OBJECTFLOW: A Modular Workflow Management System*

Experience Report

Camilo Ocampo•, Pere Botella
cocampo@goliat.upc.es, botella@lsi.upc.es

Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
Edifici U, Pau Gargallo, 5
08028 Barcelona, Catalonia
Spain
Phone: +34 (3) 401 69 94. Fax: +34 (3) 401 70 14

December, 1996

Keywords: workflow, workflow management systems, business process, object oriented.

Abstract

Workflow Management (WM) is an emerging area that involves cross-disciplinary fields as Database, Software Engineering, Business Management, Human Coordination. A Workflow Management System (WMS) is a software tool to automate Business Processes (BPs) and coordinate people of an organization. BPs are a set of linked procedures concentrated on reaching a business goal, normally following a set of procedural rules. This work presents the OBJECTFLOW[1] project, result of the cooperation between an industrial partner, the *Centro de Cálculo de Sabadell* (CCS, a software company located near Barcelona), and the Technical University of Catalonia (UPC). The main aim of OBJECTFLOW project is constructing a flexible WMS to automate BPs of the medium and big enterprise, allowing the integration to existing software systems.

Several paradigms were used to design and develop the OBJECTFLOW WMS: from an architectural point of view, it is a Client Server application. For the analysis, design and programming of the software component, the Rumbaugh OMT methodology was used. As a WMS, it was designed to support different kinds of workflow applications, integrating legacy systems and compound documents. It is composed by four principal modules (the *Designer*, the *Workflow Engine*, the *Admintool*, and *the Client Desktop*) and others secondary components. This WMS does not incorporate the latest result in WM research but shows a successful cooperation between a research group and a Software factory, producing a commercial software tool.

*This work was supported by the CCS and the Spanish Ministry of Industry and Energy under the PEIN program 1059/95. Submitted to STEP '97.
• Partially supported by CONACyT, México.
[1] Authors know that a commercial product from Digital Equipment Corporation with the same name already exists. The naming decision for the project is CCS responsibility.

1 Introduction

Workflow Management (WM) is an emerging area that involves cross-disciplinary fields as Database, Software Engineering, Business Management, Human Coordination. It is the automation of a Business Process (BP) enabling the coordination among people, the distribution of information and documents and the integration of software applications, everything this according to a set of procedural rules.

OBJECTFLOW is a project [1] comprised into the Pan European Project (PEP, a project from the company CCS), in which an integral business management system for medium or big corporations has been created. Its main objective is the development of a Business Information System integrated and automated by workflow tools. The main aim of OBJECTFLOW is the development of a Workflow Management System (WMS) that can be totally integrated with the existing software applications of a corporation.

This project has been the result of a cooperation between an industrial partner, the CCS company (a software factory located near Barcelona), and a research group from the department of Software in the Technical University of Catalonia. We collaborated with CCS providing experience in the development of workflow application, theory and technology that supports WMSs.

This work shows the architectural and related technological issues of the OBJECTFLOW WMS. The architecture of WMS was designed taking into account the Workflow Reference Model of the Workflow Management Coalition (WfMC) [2] and the principal modules are a Process Definition Tool (the *Designer*), a *Workflow Engine*, an Administration Tool (the *Admintool*), and a user interface (the *Client Desktop*). These modules are used to modeling, enacting, and managing Business Processes (BPs); and interacting with workflow participants, respectively. The interaction with other software application is done by a specific language (the User Procedure Language, UPL).

This document is organized as follows: Section 2 presents a background of the WM and brief explanation of OBJECTFLOW concepts. Section 3 describes the architecture of the WMS. Section 4 presents conclusions. Appendix A shows a case of study where the different tools are applied.

2 Background

As defined by the WfMC [2], a workflow is “the automation of a BP, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules.” We understand a BP as a set of linked procedures focused to reach a business goal embedded in an organizational structure. In a BP, the work assignment follows rules which may be less or more complicated depending on the organization and the process involved, these are called *business rules*.

A WMS is a system that support the BP automation. A WMS normally has four software components used to define, to create, to manage and to enact the workflow, furthermore it may have an interface with the workflow participants and an interface with other tools and applications. In OBJECTFLOW, the component that manages the workflow enactment is called *Workflow Engine*. The *Designer* is the tool used to define and create workflow maps from BPs. The *Admintool* is used to manage the workflow. The *Client Desktop* is the interface between the OBJECTFLOW WMS and the workflow agents.

The modeling of Information System has focused on analyzing data flows and transformation, one issue of the success of WMS is that it allows to model data flows and transformation extending computer use beyond transaction processing into human communication and coordination [3].

Workflow Management is a discipline that emerged first in industry and later as a research area. The need to improve the efficiency in the organization and the possibility to model organizations as process-centered, automating these models, have promoted the development of WMSs.

There are several successful applications reported in industry [4] and more than one hundred commercial products [5,6]. Also there are projects with both industry and research participation [7]. As a research area, there are four main fields: technology [8], conceptual modeling [9,10], organization [11] and methodology [12].

Main roots of WM come from different fields: office automation [13], software process management, manufacturing, transaction processing, organizational and management science [14]. These roots have influenced on the type of the workflow application. So, there are workflows that are imaging document based, mail based, and server based.

WMSs are well suited as tools for Business Process Reengineering because it provides a way to model, to enact, and to redesign a workflow in an organization. Permitting simulations and evaluation of BPs to detect bottlenecks, unnecessary and duplicated tasks, right sizing the organization, etc.

Under OBJECTFLOW point of view, a workflow (like databases) has a static part, the *Template*, and dynamic instances of it, *Cases*. The *Template* graphically represents a BP and can be translated to an interpretable code written in UPL. With this representation, *Cases* are created for each new occurrence of the BP. The UPL language is a Visual Basic for Application like language designed to represent a *Template* and used in other CCS applications.

A *Template* is a kind of graph, with six different types of nodes [2] (each of them represented by a graphical shape), and arcs connecting them. The most important node is the *Responsibility*, which has an associated role and task. Other nodes are: *Boolean Condition* (a binary decision step), *General Condition* (a multiple decision step, like a `case` statement in programming languages), *Device*, *Begin* (initial symbol), and *End* (final symbol).

Arcs are called *Links* and there are four kinds of them: *Normal*, the node at the beginning of the *Link* must finish so that next node can start; *And*, all previous nodes must finish to start the next one; *Or*, at least one node must finish so the next one can start; *Exclusive Or Link*, all previous nodes must finish so the next one can start. See Figure 5 for a graphical description.

Each *Responsibility* is viewed as a duty a worker must take, performing a *Task* (sometimes supported by software applications) and handling documents. *Tasks* are composed by a set of activities. The worker who wants to take the *Responsibility* must accomplish the specified role or can be assigned directly to the *Responsibility* at design time.

Each node has associated events [17], as well as other events are produced when a *Template* is activated (users can generate instances of it) or deactivated (the *Template* cannot be longer instanced). Events are also generated when a *Case* is created, suspended or resumed. The user can assign UPL code to the occurrence of whatever of these events. Node events depend on the kind of node, e.g., *Responsibility* node and its associated task have the following events: initiate, abort, suspend, resume, dismiss, forward and finish.

A worker can accomplish several roles in the organization, e.g., a worker can play the *secretary* and the *chief* roles, because he/she has the appropriated profile of rights and obligations in the company. In the same way, a role can be fulfilled by more than one worker, e.g., the *clerk* role in a bank. It is important to notice that a worker either can be a human or an Information System (IS), in this case, the *Responsibility* is directly assigned by the BP analyst to an IS, e.g., a payroll system.

3 Architecture

OBJECTFLOW is a server based WMS, it was designed using the Client Server paradigm, so there are components installed in one or more computers providing services to a set of client components located in computers of users.

The general component model is shown in Figure 1, it is basically constituted by *a Workflow Server* (containing the *Engine*), a Documental Database, a Repository and workflow clients with user interface, administrator and designer tools.

The OBJECTFLOW WMS architecture considers that the server contains all data related to the workflow (mainly the dynamic data, the repository of information), and the code used by the *Engine* to route the work where it corresponds. Thus, work routing decisions are taken at server site and all requests to start applications come from the server. Client nodes only contain needed software to receive/send requests to the server, and the interface to start applications together with the *Client Desktop* module.

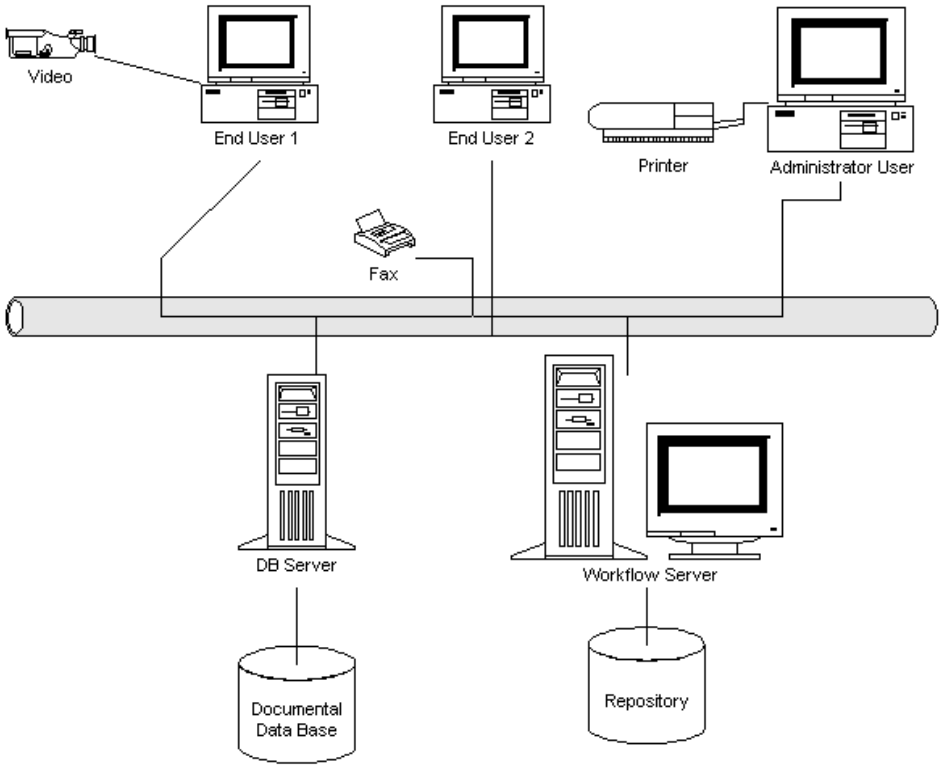

Figure 1. General Architectural Model

OBJECTFLOW WMS architectural components as are shown in Figure 2, are divided into two layers:

- Bottom level components, comprising all basic objects dealing with communications, dispatching messages, data interaction, or work routing.
- High level components, containing the user interface and providing main functionalities of the system.

The communication among all components is based on the Object Oriented paradigm using OLE 2.0. The first layer contains high level components which can interact with low level ones. High level components are:

- **Client Desktop.** It is the interface between OBJECTFLOW WMS and the end user. From this tool, the user may launch other components or to make queries to the workflow system.

- **Designer.** It is used to model BPs as workflow maps (called *Templates*).
- **Reporter.** It is the client tool employed to obtain reports of the OBJECTFLOW system status together with graphical outputs of the performance analysis or user desired statistics.
- **Simulator.** It is the client tool used to perform visualization of workloads and simulation of modification of any parameter of the workflow map.

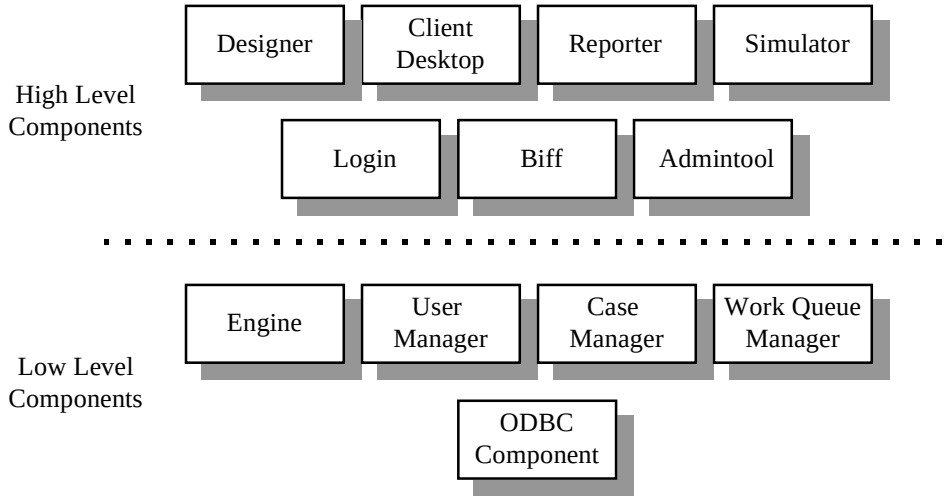

Figure 2. OBJECTFLOW WMS Architectural Components

- ***Login*.** It is the service to be provided as a prerequisite for user connections into the OBJECTFLOW WMS.
- ***Biff***. It is the tool used to notify the user a new task has arrived to his/her *Work Queue*.
- **Admintool.** It is the system administrator tool. It is executed from the *Client Desktop* and allows the responsible of the system to perform system configuration, maintenance on user databases.

The second layer contains all OBJECTFLOW WMS low level components that provide the service to the high level components. These basic components are:

- **Engine.** Core of the OBJECTFLOW WMS. This component is a container of the other following objects: *RPC Client and Server***,** *Dispatcher***,** *Scheduler***,** and *Interpreter***.**
- **User Manager.** It is the component that deals with all management of the OBJECTFLOW users, the organization employees.
- **Case Manager.** It is the component used to work with the different *Cases*. It resides in the server.
- **Security Component.** It is the component used to maintain workflow data security.
- **Template Manager.** It is the component employed to work with the different activated workflow maps. It resides in the server.
- ***Work Queue* Manager.** It is the component used to work with the different user *Work Queue*s. It resides in the server.
- **ODBC Component.** It is the component employed to interface the existing databases to OBJECTFLOW WMS. ODBC is used by the OBJECTFLOW managers to

access workflow data. Also, it is the mean by which PEP applications interact with its data.

3.1 Workflow Server

The *Workflow Server* is the responsible to enact processes modeled as *Templates*, and to interact with workflow clients. Figure 3 shows *Workflow Server* main modules.

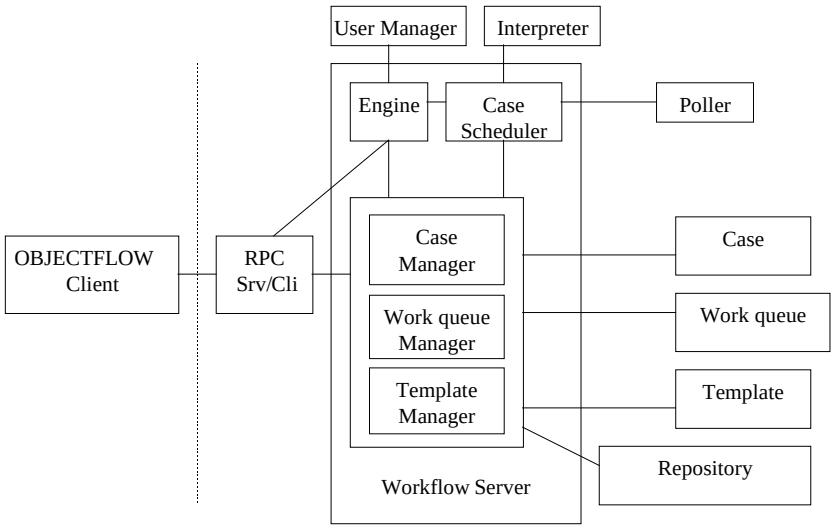

Figure 3. *Workflow Engine* components

Interaction with *Templates*, *Cases*, and *Work Queue*s are made through Managers, which are, not only responsible to attend queries related with them, but also to control them, e.g., when stopping all the *Cases* related with a *Template*. The *Workflow Server* coordinates OBJECTFLOW WMS modules, schedules tasks, activates remote tasks, and manages *Templates*, *Cases* and *Work Queue*s.

3.1.1 Workflow Engine

It is the most important component of the *Workflow Server*. It is the responsible to attend and to respond requests. It also manages and executes UPL processes and programs. The *Workflow Engine* is composed by the following modules:

RPC Client and Server. The Remote Procedure Call (RPC) client is used to send requests (commands, messages) to the RPC server, which receives and service all received requests. RPC objects are the basic mechanism for communication in OBJECTFLOW WMS. Servers and clients have both objects.

Dispatcher. It is the server component that it schedules requests received and distributes all messages coming from activated *Templates*, e.g., a worker has finished the assigned task.

Case Scheduler. This module is the responsible to ensure the execution of the different *Cases* being interpreted, detecting event occurrences.

Interpreter. It is the component that interprets a given compiled UPL program. It runs executing different routines of the program depending on the requests to be serviced.

3.1.2 Managers

Template Manager. It is responsible to manage queries about *Templates* and control commands related with groups of them.

Case Manager. It is responsible to manage queries about *Cases* and control commands related with groups of them.

***Work Queue* Manager.** It manages the information about the work to be done by each user in its workplace. When a user is assigned to a task to do or it take the related *Responsibility*, in run-time, an entry, is added to his *Work Queue*. Each user has his/her own *Work Queue*.

User Manager. It is the responsible to maintain information related with users, to attend queries to the database, to provide support for security purposes. Since the workers, from the workflow point of view, are seen as a list of users, groups, or by roles in a hierarchy, the database is structured to provide all this information. In fact, it is thought to expand UNIX control over users.

Document Manager. This module is responsible for managing all documents flowing through *Templates*, supporting distributed and heterogeneous environments. It deals with compound documents, e.g., documents with text, pictures, videos, and sound.

3.1.3 ODBC Component

Since OBJECTFLOW WMS automates information flow in the company, it has to access heterogeneous databases in a distributed environment. To deal with this, CCS applications use the ODBC API, because it is an open, vendor neutral API that enables access to this kind of databases.

But, although this interface is independent of the Operating System, the existing implementations of ODBC are on the Microsoft Windows operating system. Therefore, both, the workflow clients, and the workflow Windows NT servers, can take benefit of this. The problem arises in UNIX servers: the workflow server must access, not only the workflow repository, which could be local, but the complex database environment where the enterprise information resides. Taking into account the benefits obtained, mainly in development time, by using the ODBC approach, CCS has developed UNIX ODBC drivers, for the main DBMS, and of course the ones used in CCS.

With this approach, neither the workflow clients, nor the servers are required to know where the database is located, what kind of communication method is used to get the database, or what DBMS is being accessed. The existence of a common language interface allows the workflow developer to avoid conditional code like the above. There can be one common code line which can be used to access multiple databases as long as a driver exists for that database. Summarizing, by using the UNIX drivers for ODBC developed by CCS, the already explained benefits obtained by using ODBC drivers, are extended to UNIX applications, in this case, the workflow server.

3.2 Client Modules

3.2.1 Designer

Designer is the component used to model BP, providing a Graphical User Interface (GUI) which facilitates the description of *Templates* and the intuitive understanding of designs [18]. The output is a binary file or UPL code, interpretable by the workflow engine. The design environment provides all kind of graphical tools to improve productivity. Figure 4 shows the graphical interface of this module.

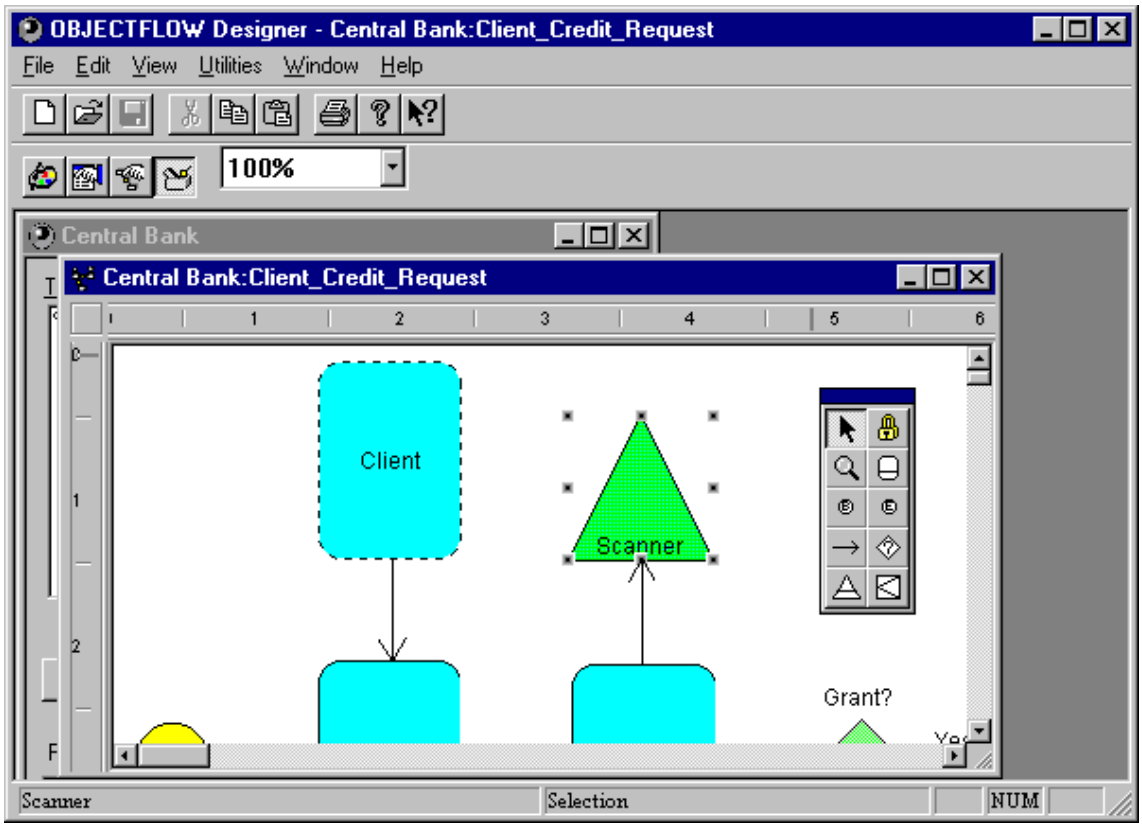

Figure 4. *Designer*

A graphical language is provided with the purpose to model a BP. This language is shown in Figure 5. All elements participating in a workflow can be modeled with the language. The user can translate the graphical representation of *Templates* in UPL code using a module of the *Designer*: the *UPL Compiler*, that generates the corresponding UPL representation that can be interpreted by the workflow server to enact and manages *Cases*. Appendix A shows a *Template* of a BP process.

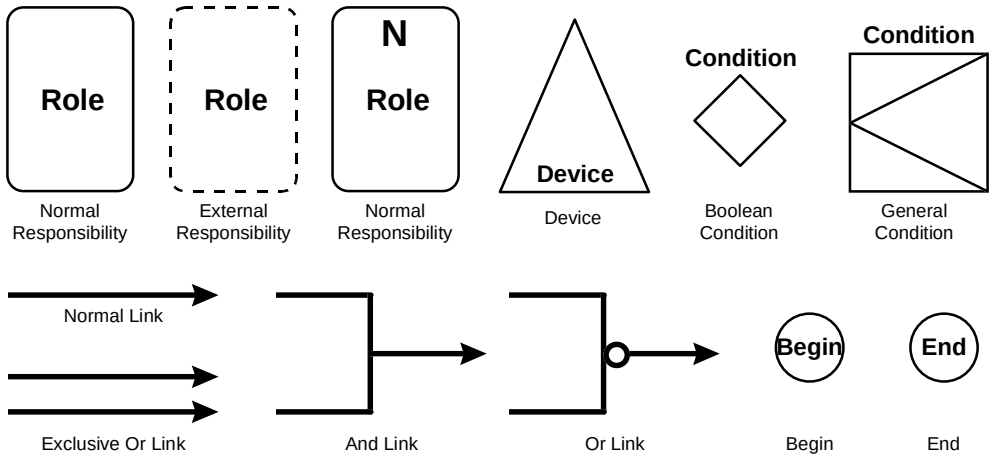

Figure 5. Graphical nodes

Definition of nodes in the *Designer* is done at design time, then a graphical interface is displayed and some information regarding the node must be specified by the user. Required information varies depending on the semantics of the node, i.e., information required to specify a condition is far different from the information required to define a *Responsibility*. The informa-

tion defined by means of the *Designer* (using its different components) is stored in the server as a repository of information.

3.2.2 Admintool

It is the tool used to administrate the OBJECTFLOW system, used by the system administrator. By means of a graphical user interface, as it is shown in Figure 7, it allows the system administrator: to perform system configuration, to maintain user's database, to monitor system performance, and to account different parameters. *Admintool* runs from the client, although a command line interface is also offered at the server site. This object holds a set of monitoring and accounting operations. Also, it is the interface to the server commands.

Admintool is used to maintain the OBJECTFLOW user's database. Also it has a set of operations to bring the administrator control running *Cases*, to display all activated *Templates* (*Templates* compiled in the engine database and ready to be instantiated).

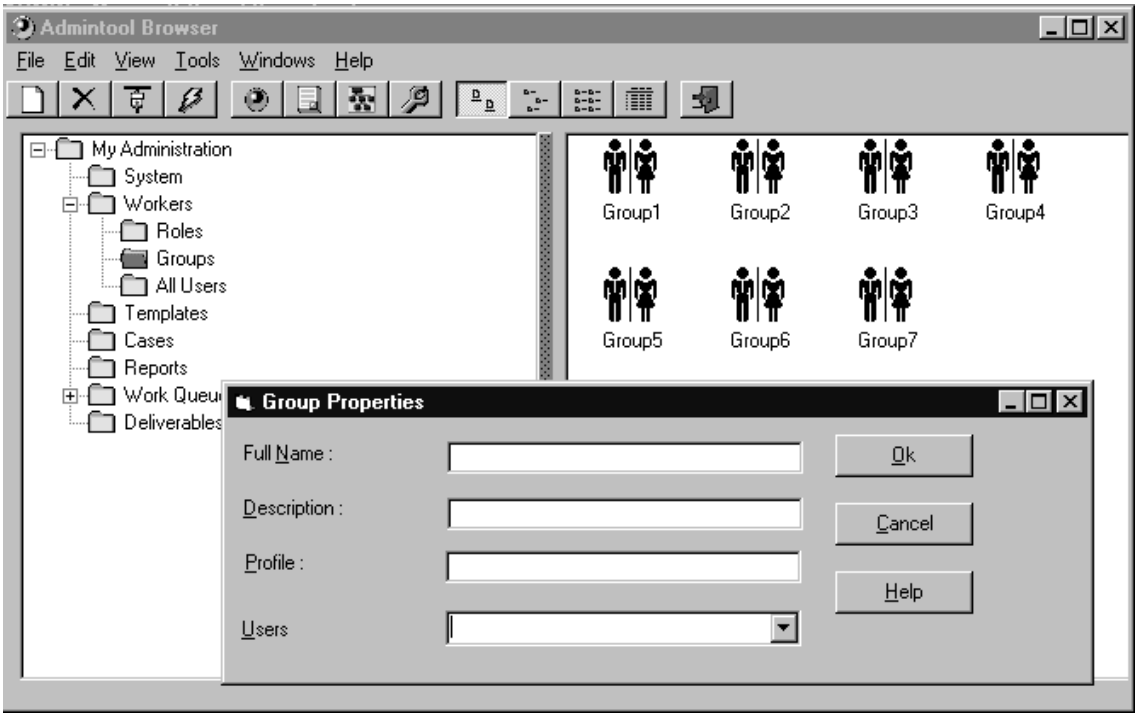

Figure 6. ***Admintool***

The *Admintool* allows the system administrator to create statistics on the system performance. Calling the *Reporter* component, a system statistic report is produced. It also permits the maintenance on the standard definition of reports (to be stored on the server).

3.2.3 Client Desktop

It is the OBJECTFLOW WMS front-end, it allows the user to perform main OBJECTFLOW tasks and launch all high level components. It is used to display and manage different OBJECTFLOW objects, always related to the logged user: *Templates* stored in the client computer as files, compiled *Templates* located in the *Workflow Server*, *Cases* instances, statistical reports, *Work Queue*, and pending tasks. The user can create, can delete or can update any of these objects. The graphical interface of this module is shown in Figure 7.

Work Queue folder of the user contains those *Responsibilities* still not taken by any user, having a role which the user accomplishes. The pending task folder has those *Responsibilities* which the user has taken but the task still has not been started.

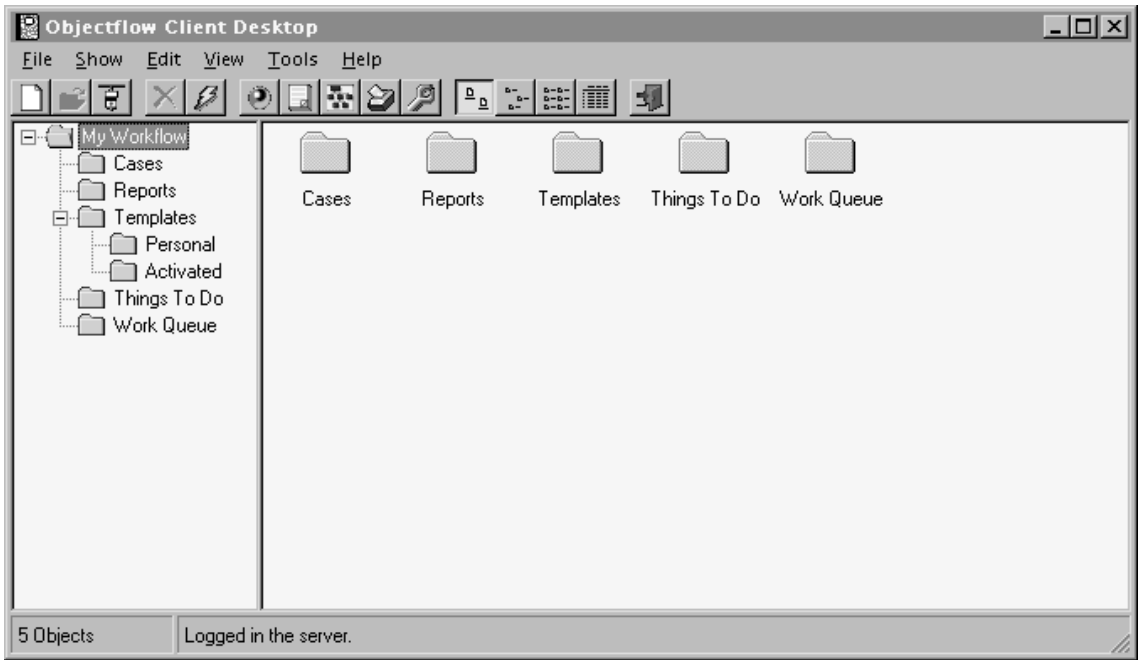

Figure 7. ***Client Desktop***

3.2.4 Biff

It is responsible for the notification to the user of a new work item arriving to his/her *Work Queue*. Also, it allows the execution of the corresponding task associated to the work item.

3.2.5 Simulator

The *Simulator* tool allows the user to dynamically analyze the workflow template for optimizing the flow of tasks associated to a given *Responsibility*. In this way it is used to view and to analyze running cases over a given *Template*. It is not a core tool in the workflow system but is a great help in dimensioning the workflows of an organization.

The *Simulator* is used to display the workloads of an activated *Template*, showing its instantiated *Cases* and the nodes where the work is being carried out. Besides from the visualization aspect, storing of experiment values can be done helping the user in the sizing of the system.

Any parameter of the *Template* can be modified and after running a simulation. Starting from the given values, or having the possibility to enter new ones, the simulator tool starts and experiment in which a total simulation of workloads, *Cases*, and servicing tasks is done.

3.2.6 Login

It allows the user to enter into OBJECTFLOW system. The *Login* component is called either directly from the user or from any of the OBJECTFLOW components in the first access to the system. *Login* shares user and password information with other PEP components. Its graphical interface is shown in Figure 8.

3.2.7 Reporter

This component is responsible for the generation of reports and the printing of the OBJECTFLOW information. Several reports have been defined in the project according to the areas of interest, i.e., statistics, system analysis report, BP report.

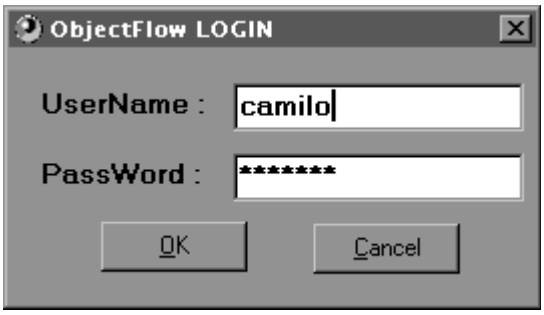

Figure 8. ***Login***

3.3 Information flow

The OBJECTFLOW WMS basic information flow can be globally described in the Figure 9, in which several basic scenarios can be specified:

Connection. The user enters into the system by either calling the *Login* component or by starting any other OBJECTFLOW client application, e.g., the *Admintool*, the *Designer*, the *Client Desktop*. In the case the user calls directly any of the client components, the system validates that the *Login* component has been run and it launches it if the *Login* session was not established.

To launch applications. It can be done directly from the user environment. The *Work Queue* can be seen from either the *Client Desktop* or the *Biff* facility. The *Biff* facility has been created in case the user is not running the *Client Desktop* and a new work item arrives.

To start *Cases*. It can be done from the *Client Desktop* by selecting the appropriate method of *Template* objects, or by running an application which has the capability to communicate with OBJECTFLOW WMS.

To create *Template*. It is done by running the *Designer*, which can be launched directly from the system or through the *Client Desktop*.

Reports on the system performance or *Case*'s evolution. This can be done either by executing the *Reporter*, which can be launched directly from the system, or through the *Client Desktop*.

Snapshot of the actual system or simulating a given scenario. This can be done either by executing the *Simulator* which can be launched directly from the system, or through the *Client Desktop*.

All this basic scenarios imply that a basic interaction is given among all OBJECTFLOW WMS high level components. On the other hand, obtaining information or deciding on the workflow is performed through the *Engine* and the help of its main managers: *User*, *Work Queue*, *Case* and *Template*. These basic components allow the engine to control the dynamic information on the system in a secure manner. The proper access to the information is directly done by using the ODBC component.

4 Conclusions

In this paper we present the architecture of a WMS: the OBJECTFLOW WMS. The resulting project was the fruit of the cooperation between CCS and the Software Department of the UPC, for developing a commercial software tool, integrable to the line of business products of this company. This has been, in our opinion, an interesting and fruitful experiment on technology transfer, since the innovative work and the best practices have been applied (transferred), as has

been said, to a commercial product. In some sense, we see that as the main contribution of our work.

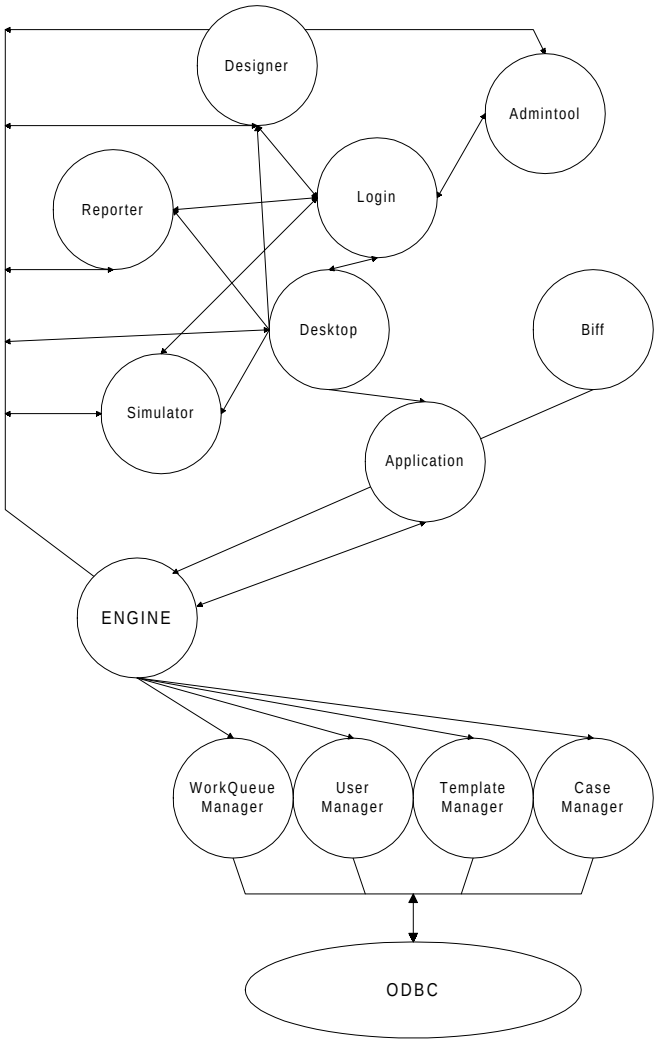

Figure 9. OBJECTFLOW WMS Information flow.

OBJECTFLOW WMS allows users to model, manage and enact BPs of their organizations, besides providing simulation capabilities to help redesign of such BPs. These functionalities are provided by different tools and services, designed and implemented as OLE objects enabling an easy integration with other software components.

At the moment, the following high level components are finished: *Designer*, *Client Desktop*, *Login*, *Admintool*. The following low level components are already finished: *ODBC Component*. Partially finished (with a reduced functionality) low level components are: *Engine*, *Case* and *Work Queue Manager*. *Biff*, Reporter and *Simulator* are not already finished.

Acknowledgments

Authors are very grateful to CCS staff: Joan Canal, J.M. Espejo, J.R. Freixanet and J.L. Albero for their cooperation and participation in the development of the OBJECTFLOW project. We also thank M. Barceló, X. Burgués, C. Guberna and J. Carpintero for the UPC part. Finally we thank S. Silva for their helpful commentaries to this work.

References

1. J.M. Espejo, H. Mateo, J. Canal, P. Botella, and M. Barcelo. Objectflow global architectural specifications. Internal Report specs/270595-1, CCS, Barberá del Vallès, Catalonia, April 1995.
2. The Workflow Management Coalition. Terminology & glossary, issue 2.0. Technical Report WFMC-TC-1011, June 1996.
3. B. Curtis, M.I. Kellner, and J. Over. Process Modeling. *Comunications of the ACM*. 35(9): 75-90. September 1992.
4. T. White and L. Fischer. *New Tools for New Times: The Workflow Paradigm*. Future Strategies Inc., Book Division, Alameda, CA, 1 edition, March 1994.
5. T. McCusker. Workflow takes on the enterprise. *DATAMATION*, 39(23):88, 90-92, December 1, 1993.
6. T.A. May. Know your work-flow tools. BYTE, 14(1):103-104,106,108, July 1994.
7. C. Mohan, G. Alonso and M. Kamath. Exotica: A research perspective on workflow management systems. *Data Engineering Bulletin*, 18(1), March 1995.
8. G. Alonso and H. Schek. Research issues in large workflow management systems. In A. Sheth, editor, *Proceedings NFS Workshop on Workflow and Process Automation in Information Systems*, pages 126-132. University of Georgia, 1996.
9. F. Casati and S. Ceri and B. Pernici and G. Pozzi. Conceptual modeling of workflows. *Lecture Notes in Computer Science*, 1021(9), September 1995.
10. T. Winograd and F. Flores. *Understanding Computers and Cognition: A New Foundation for Design*. Ablex Publishing Corporation, 335 Chesnutt Street, Norwood, New Jersey, 1986.
11. P.J. Denning and R. Medina-Mora. In *New Tools for New Times: The Workflow Paradigm*, *Case study: George Mason University*, pages 235-251. Future Strategies Inc., Book Division, Alameda, CA, 1 edition, March 1994.
12. S. Joosten. Trigger modelling for workflow analisys. In G. Chroust and A. Benczur, editors, *Proceedings CON '94: Workflow Management, Challenges, Paradigms and Products*, pages 236-247, Oldenburg, Munich, October 1994.
13. C.A. Ellis and G.J. Nutt. Office information systems and computer science. *ACM Computing Surveys*, 12(1), March 1980.
14. T. Davenport. *Process Innovation: Reengineering Work Through Information Technology*. Harvard Bussiness School Press, Boston, Mass., 1993.
15. Concordium Software Ltd. *The Workflow Application Classification Scheme*. 1995.
16. Khoshafian and M. Buckiewicz. *Introduction to Groupware, workflow, and Workgroup Computing*. John Wiley & Sons, Inc., New York, New York, 1995.
17. J.M. Espejo, J. Canal, J.R. Freixanet, and J.F. Carpintero. Upl language specification. Internal Report spec/upl/230695-10, CCS, Barberá del Vallès, Catalonia, June 1996.
18. J.M. Espejo, H. Mateo, J.R. Freixanet, C. Ocampo, C. Guberna, M. Barceló and X. Burguès. OBJECTFLOW Designer Technical Specification. Internal Report spec/230695-1, CCS, Barberá del Vallès, Catalonia, April 1995.

Appendix A. Client Credit Request BP

The following example describes the process followed by a client who requests a credit from a bank. The corresponding UPL code generated by the tool is shown in Appendix A. The scenario begins with the client who does a credit requests to the Credit clerk. The clerk has to receive the documentation delivered from the client and registers the credit request. He delivers to the Credit Analysts the documentation needed, i.e., documents specifying data about the request (date, amount of money, name of the client, kind of client, etc.).

Next, Credit Analysts verify whether the client is a credit subject with a restriction of time: two days at most. A decision must be taken after this task has finished: Is the credit granted?. If it does, another decision must be taken: if the credit amount is greater than $500 but less than $1,000, the Credit Manager must review the credit; if it is greater than $1,000, the Bank Manager must approve or deny the credit. Otherwise the process continues and the Credit Department has to update the appropriated database and deliver to the Credit clerk and the Financial clerk the client data and the credit amount. After this step, parallel tasks must be done: The Credit clerk must prepare the document which will be given to the client, and the Financial Functionary must give the money check to the client.

After these tasks have been done, the client finally receives the credit and the workflow finishes.

If the credit is not granted, the Credit Functionary has the task of notifying the client the situation, and then the workflow finishes.

The corresponding *Template* to this BP is shown in Figure 10.

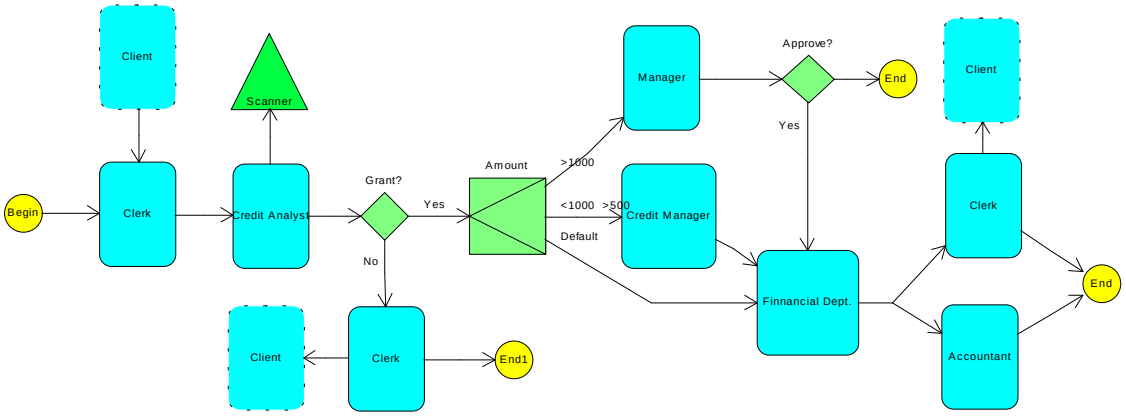

Figure 10: The Client Credit Request *Template*.

Following we show part of the corresponding UPL code generated when the *Template* is compiled.

```
Template included in project:
         Central Bank
' Template contained in file:
         C:\Workflows\Central.p
rj
' Template created:
                    Sunday,
November 24, 1996
' Template last modification
made:               Monday,
November 25, 1996
Template Client Credit Request
Const True = 1
Const False = 0
Const Default = 0

' Variables of the template
Dim counter As Integer
Dim day As Date

' Functions of the template
Sub Inc
  count = count + 1
End Sub
```

```
Sub Init
  counter = 0
  day = $today()
End Sub

' Internal Functions of the
template
Sub  ofte69960()
  Init()
End Sub

Sub  ofbe69960()
  Inc()
End Sub

' Functions of Boolean
Conditions of the template
Function  EvalGrant?() As
Integer
  If  Then
    EvalGrant? = True
  Else
    EvalGrant? = False
  End If
End Function
```

```
Function  EvalApprove?() As
Integer
  If  Then
    EvalApprove? = True
  Else
    EvalApprove? = False
  End If
End Function

' Functions of General
Conditions of the template
Function EvalAmount() As Integer
  Select Case
  Case  >1000
    EvalAmount = 1
  Case  <1000 & >500
    EvalAmount = 2
  Case Else
    EvalAmount = Default
  End Select
End Function

' Responsibilities of the
template
```

```
Responsibility  Responsibility1
  Source BEGINITEM,
Responsibility3
  Role  Clerk
  Task  Task1
  Target  Responsibility2
End Responsibility
                .
                .
                .
Responsibility  Responsibility12
  Source Responsibility9
  Role  Client
  Task  Task12
End Responsibility

' Tasks of the template
Task  Task1
End Task
                .
                .
                .
Task  Task12
End Task

' Boolean Conditions of the
template
BoolCondition  Grant?
  Source Responsibility2
  Target (True) Amount
  Target(False) Responsibility9
End BoolCondition

BoolCondition  Approve?
  Source Responsibility4
  Target (True) Responsibility6
  Target(False) ENDITEM
End BoolCondition

' General Conditions of the
template
GenCondition  Amount
  Source Grant?
  Target(1) Responsibility4
  Target(2) Responsibility5
  Target(Default)
Responsibility10
End GenCondition

' Devices of the template
Device  Scanner("Clerk's
scanner","Workstation1","scanner
")

' Begin item of the template
BeginItem  Responsibility1

' End item of the template
EndItem  Approve?,
Responsibility7,
Responsibility8, Responsibility9

' Template events of the
template
Template.Activate = ofte69960()

' Begin events of the template
BeginItem.Init = ofbe69969()

' GeneralCondition events
Amount.Evaluate = EvalAmount()

' BooleanCondition events
Grant?.Evaluate = EvalGrant?()
Approve?.Evaluate =
EvalApprove?()

End Template
```