An Architecture for Supporting Disconnected Operation in Workflow: An XML-Based Approach

Pedro Arroyo-Sandoval, Ana I. Martinez-Garcia, and Cristina Ramirez-Fernandez

Centro de Investigacion Cientifica y de Educacion Superior de Ensenada
Km. 107 Carretera Tijuana-Ensenada
Ensenada, B.C., Mexico.
{parroyo,martinea,cramirez}@cicese.mx

Abstract. Nowadays, there exist few options to support workflow and cooperation among mobile users using non-traditional computing platforms such as *Personal Digital Assistants* (PDAs). Considering the growing need for mobility and the more frequent use of mobile devices, it is necessary to provide support for the integration of these to the work environments. In this work, we present the development of the SysCoor *Workflow Management System* (WFMS) architecture, which provides support for the semi-automatic generation of Web-based workflow systems and disconnected workflow through the use of models described in the eXtensible Markup Language (XML) and also considers the use of image, video and multimedia presentations as part of the workflow information entities. We present a scenario that illustrates a disconnected process and how it could benefit by this approach. We establish the requirements for supporting disconnected workflow and discuss the architecture of the system that is based in the use of XML for the specification of the workflow systems and disconnected operations. The workflow systems' specification is generated in XML to enable interoperability with other WFMS engines. The development of the disconnection mechanism is presented in terms of a lightweight architecture for PDA devices developed using the SuperWaba programming language, however, the architecture described can be implemented using other programming tools.

1 Introduction

A Workflow Management System (WFMS) is a system that automates flows of information and control in a business process by managing the sequence of work activities within the process and the invocation of human or/and Information Technology (IT) resources [1]. Thus, agents (people involved) in a process within an organization focus only in the execution of their tasks, letting the WFMS handle the flow and control of the work.

Traditional workflow systems have been developed under architectural and design considerations that propose the implementation of client-server architectures. Under this approach, the workflow system definition and control, is kept in a centralized structure [2]. This approach requires users to keep a continuous connection to the

J. Favela and D. Decouchant (Eds.): CRIWG 2003, LNCS 2806, pp. 277–292, 2003.

network, where the workflow systems reside, if they want to participate in the coordinated execution of their work. This makes the control of the workflow activities easier with the tradeoff of reduced possibilities for mobility.

Newer approaches like the Web-based systems have increased the accessibility of those workflow applications to geographical distributed users [3]. Thus, augmenting the possibilities for users to be distributed along several geographical places and still continue with the coordinated execution of their tasks. However, this approach still requires the permanent existence of a network connection to a central server in order to participate in a workflow activity.

In contrast, organizations are now moving forward to environments where the mobility is required. In recent years we have seen the spread of computer technologies beyond the traditional desktop environment [4] towards the use of new technologies like Personal Digital Assistants (PDAs). However, few options exist to support the workflow among mobile agents using non-traditional computing platforms such as smart phones and PDAs [3]. It is important to point out that mobility should not be a limitation to keep the regular flows of work nor to be a barrier on organizational collaboration, and neither to break with concepts like workgroup collaboration [5].

Considering the more frequent use that organizations make of mobile devices, it is necessary to provide support for the integration of those devices to the work environments. These kinds of devices observe special characteristics like intermittent network connections and limited storage space. Due to the first characteristic, we need to look for options that enable process agents to continue with their work and keep coordinated interactions in their work environments in cases where network connections are not available.

Due to its nature, disconnected operation, defined as the disconnection from services provided by a server [6], appears as the direct option to support mobile workflow agents. The disconnected operation is quite useful in environments where network connections exist intermittently or do not exist at all. This mode of operation enables users to continue with their work, even when they are not physically connected to the network where they normally perform their tasks.

To explore the potential of this approach, we have developed *SysCoor,* a WFMS which architecture provides support for the semi-automatic generation of Web-based workflow systems and also enable the generation of disconnected workflow systems making use of processes models described in the eXtensible Markup Language (XML). On the other hand, WFMS is a fast evolving technology, which is increasingly being exploited by businesses in a variety of industries. This fact has been pointed out and has given place to a new opportunity, the development of options for achieving interoperability among different WFMSs. As an example of this effort, the Workflow Management Coalition (WFMC) has proposed a reference model [1] to achieve interoperability between workflow products. This reference model is a set of interfaces and data interchange formats between such components. Also, the WFMC proposed a process definition language [7] described in XML to make feasible the migration of process definitions between tools. Therefore, we devise in the use of XML an excellent opportunity to enable interoperability among WFMSs.

In this paper, we focus on the disconnected workflow paradigm as a means of extending collaborative work among users even when a communication network is not present. At first, we discuss the concept of disconnected workflow, the work that has been done and the future work literature reports on the area. We follow by pointing out the requirements that previous work has established and the requirements we

found through the realization of two case studies. Then, we describe the *SysCoor* architecture and how it supports the disconnected workflow paradigm. We finish the paper presenting our conclusions and future work.

2 Disconnected Workflow

Disconnected operation is the ability of a client to continue his/her work operations when and where network connections are not present and therefore access to the central main data repository is not possible [8]. With the development of mechanisms to support the disconnected operations, users within an organization can work independently of the main computer facility [9]. Clearly, the combination of workflow and disconnected operation concepts, offers a wide range of application in current workflow scenarios. By combining the two concepts, we found the concept of disconnected workflow as the availability of services provided by a WFMS in a disconnected state.

Most of the efforts to support a reliable performance on disconnected operation have been made in file systems and showed that supporting disconnected operation was very effective in the fundamental level of services such as file operations [8]. But at the application level, like workflow execution, not much work has been done in depth [2, 10]. In this section, we briefly review four of the most important works that have been done to support disconnection in workflow.

The Exotica/FMDC project [2], represents the first reported research on the area of disconnected workflow. In this work, the authors propose a series of mechanisms that enable the *FlowMark* [2] workflow architecture, to support disconnected clients. One of the main contributions is the pointing out of the impact of disconnected clients on WFMS. The system proposed feasible mechanisms related with handling relevant data, like the notation of locked activities and the user's commitment to eventually execute them.

Magi [3] implements an architecture in terms of a HTTP server (micro-Apache). This implementation makes use of server-primitives to enable the management, synchronization and archiving of online information. However, *Magi* was designed to specifically address the paper-based workflow [3] and it represents a heavy-weighted in terms of the HTTP server needed to support disconnection.

The ICU/COWS architecture [10], analyzes task models using the concept of the *type* and *scope* of service when providing disconnection. By *type* they refer to the set of activities involved in a process (roles and activities), and by *scope,* the whole range of targets where the services are performed (information entities). Based on this analysis they identify four important considerations for supporting disconnected operation in WFMSs: task classification, data and application handling, and emulating task state transition. This last work [10], provides a good review of the aspects that should be generally considered to support disconnection in WFMS.

Finally, the server-based software *Domino Everyplace Enterprise* [11] provides a platform for the development of mobile applications that integrate with the workflow architecture of Domino [11]. It implements the use of XML for the definition of application. However it lacks from a global integration of tasks and does not provide evaluation of the task disconnection feasibility.

In general, the proposed future work found in the literature in the area is concerning the following challenges: develop interoperability with XML, mobility supported with software autonomous agents and concurrency control handling [10][12]. In this sense, the contribution of our proposal will be to explore the use of XML as a means of extending compatibility and integration to existing tools.

To give a clearer understanding of how the disconnected operation in workflow systems can keep the coordination and still accomplish the enactment of an organizational process, we present a workflow scenario taken from a case study that was developed in a winery company. Our scenario presents the sales process in this company. The scenario presented applies to almost any traditional sales scenario where the visits to clients are a necessity and where each link in the sales chain - the client, the salesman, the warehouse, - acts independently according to its function.

2.1 A Disconnected Workflow Scenario

Most of the time, salesmen face the need to visit clients in their locations, sometimes out of town, in order to perform a sale. These visits frequently require the salesman to be away from the office for long periods of time, which can go from a complete day to several days. During these visits, a salesman requires to collect information about costumers' product inventory and fill out invoices among other activities. To collect an invoice, he needs to have detailed information about a costumer's previous purchases and their debit and credit status in order to take a decision and perform a good sale for both parties. Some of the most important visits are those made to potential clients for the company, called promotional visits. In these, the salesman needs to carry graphical material such as pictures of the vineyards and of the production facility, and, when the customer has multimedia equipment, the salesman can also give a multimedia presentation, all of the above, with the purpose to give the client a more complete view of the company and its installations.

Normally, a salesman prepares his daily activities during the mornings; he schedules his visits and prepares all the information and graphical material he needs to carry for those visits. If during a visit a need for information rises, for example to clarify details of the costumer's credit or about product inventory in the company's warehouse, the salesman needs to call the office and ask for help to solve the issue. The salesman always relies on the availability of an office clerk to solve the problem. This situation delays the sales process, which affects the customer service quality damaging the external perception of the company.

At the end of his day, the salesman usually returns to his office with the information collected during the visits. This information is needed to schedule the next day's visits, to update customer's credit and debit records and to process product deliveries. Since the collected information is in paper, when the salesman returns to the workplace he still needs to input this information into the current information system and transcribe the invoices collected. These duties are not performed until next morning, delaying the following process tasks such as delivering and billing until the next day.

By integrating the disconnected workflow schema to the above scenario the streamline of the process can be achieved. During the mornings when a salesman prepares his visits, he uploads to the PDA the previously defined tasks related to the sale process. This includes information such as debit and credit status of the client, inventory product information and invoice forms. With the above data available, the

salesmen can proceed to solve the problems mentioned before, without having to consult the office in cases where information needs arise.

When the salesman returns to his workplace, he reconnects his PDA device and re-integrates to the workflow. He might decide to download the information collected and/or disconnect the next tasks he needs to perform. When he downloads the collected information, this information is integrated to complete the workflow of the sales process; the invoices are routed to the warehouse where the delivery is prepared and the sales information is sent to the accountant clerk to update the credit and debit client status.

Whenever is possible, the salesman can reconnect temporarily to the central server, he can download the information collected and receive the following tasks and the necessary data to perform those tasks.

The scenario goes further from a simple data collection when gives the salesman all the necessary information to take a decision to perform a sale, and when enables the salesman to reconnect and receive the following tasks and feedback from the central workflow system.

In the case of a promotional visit, the salesman could upload to the PDA the previously defined tasks related to this process and the required information which includes pictures, video and multimedia presentations. This PDA must have multimedia capabilities that will enable the salesman to give to a potential client, a complete presentation of the company, in spite of the technological resources available in the site.

In the following section, we present the necessary requirements to support disconnection in WFMS.

2.2 Requirements for Disconnected Operation

In order to develop mechanisms to support the disconnection of workflow, we identified the requirements already established in the literature on disconnected operation. Furthermore, we developed two small case studies that enabled us to observe real scenarios where people switch from a connected to a disconnected work environment. The scenario described in the previous section belongs to one of these case studies, from which we analyze the forms the work is performed in these kinds of environments. These scenarios were not technological equipped, so we had the opportunity to observe only the human aspects of work and contrast the literature requirements with the requirements obtained from the case studies.

Besides requirements obtained from literature and from our case studies, a third kind of requirements needed to be considered, we refer to the *SysCoor* architecture structure and its requirements.

Requirements from Cases Studies

a) Coordination among process agents must be kept in spite of disconnection
b) A mobile agent should be able to execute from 1 to n roles while in disconnection
c) Role execution requires an information input that must be present at the time disconnection occurs.

Requirements from the Literature Review

a) To allow clients to work on a disconnected mode, they must have their own storage and access to the necessary information to be able to proceed without consulting a centralized data structure [2].

b) Clients must have autonomy, this means, a central server must ensure there are no conflicts between the different disconnected clients [2].

c) When providing information prior to disconnection, no out-of-date input information should be used [12].

d) When reconnecting to WFMS, there must exist the handling of output information to ensure data consistency [12].

e) There must exist, the task classification in order to decide if a task can be executed in the mode of disconnection, and it should consider the task type and task relevant data [10].

f) There must exist the handling of task state, related applications and relevant task data served during disconnection [10].

Requirements from SysCoor Design

a) XML should be used to support interoperability, to enable the execution of the developed workflow models though a simple transformation of the XML definitions in other XML-based workflow machines.

b) The centralized control of the process state must be kept by a Global Process Administrator.

c) Users should be allowed to visualize the state of their processes at any time.

d) There must exist multimedia data support.

Summarizing, we classified all of the previous requirements in four main areas that need to be considered when supporting disconnection: evaluation of the task (role) disconnection feasibility [10], information hoarding previous to disconnection [2,10,12], synchronization to the workplace after disconnection [12] and control of coordination during disconnection [2].

At this point, we need to distinguish between the two existing types of disconnection [10]: involuntary and voluntary. Involuntary disconnection occurs when a remote failure impedes users to continue with their work, while voluntary disconnection happens when a user explicitly takes all her tasks offline. We limit our work only to voluntary disconnections.

In the next section, we describe the architecture of *SysCoor* and how it fulfills the four classification areas just mentioned.

3 SysCoor

SysCoor is a WFMS with an architecture intended to facilitate the generation of workflow systems through the use of process models captured using diagrammatic techniques [13] and represented in XML. In *SysCoor*, the definition of the workflow system starts with the definition of the organizational process we aim to automate. To

do so, we capture a process using Role-Activity Diagrams (RAD)[1], State Transition Diagrams (STD) and defining the information entities used in every task of the process as well as those that are exchanged during the execution of the process tasks.

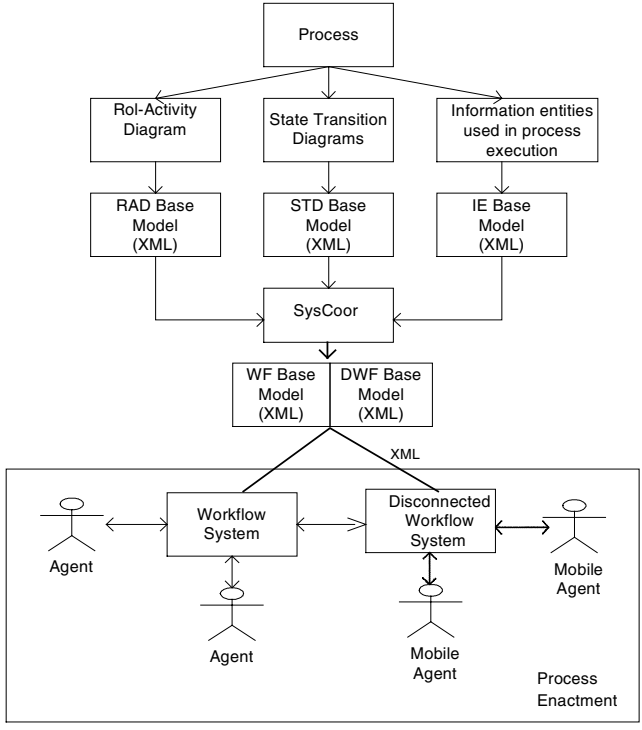

Fig. 1. Process flow for generating a Workflow System

The three models are formally represented in XML and validated using a Document Type Definition (DTD) for each model. The XML documents for the RAD, STD and information entities (IE) are known as the RAD, STD and IE base models, respectively [13]. Figure 1 describes the flow in which a workflow system is generated. First, we develop the RAD base model to identify roles, interactions, states and activities of the process. Then, the STD base model is constructed for each of the roles defined in the RAD. Finally, the information entities that participate in every process task have to be identified and properly defined (IE base model). The RAD, STD and IE base models are parsed and then transformed into a Workflow System base model (WF base model). This model defines the elements of an executable workflow system in XML. This WF base model is then used to create a new model specification called the *Disconnected Workflow base model (DWF base model).* This new model contains the special considerations for devices such as PDAs.

[1] The Role Activity Diagrams (RAD) are among the most useful modeling techniques for process representation [14]. These diagrams map or capture the tasks, objectives and people (agents) involved in a process into roles, activities, decisions and interactions [15,16].

3.1 Architecture of SysCoor

The architecture of *SysCoor* consists of five components (figure 2): a workflow system generator, a data repository to store models and workflow systems' definitions, a coordination engine, an operation disconnection module and a lightweight client architecture intended for PDA devices (A).

The workflow system generator has two sub modules: the model generator and the code generator. The model generator is responsible of reading the *RAD, STD and the IE base models*, and creates the *WF* and the *DWF base model*. The code generator creates the base role structure that includes: an activity handler, an information entity handler, an operation handler, an application handler and a state handler. Finally, the coordination engine is the responsible of maintaining the collaboration, communication, coordination and process information interchange.

Since, for the disconnected operation, our target technology is a PDA device with still limited operational resources, the client side proposal is a lightweight architecture with minimal mechanisms. This lightweight architecture is made up of four main components: a local coordinator (LC), a graphical user interface (GUI) generator (IG), a local data repository (DR) and a base role (BR) that handles all the elements that build a role such as the internal activities, states, information entities, operations involved and external applications used during execution. Motivated by the differences of GUIs between desktop computers (original target platform of *SysCoor*) and PDAs devices, we implement a *Code Translator (CT)* as part of the *Code Generator*, for making the code adjustments in the generation of the GUI when creating the *DWF base model*. The *Operation Disconnection Module* consists of three mechanisms: the *Determination of Disconnection, XML Document Transfer* and *Synchronization Mechanism*.

At this point, it is necessary to explain the approach in *SysCoor* for the generation of an executable Web-based workflow system. Based on the *WF base model*, serialized java objects are generated. These objects are later called and executed within predefined html basic page layouts that help to display a graphical user interface. Each serialized object has an html page layout associated. With the information stored within those objects, workflow functionality is achieved. These basic layouts fairly cover the most common workflow scenarios, such as, submit an information entity, wait for an answer, etc.

Due to the graphical display limitations in PDA devices, the predefined html pages cannot be used in the new disconnected schema. Due to this fact, we redefined the existing html pages layout, into traditional GUI interfaces defined through an XML document; we called it the *GUI base model* and it is general for all the *DWF base models*. This document enables interoperability among different PDAs platforms. Figure 3 shows the DTD (a) and an example of this document (b).

In contrast with the serialized objects approach, that gives functionality to the workflow systems, the *DWF base model* stays as a pure XML document. This XML document is formed with information obtained from the *GUI Base Model* and the *WF Base Model.* The *DWF base model* specification is validated with a DTD which also includes the definition of data types, data contents and relationships between data and GUI interfaces as illustrated in part of the DTD shown in figure 5.

The *DWF base model* contains sufficient information so that, once it is parsed it provides enough functionality to the definition of the disconnected operation in a workflow system.

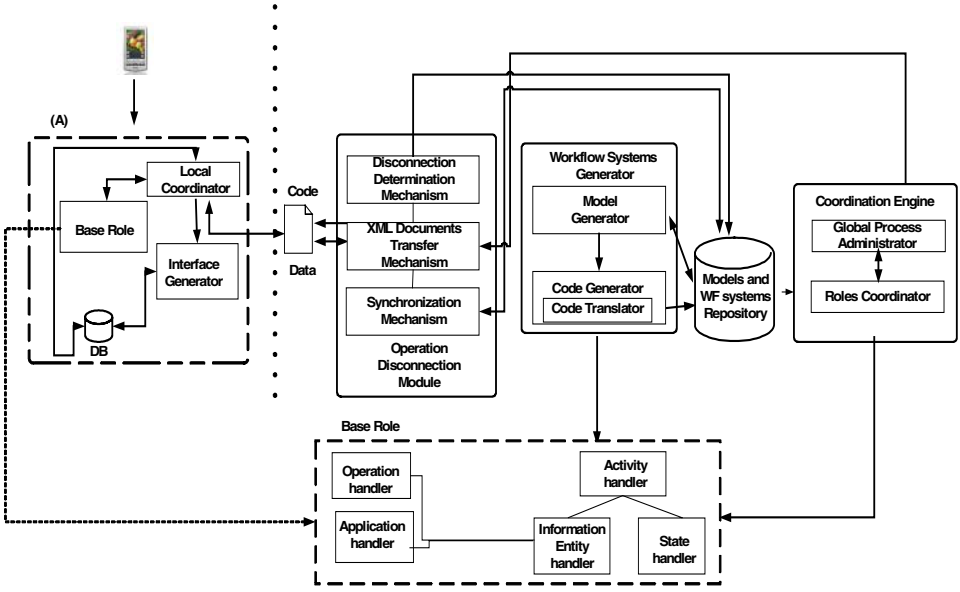

Fig. 2. SysCoor WFMS Architecture

The *DWF base model* is sent to the PDA device before disconnection occurs and, once it is stored on the local repository, it can be accessed by request of the local coordinator and parsed locally. The information obtained after parsing is processed by the handlers inside the base role and by the *GUI Interface Generator*.

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT elementgui EMPTY>
<!ATTLIST elementgui
          id_gui CDATA #REQUIRED
type(CheckBox|InputBox|ItemButton)#REQUIRED
         text CDATA #IMPLIED
         x CDATA #REQUIRED
         y CDATA #REQUIRED
         value CDATA #IMPLIED>
   <!ELEMENT act (elementgui+)>
   <!ATTLIST act
         id_act CDATA #REQUIRED
   >
   <!ELEMENT interfaz (role+)>
   <!ELEMENT role (act+)>
   <!ATTLIST role
            id_role CDATA #REQUIRED
>
```

(a)

```
<!DOCTYPE interfaz SYSTEM "GUI.dtd">
<interfaz>
   <role id_role="1">
   <act id_act="1">
   <elementgui      type="ItemButton"
text="Submit" x="10" y="5"/>
   <elementgui        type="InputBox"
text="Enter   Amount"   x="10"   y="5"
value="56"/>
   <elementgui        type="CheckBox"
text="Yes/No"      x="10"      y="5"
value="yes"/>
    </ act>
    </role>
    <role id_role="2">
       <act id_act="1">
                .
       </id_act>
   </role>
</interfaz>
```

(b)

Fig. 3. The DTD for the *GUI base model* and an example of a XML document that represents a simple input window

3.2 Evaluation of Role Disconnection Feasibility

Prior to disconnection, and after the *DWF base model* was created, the next task to be performed is to examine this base model and evaluate the activities that are candidates for disconnection; this task is executed under request of a role disconnection by a process agent. To do so, we need to know the information entities used by every task and the state (locked-unlocked, available-no-available) of these entities in a given moment. This information already exists in the data repository. Later, we will use this information to successfully execute the data hoarding prior to disconnection. Also, we need to know the interactions that exist between tasks. In other words, we must know if a task relates to other tasks within other process role. In this regard, we parse the *DWF base model* and create a data structure that holds the interactions of every task with other tasks in other roles. This data structure is queried every time a disconnection is requested.

Information about interactions among tasks, along with information about information entities used in each of those tasks and the states of the latter, help us to determine the information dependencies in a process role to be disconnected. By establishing the information dependencies, we answer questions like, *what information is received?, what information is produced?* and *what information is sent during the process execution?* Having defined what information is received is a quite important step prior to disconnection, because if this information is not available at the time the disconnection is required, the role cannot be disconnected. On the other hand, the answer to what information is sent during the process helps us in the reconnection phase, since it gives us enough detail about what information needs to be sent back to the central data repository.

The activity of retrieving and analyzing the interactions in a given role and the information entities used by the role is executed by the *Determination of Disconnection*

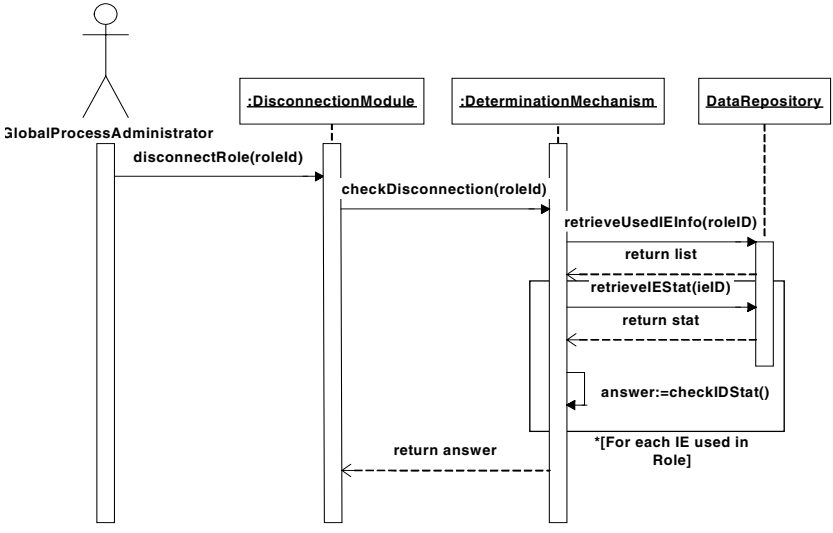

Fig. 4. Sequence diagram of the evaluation of role disconnection feasibility process

Mechanism (DM). In figure 4 we present a sequence diagram that shows the sequence in which the evaluation of role disconnection feasibility is executed. As shown, the *Global Process Administrator (GPA)* makes a request of disconnection to the *Operation Disconnection Module*; this requests the *DM* to check if a disconnection is feasible, then the *DM*, based upon the *DFW base model* information, retrieves from the data repository the state of each of all the information entities used by the role that requests disconnection. It returns an answer to the *Operation Disconnection Module,* based upon de availability of the information entities. If it is a negative answer, it sends a message to the *GPA* explaining the reason why the disconnection is not possible. If it is a positive answer, the code generator is requested to retrieve from the *DWF base model,* the definition of the tasks to be disconnected. Then, the data hoarding phase starts, in this phase, the contents of the information entities used by the role requesting disconnection and specified in the *DFW base model*, need to be retrieved and added to description of the roles that will be disconnected.

3.3 XML Data Specification

The workflow relevant data represents the information entities used during workflow execution. This data is created and used within each process instance during process execution. It is made available to activities or applications executed during the workflow and may be used to pass persistent information or intermediate results between activities and/or for evaluation in conditional expressions such as in transitions or participant assignment [7]. SysCoor includes in the *IE base model* definition of various basic data types, (including integer, string, etc.) and also documents (spreadsheets, presentations, etc). The *IE base model* also defines basic operation over basic data types, such us, addition, subtraction, division, multiplication, and also, the references to the location of the multimedia data used in the tasks.

Currently SysCoor enables users to use *Microsoft Office Documents* as a part of the information entities handled during workflow execution. These documents can be accessed and edited on a Web browser.

However, the *IE base model* only defines the structure of the relevant data but not the data itself. The current workflow relevant data is stored on the data repository. If disconnection is required we need to transfer not only the workflow logic and GUI representation, but also the information entities needed for the execution of the workflow system. We use the DWF DTD (figure 5) that helps to describe information entities, their types, their contents and their display characteristics. The *Code Translator* uses this DTD to generate the structure of the relevant data information, not the content, needed in the *DWF base model* when is generated. The data content is retrieved and added to the *DWF base model* later, when disconnection has been determined to be feasible, and only for those roles that will be disconnected. The main purpose of this new document is to extend the information stored in the *IE base model.*

In the case of *Microsoft Office Documents* currently supported on SysCoor we make an exception due to compatibility among platforms issues and we do not consider them in our initial prototype.

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT act (data+)>
<!ATTLIST act
      id_act CDATA #REQUIRED
>
<!ELEMENT data EMPTY>
<!ATTLIST data
      id CDATA #REQUIRED
      type (I | S) #REQUIRED
            value CDATA #REQUIRED
      id_gui CDATA #REQUIRED
>
<!ELEMENT datas (role+)>
<!ELEMENT role (act+)>
<!ATTLIST role
      id_role CDATA #REQUIRED
>
```

(a)

```
<!DOCTYPE datos SYSTEM "data.dtd">
<datas>
  <role id_role="1">
  <act id_act="1">
  <data id="1" type="I" value="10" id_gui="1"/>
  </act>
  <act id_act="2">
  <data id="1" type="I" value="10" id_gui=""/>
  <data id="2" type="S" value="Test" id_gui=""/>
  </act>
  </role>
  <role id_role="2">
      <act id_act="1">
        <data id="1" type="I" value="10" id_gui="1"/>
      </act>
    </role>
</datas>
```

(b)

Fig. 5. Part of the DWF DTD for the data definition and an example of part of a XML document that illustrates data specification

3.4 Data Hoarding Phase

During this phase two important activities are executed: loading and locking activities and information entities. The *Operation Disconnection Module* triggers this phase once it has received a positive answer from the *DM* signaling the feasible disconnection of a role.

The loading phase involves two steps; first, retrieve the XML Data Document and add the information about data content, required only for the activities to be disconnected, to the *DWF base model*. In the locking phase, all information transferred to the PDA device is communicated to the *GPA* that is in charge of handling the locking and unlocking of tasks and information entities. In figure 6 we present a sequence diagram that shows the data hoarding process.

As mention earlier, the *Operation Disconnection Module* consists of three mechanisms: the *Determination of Disconnection, XML Document Transfer* and *Synchronization Mechanism*. We already developed on the *Determination Mechanism,* now, we will describe the two remaining mechanism.

XML Document Transfer Mechanism (TF)
Upon request of the *Disconnection Module*, the *TF* retrieves, from the workflow systems' repository the XML document that defines the modified workflow system definition *(DWF base model)*. The *TF* then transfers this document to the *Local Coordinator (LC)* in the PDA. In the case of those tasks using multimedia data, this type of data is retrieved and sent to the PDA devices using a raw binary file transfer mode.

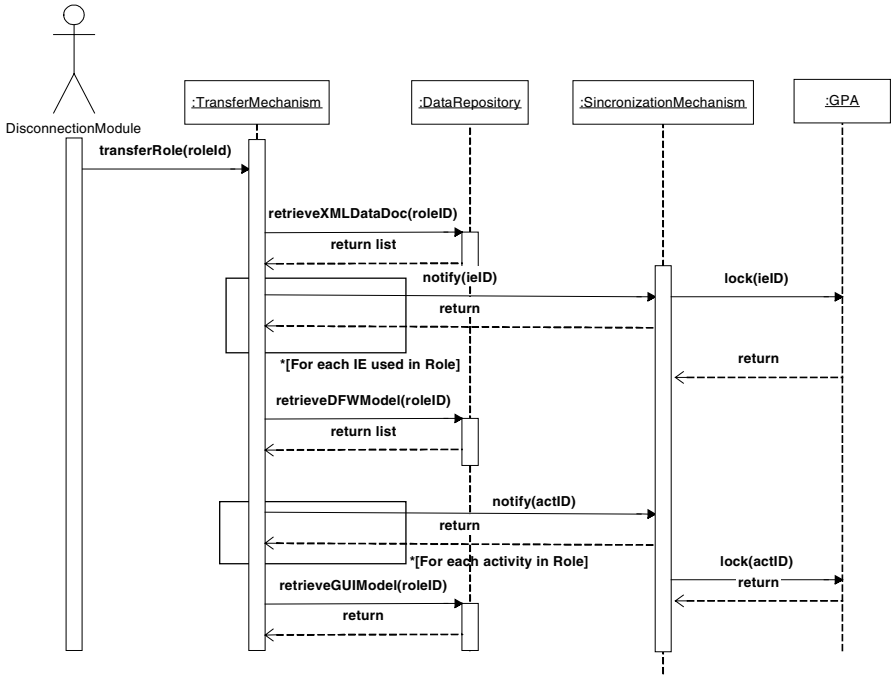

Fig. 6. Sequence diagram of the data hoarding process

The *TF* is also responsible of receiving the changes generated during disconnection. These changes are also described through the use of a XML document. It receives the documents and passes the information received to the *Synchronization Mechanism (SM)*.

Synchronization Mechanism (SM)
Basically the *SM* performs two actions that take place prior to disconnection and when disconnection ends. Prior and post disconnection, it notifies the *GPA* about the activities and the information entities that were successfully transferred both ways so the *GPA* can update the process state. The *SM* also receives the information updated during disconnection and access the data repository to perform the necessary updates on data.

3.5 Client Architecture

Due to limitations on PDA devices, we define a lightweight architecture on the client side. As depicted on figure 2 (section A), we have four components: a local coordinator, a GUI interface generator, a base role and a data repository.

Local Coordinator (LC)
In essence, the LC component is a micro version of the *GPA*. Its first task is to receive, from the *TF,* the *DWF base model* that defines the workflow system and stores it in a local repository. Similar to the *GPA*, it shows the available roles in the PDA and once the user has chosen a role to be executed, it controls its display and execution. The display of the workflow system is achieved through the *GUI Interfaces Generator (IG),* and the execution through the *Base Role (BR)*. It keeps the coordination if more than one role resides in the device.

Base Role (BR)
The RB is a component inside the LC that parses the *DWF base model,* interprets the XML information and converts it to logic instructions that enable the execution of the workflow representation. It parses and retrieves from de *DWF base model* the different definitions of the elements that build a role, the activities, the states, the information entities used, the operations involved and the external applications used during the execution of the workflow.

GUI Interface Generator (IG)
This component retrieves from the local repository the *DWF base model* and uses it to build de GUI. Basically, this component is a XML parser with instructions to interpret the definition in the base model, and display the GUI elements.

The disconnection components, except from the data repository, have been developed using the SuperWaba programming language and tested on the Palm operating system. The tool was chosen based upon the fact of its capability to run in PDAs with different operating systems. However, the architecture described above, can be implemented using other programming languages intended for pda devices.

4 Conclusions and Future Work

WFMSs have provided support to the automated execution of work, enabling organizations to achieve a more coordinated and controlled work environment. However, today's organizations require mobility, given place to the use of mobile computing that work, most of the time, in disconnected mode. In this paper, we presented the concept of disconnected workflow as a means of supporting the disconnected operations in a WFMS.

We established requirements for supporting disconnected operations in WFMSs from: previous work reported in the literature, two case studies and WFMSs' literature.

Based on the requirements found during our literature review and through the realization of case studies, we established support for disconnected operation in workflow for the *SysCoor* WFMS architecture. Our proposal for disconnected operation is targeted to PDA devices as means to carry on workflow tasks. In this account, we designed the architecture in two parts, a centralized architecture, and lightweight client architecture, all of the above, through the use of XML to represent the workflow system and enable interoperability. Also, we have considered support to workflow processes where multimedia data is required.

We devise several benefits and advantages that can be obtained from the use of the *SysCoor* architecture and also from the implementation of disconnected operation. Some of these benefits are:

1. SysCoor represents and XML based architecture, this fact enables the possibility to transfer workflow definitions from other systems that implement the use of XML and also to transfer SysCoor XML documents to other XML- based workflow machines.
2. Since the client application is defined as a XML document, the client only needs an application capable of parsing those documents and some implementation that can implement the instructions defined in the documents. Therefore, this light-weighted architecture can be implemented in PDA devices with different processing capabilities.
3. The evaluation of the task disconnection feasibility enables an appropriated election of tasks to be disconnected in order to keep a coordinated workflow environment.

We have explored the interoperability issue through the use of XML, however, in the future, we plan to do further work exploring aspects such as the integration of mobility supported with software autonomous agents, address the concurrency control handling issues and investigate mechanisms to support involuntary disconnections between others.

Acknowledgments. This work was partially supported by CONACYT under grant C01-40799 and the scholarship 164716 provided to the first author.

References

[1] Workflow Management Coalition. *The Workflow Reference Model*. Document number TC00-1003, January 19, 1995

[2] Alonso G., Gunthor R., Kamath M., Agrawal D., El Abbadi A. and Mohan C. *Exotica/FMDC: Handling Disconnected Clients in a Workflow Management System,* IBM Almaden Research Center, In Proceedings of Third International Conference on Cooperative Information Systems, Vienna, May 1995

[3] Bolcer G., *Magi: Architecture for Mobile and Disconnected Workflow,* IEEE Internet Computing. Vol. 4, No. 3, pages 46–54, June 2000

[4] Greenberg S., Boyle M. and Laberge J. *PDAs and Shared Public Displays: Making Personal Information Public, and Public Information Personal,* Personal Technologies, March 1999

[5] Grudin J. *CSCW: History and Focus.* Communications of the ACM. Vol 37, No 1, pages 92–105

[6] Satyanarayanan M., Kistler J. J., Mummert L. B., Ebling M. R., Kumar P., and Lu Q. *Experience with Disconnected Operation in a Mobile Computing Environment*. In Proceedings of the 1993 USENIX Symposium on Mobile and Location-Independent Computing, Cambridge, MA, August 1993

[7] Workflow Management Coalition. *Workflow Process Definition Interface – XML process definition Language*. Document number WfMC TC-1025, October 25, 2002.

[8] Kistler J., *Disconnected Operation in a Distributed File System*. PhD thesis. Department of Computer Science, Carnegie Mellon University, May 1993

[9] Alonso G., Agrawal D., El Abbadi A. and Mohan C. *Functionalities and Limitations of Current Workflow Management Systems*, Research Report, IBM Almaden Research Center, 1997

[10] Seungil L., Dongsoo H. and Dongman L. *Supporting Voluntary Disconnection in WFMSs*, Proceedings of the Third International Symposium on Cooperative Database Systems for Advanced Applications (CODAS 2001), pages 147–154, Beijing, April 23–24, 2001, China

[11] Pepin, C. and Kriger C., *Creating a Lotus Notes application with Domino Everyplace Enterprise*, The Technical Resource for Lotus Software, July 2002

[12] Yung Y. *Enabling Cost-effective Lightweight Disconnected Workflow for Web-based Teamwork Support*, Journal of Applied Systems Studies, Cambridge, England, 2002.

[13] Garcia F., Ramirez C., Martinez A. and Rojas F. *An Architecture for Generating Organizational Coordination Systems using Reusable Models in XML (Draft).* CICESE Research Center, June 2002

[14] Miers D. *Use of Tools and Technology within a BPR Initiative in Business Process Reengineering*, (Coulson-Thomas Colin ed.) Kogan Page Limited, pages 142–165, 1996

[15] Rojas, F. and Martinez A. I., *From Process Modeling to Enactment and Simulation*. Proceedings of the 12th European Simulation Multiconference, Manchester UK, pages 844–848, 1998

[16] Ould, M. O. *Business Process: Modeling and Analysis for Reengineering and Improvement*. John Wiley and sons Inc, 1995