as subprocesses.

3. Define workflow relevant data as input and output parameters.

After making the definitions, all the definitions for the parent process are distributed to the process definitions for the child processes, and these are detailed in each organization. (See Figure 16.)

The engine executing the parent process uses interface 4 to invoke the processes of other organizations when an activity is performed in other organizations.

In the workflow interoperability interface currently defined in the WfMC, work interaction with other organizations is the type by which a work instance is created anew in the other organization, and the final result of execution by that instance is received; in other words, data exchange among instances is limited to the time the instance is created and the time it is terminated. Within this scope, performing work in another organization is equivalent to a combination of instance creation and data transfer, so the present specification which defines only the control flow is sufficient.

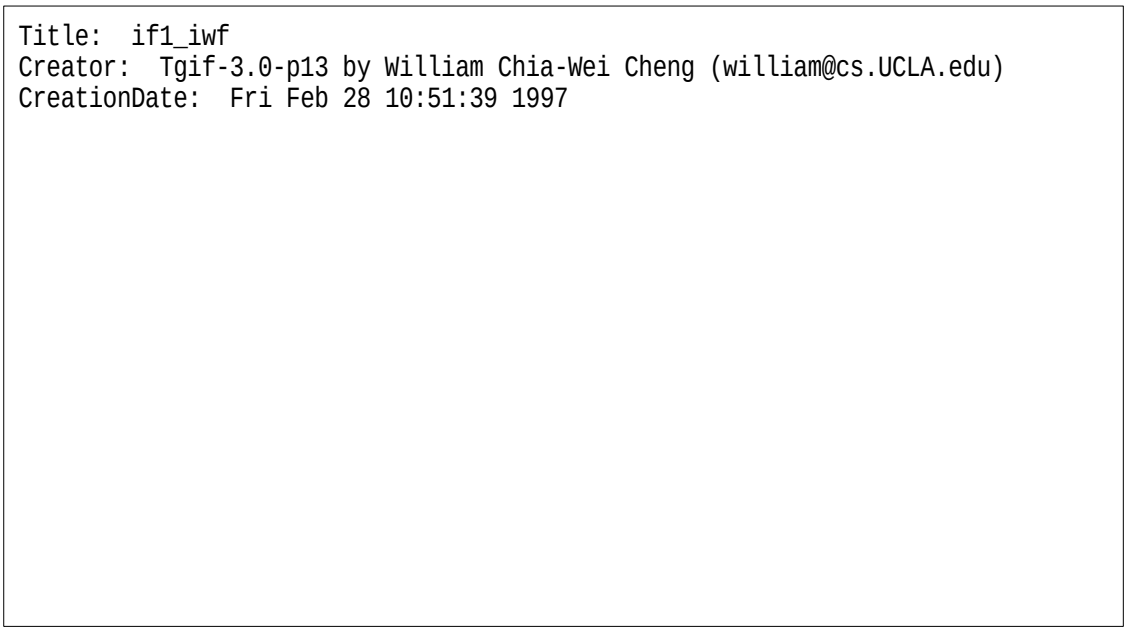


Figure 16. Definition procedure

- **Applicability to interworkflow**

Next we consider the applicability of WPDL as an interworkflow description language. The language elements required by the interworkflow language are (1) instance creation, (2) message sending and receiving, and (3) thread spawning.

As the discussion above shows, instance creation and message send/receive functions do not exist as WPDL language elements, but rather these functions take place as the result of having the workflow engine interpret a defined process. In order to describe these functions explicitly it is necessary to describe them as activity behavior, which means WPDL is inadequate as an interworkflow description language.

Thread spawning, likewise, is not specified in the workflow interoperability interface and is not

present as a language element.

For describing an interworkflow, the above three elements would need to be incorporated as language elements.

- Summary

In this section we have given a brief overview of the WfMC and discussed the workflow interoperability interface and workflow definition language being studied in the WfMC.

The applicability of these WfMC interfaces to interworkflow is as follows.

1. Workflow interoperability interface

The basic functions of instance creation and message sending and receiving are met. Thread spawning, however, is not provided for, which means the interface is not applicable to certain kinds of work.

2. Workflow definition language

Instance creation, message sending and receiving and thread spawning are missing as basic elements. Under these circumstances the language would be difficult to apply to interworkflow description.

7. Conclusions

In this paper we have presented our studies of the support functions needed for realizing interworkflow, that is, coordinated interaction among the workflow systems in different divisions or companies. We first outlined the studies undertaken in the previous fiscal year aimed at realizing interworkflow support functions, and on that basis we indicated the ideal mechanisms for supporting interworkflow. Then we presented the results to date of our studies on some of these support functions, for interworkflow modeling and for workflow design.

The following points were noted regarding the ideal interworkflow support functions. First, we classified these into three main kinds of support, for interworkflow modeling, for designing a workflow based on the overall interworkflow, and for managing workflow operations in a distributed environment. We then discussed the specific kinds of support functions required for each of these.

The following matters were discussed with regard to the support mechanisms for interworkflow modeling and for workflow design. Noting that the most important basis for support of interworkflow modeling is the provision of a description language for formally defining interworkflow, we discussed the development of such an interworkflow description language. The language adopts a description system based on the actor model, in which each of the workflows is seen as an actor. By describing rules for the order of interaction among these actors, the language defines the process of interaction among workflows. To enable more complex sequences to be described, control structures are introduced for parallel branching, conditional branching, and looping. In addition the concept of threads is introduced, for expressing interworkflow that would be difficult to describe with the actor model alone, such as the interworkflow for procurement by public bidding. In this way the interworkflow language is given the ability to describe interaction processes even in relatively complex interworkflows.

As a support mechanism for designing individual workflows in the context of interworkflow, we studied a means for automating the process of converting from the interworkflow language descriptions to individual workflow process descriptions. With this support function, activity definitions and other minimum necessary information for interaction with other workflows are generated when defining workflow processes. The burden of the workflow designer is therefore reduced, and a workflow can be designed more flexibly.

The following issues need further study. First, a processing system for the interworkflow description language and a compiler need to be developed, for converting from the language descriptions to actual workflow process definitions. Because of the close affinity between the actor model adopted for the interworkflow language and concepts such as process logarithms, it should also be possible to create interworkflow simulation tools and quantitative analysis tools on this base. In addition, more work is needed on realizing support for workflow operations management in a distributed environment, which was one of the ideal interworkflow mechanisms pointed to in our study. Methodology for implementing this support needs to be studied based on the studies to date as well as on WfMC trends.

References

[1] D. Georgakopoulos, M. Hornick, and A Sheth. An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure. *Distributed and Parallel Databases*, 3:119-153, 1995.

[2] C. Hoare. *Communicating Sequential Processes.* Prentice-Hall, 1985.

[3] Ishizuka. *Object-Oriented Programming*. ASCII Press, 1988. [in Japanese]

[4] M. Rusinkiewicz and A. Sheth. Specification and Execution of Transactional Workflows. In W.Kim ed., *Modern Database Systems: The Object Model, Interoperability and Beyond*. Addison-Weskey, 1994.

[5] J. Veijalainen. Issues in Open EDI. In *Proceedings of the Systems Integrations '92*, pp.401-412, 1992

[6] Workflow Management Coalition. *Interface1: Process Definition Interchange*, Feb. 1996.

[7] Workflow Management Coalition. *Terminology & Glossary*, June 1996

[8] Workflow Management Coalition. *Workflow Interoperability-Abstract Specification*, June 1996.