Automated Workplace Design and Reconfiguration for Evolving Business Processes

Qi Zhang and Ying Zou
Dept. of Electrical & Computer Engineering
Queen's University, Kingston, Ontario
{qi.zhang, ying.zou}@ece.queensu.ca

Tack Tong, Ross McKegney and Jen Hawkins
IBM Canada Laboratory
Toronto, Ontario
{tacktong, rmckegne, jlhawkin}@ca.ibm.com

Abstract

In this ever-changing business environment, business processes are constantly being customized to reflect the up-to-date organizational structure and business objectives. Furthermore, technological updates and innovation also affect the way business is carried out. A workplace application provides an interactive electronic working environment that integrates software applications to assist users in performing their daily work more efficiently. It is challenging to maintain workplace applications as it often involves labor-intensive manual reconfiguration to adapt workplace applications to the changes to business processes. In this paper, we propose a workplace design framework that automatically analyzes business processes and generates workplace applications. Furthermore, it permits workplace reconfiguration at run time, which minimizes the interruption to users' work and simplifies deployment of business process changes to workplace applications. A prototype workplace application is designed and developed to demonstrate the effectiveness of the proposed framework.

1 Introduction

In recent years, many organizations have employed workflow management systems to automate business processes in order to achieve high efficiency while minimizing the operational costs. A workflow management system can leverage enterprise applications to manage the flow of information within an organization. A workplace (i.e., an enterprise application) as a front end of a workflow management system supports the automation of business processes [15]. Moreover, the workplace can provide an interactive working environment to assist users to perform business activities in an efficient manner.

Unfortunately, business processes in an organization tend to evolve over time in order to reflect changing organizational structures and business objectives. Furthermore, technological updates and innovation also affect the way business is carried out. To maintain the competitive edge in this ever-changing business environment, organizations must keep on modifying or reconfiguring their enterprise applications, such as workplace applications, in time to meet the latest business requirements. However, the traditional software design approach demands requirements to be fully specified before the design stage. This cannot be applicable to the design and development of enterprise applications (e.g., workplace applications) where agile and fast development is required to cope with dramatically changing requirements. Furthermore, most organizations cannot afford to manually redesign or reconfigure their workplace applications every time a change in business processes occurs, since it is a labor-intensive activity and often requires shutting down existing systems to perform maintenance operations. As a result, it is critical to consider the frequently evolving nature of business processes when designing a workplace application.

In this paper, we propose a workplace design framework that automatically generates workplace applications from business processes. To ensure the workplace application can accurately execute the activities specified in business processes, we further establish bindings between business processes and workplace applications. In

The research is funded by IBM Centers for Advanced Studies and NSERC.

order to minimize the impact on organizational operations while reconfiguring workplace applications, we provide a dynamic reconfiguration process that permits a workplace application to gradually migrate an obsolete business process to a new one without introducing inconsistencies or shutting down the system. As a result, the maintenance costs and disturbance to users' work can be minimized for the evolving business processes.

This paper is organized as follows: Section 2 gives an overview of business processes and workplace applications. In Section 3, we present a motivating example. Section 4 presents our proposed framework that automatically generates a workplace application from business processes. In Section 5, we discuss the issue of binding tasks to workplace applications. Section 6 discusses the dynamic reconfiguration process. Section 7 introduces the case studies. Section 8 gives a survey of the related work. Section 9 presents our conclusions.

2 Overview of Workplace Applications

In this section, we present an overview of business processes. We also elaborate on workplace applications and their relationships with workflow management systems.

2.1 Business Processes

A business process is a sequence of activities undertaken by an organization in order to achieve a desired business goal. A workflow definition is a computerized representation of a business process. An example *Purchase order* business process is shown in Figure 1. This process contains four tasks: *Select product*, *Select payment method*, *Purchase approval*, and *Create invoice*. A task is an inseparable activity that serves as a basic building block in a business process. Each task can have input data and/or output data. For example, *Order request* is the output data of *Select payment method* and the input data of *Purchase approval*. Each task is also associated with roles, resources, and applications. For example, *Select product* and *Select payment method* are performed by the role of *Customer*, whereas *Purchase approval* and *Create invoice* are performed by the role of *Sales Manager*. An organization typically defines a set

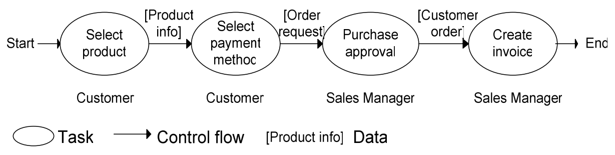


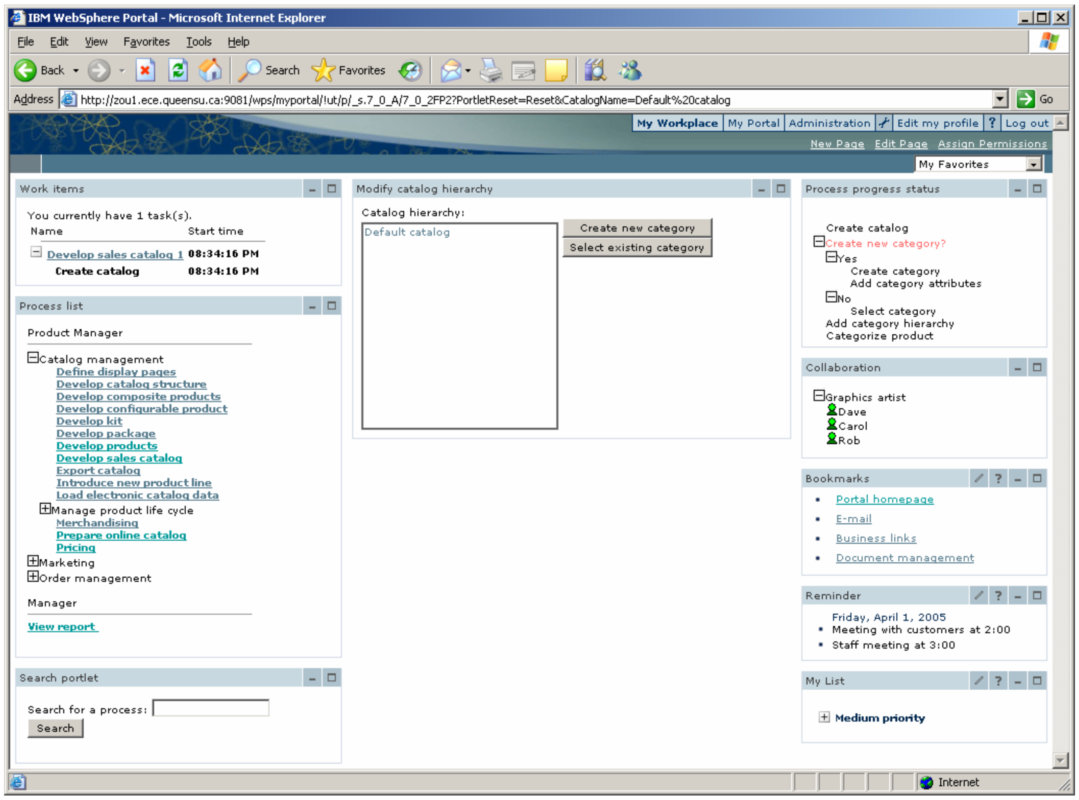
Figure 1: Purchase order business process

of business processes to represent its operations. A workflow management system is responsible for automating these business processes and assigning tasks to individual role players (i.e., users) [15].

2.2 Workplace Applications

To improve user experience in a business environment, workplace applications are designed to assist the users to "get the job done" more efficiently. The goal of the workplace application is to deliver the right tools and the right information to the user at the right time. A workplace is role-based, which means different roles, such as *Customer* and *Sales Manager*, have different tools and interfaces assigned to them. . In the context of workflow, a workplace application serves as a workflow client application [15] of the workflow management system. At run time, when a task is distributed by a back-end workflow management system, a workplace application invokes an appropriate application that implements the task. Once a task is completed, a completion message is sent to the workflow management system. A workplace application can be implemented using Web portals [4] and the Eclipse Rich Client Platform (RCP) [19].

As an example of a workplace application, a screen shot of our prototype workplace is illustrated in Figure 2. The workplace consists of several major user interface components, such as Work Items, Process List, and Process Progress Status. The Work Items component lists a collection of tasks that are assigned by a workflow management system and are to be completed by the user. The Process List component gathers a set of business processes that the user participates in. The business processes are organized based on the role of the user in a business process. Multiple roles may participate in a business process to fulfill tasks through collaboration mechanisms provided by workplace platforms. As illustrated in the Process List component, the user is assigned to two roles, including Product Manager and a


IBM WebSphere Portal - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address http://zou1.ece.queensu.ca:9081/wps/myportal/!ut/p/_s.7_0_A/7_0_2FP2?PortletReset=Reset&CatalogName=Default%20catalog

My Workplace | My Portal | Administration | Edit my profile | ? | Log out

New Page Edit Page Assign Permissions

My Favorites

Work items

You currently have 1 task(s).

Name	Start time
Develop sales catalog 1	08:34:16 PM
Create catalog	08:34:16 PM

Process list

Product Manager

- Catalog management
  - Define display pages
  - Develop catalog structure
  - Develop composite products
  - Develop configurable product
  - Develop kit
  - Develop package
  - Develop products
  - Develop sales catalog
  - Export catalog
  - Introduce new product line
  - Load electronic catalog data
  - Manage product life cycle
  - Merchandising
  - Prepare online catalog
  - Pricing
- Marketing
- Order management

Manager

View report

Search portlet

Search for a process:

Search

Modify catalog hierarchy

Catalog hierarchy:

Default catalog

Create new category

Select existing category

Process progress status

- Create catalog
- Create new category?
  - Yes
    - Create category
    - Add category attributes
  - No
    - Select category
- Add category hierarchy
- Categorize product

Collaboration

- Graphics artist
  - Dave
  - Carol
  - Rob

Bookmarks

- Portal homepage
- E-mail
- Business links
- Document management

Reminder

Friday, April 1, 2005

- Meeting with customers at 2:00
- Staff meeting at 3:00

My List

Medium priority

Internet

Figure 2: A sample screen shot of workplace

Manager. One application in the workplace implements one or more tasks specified in a workflow definition. The user interface of a task-related application is shown in the middle of the workplace. In this example, an application is invoked to accomplish the task of *Modify Catalog Hierarchy*. The application enables the user to create a new category in a catalog hierarchy or to select an existing category.

3 A motivating example

We present a purchase order example to illustrate how modifications in workflow definitions can be reflected in the workplace. As shown in Figure 3, a *Purchase order* business process handles a customer's purchase request and generates an invoice upon approval by four consecutive tasks. The *Select product* task and *Select payment method* task are performed by the role of *Customer* by filling out a purchase application on a *Sales Website*. Consequently, the role of *Sales Manager* verifies the order and approves it in the *Purchase approval* task, and creates the invoices in the

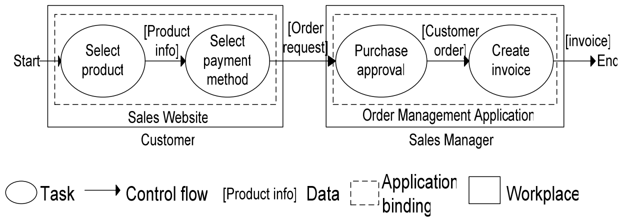


Figure 3: A purchase order process

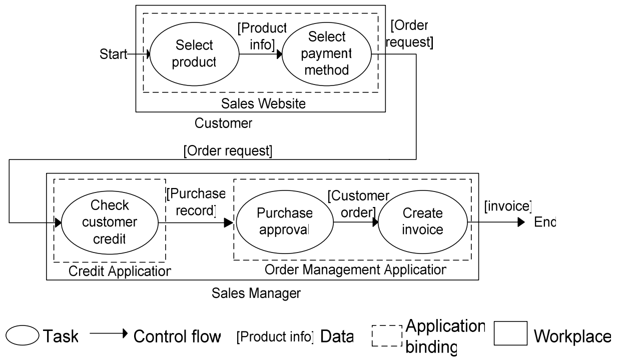


Figure 4: Modified purchase order process

Create invoice task. Moreover, the workflow definition specifies data flowing between tasks, including *Product info*, *Order request*, *Customer order*, and *Invoice*. In the workplace, the application, named *Order Management Application*, is used to perform the *Purchase approval* and the *Create invoice* tasks, respectively.

Suppose the company decides that the sales manager needs to verify the credit information of the customer before a purchase can be approved. This causes a new task, called *Check customer credit*, to be inserted before the *Purchase approval* task in the *Purchase order* workflow definition. Nevertheless, a data element, called *Purchase Record*, is added to the workflow definition. The modified workflow definition is shown in Figure 4. In this case, a new application, *Credit Application,* bound to the *Check customer credit* task, needs to be installed.

The structure of a workplace application is highly dependent on the underlying business processes. As a business process tends to change over time, it is labor-intensive and error prone if a workplace application is redesigned and re-installed every time a change occurs. It is desirable to automate the redesign process by automatically generating a workplace application from business processes. Furthermore, re-installing a workplace application often requires shutting down the existing workplace application, which may not be feasible in many circumstances, such as business processes with high availability and cross-organizational processes. For example, a sales manager is engaged in several business activities, and collaborates with other roles. The system interruption of his/her workplace may cause the loss of the work requests from other roles, and cause the suspension of other business processes that depend on the work accomplished by this role. Moreover, shutting down the system may cause a decline in business performance and hence increase maintenance costs. As a result, run time reconfiguration of the workplace application is necessary to address the changing nature of business domains. In the context of dynamic workplace reconfiguration, we aim to address the following issues:

- As a workflow definition evolves, such as task addition, deletion, and modification, users' activities in the workplace are affected. For example, if a new task is added to a currently progressing business process, the user must conduct the additional task.
- Bindings between tasks and the applications that execute the tasks may change, such as software upgrades. We treat different versions of an application as independent applications.
- A role may be assigned to other tasks. A list of new applications is added into each role player's workplace.

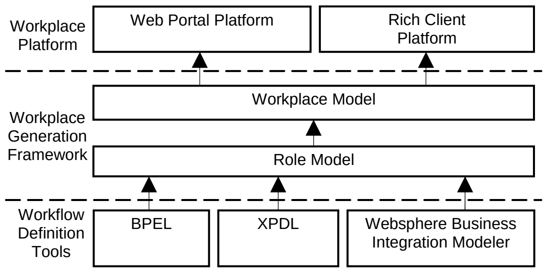

Workplace Platform
Web Portal Platform
Rich Client Platform
Workplace Generation Framework
Workplace Model
Role Model
Workflow Definition Tools
BPEL
XPDL
Websphere Business Integration Modeler

Figure 5: Workplace Generation Framework

4 Workplace Generation Framework

Our workplace generation framework leverages the knowledge imbedded in business processes and automatically generates a workplace application from workflow definitions. We aim to develop a role-based workplace that contains the content and tasks relevant only to one role. As a consequence, changes to workflow definitions can be easily tracked down to the affected roles, and the corresponding workplaces can be updated using the proposed framework with no need for manual intervention. This framework is depicted in Figure 5. We first analyze the workflow definitions represented in a variety of business process specification languages such as BPEL [13], XPDL [14] or modeled using the IBM® WebSphere® Business Integration Modeler product [2]. We focus on capturing role-related information such as the responsibility, rights, and required resources for a role to accomplish their daily work. Moreover, we represent this role-related information in a role model. To implement a workplace application, we further generate a generic workplace model that describes abstract user interface components and layouts of a workplace. By employing this workplace model, a platform-specific workplace application can be automatically generated using available workplace implementation technologies, for example, Web portals and the Rich Client Platform. Details of each step of the process will be explained in the next few subsections.

4.1 Generating a Role Model

A workplace application provides a role-based working environment to help users to perform their tasks. In this context, a user is assigned a set of roles in the workplace. For example, a user can be both the *Sales Manager* and *Product Manager* in the organization. Workflow definitions encode the knowledge of roles in an organization. Typically, a role is involved in a number of business processes; in each process, the role has to perform a set of tasks. While performing the tasks, the role may need to access applications and resources in the organization. Additional constraints, such as task priorities, scheduling information and time constraints could determine more precise behavior of the role in the workplace. We analyze workflow definitions, extract role-related information, and represent it in a domain model, which we are calling a role model. Essentially, such a role model captures all information about the role within the organization.

4.2 Generating a Workplace Model

Based on our knowledge of workplace applications, we develop a workplace model for representing the high-level structure of a workplace. This model allows task-related applications, supporting functionality, layout, and contextual information of a task to be specified. In our workplace model, as illustrated in Figure 6, a user can be a member of several user groups. When the user logs on to the workplace, the user has a view of the user-specific workplace interface. In the workplace, the applications that implement tasks are organized in a number of *categories*, which are shown as tabs in the workplace user interface. Each *category* may contain many *pages*, each of which displays applications in the workplace. Each *application* presents its own information, and allows users to perform activities in the application window. The layout of each page is defined by the *LayoutElements*. Since the user can simultaneously interact with several applications, these applications need to communicate in the underlying infrastructure. We define events to

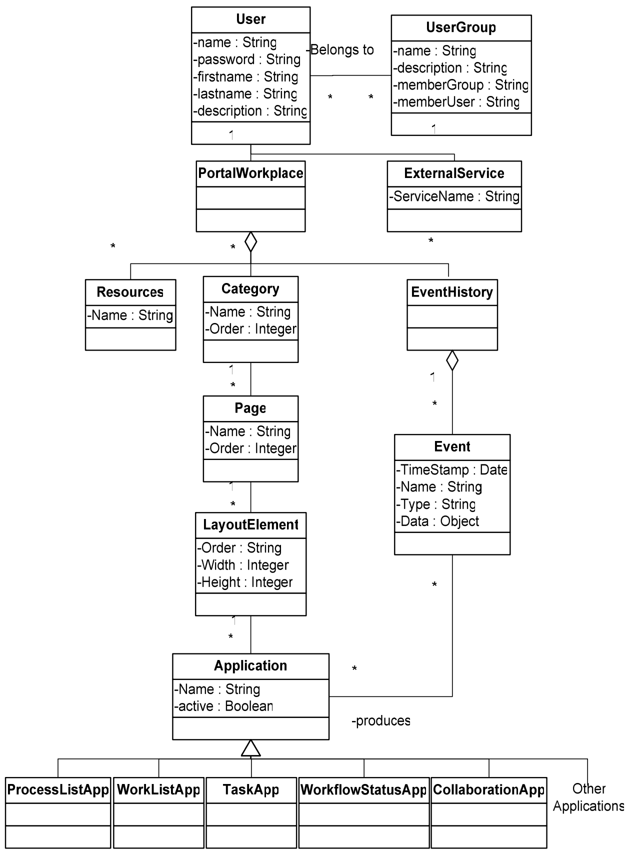


Figure 6: Workplace Model

handle the communications between the applications.

In our framework, the workplace model is automatically generated from the role model through a domain model transformation. Based on our domain knowledge of the role model and the workplace model, we define a collection of transformation rules. For example, the *tasks* in the role model are mapped to the pages for applications in the workplace model.

4.3 Generating the Workplace Application

We generate a workplace application using the workplace model obtained in the previous step. Moreover, we generate the source code for workplace infrastructure and configuration files based on specific workplace implementation technology. Specifically, we develop an individual code generator for each platform. For example, to generate a workplace application deployed on Web portals, our code generator produces a portal configura-

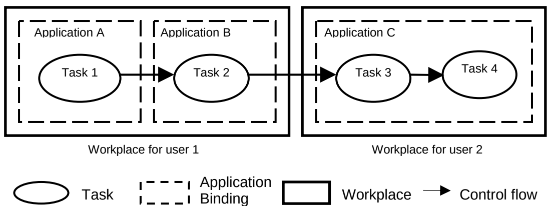


Figure 7: Example bindings: tasks to applications

tion file and source code of portlets [20], each of which represents one application in the Web portal platform. The code generator also generates the code for coordinating the running applications. Moreover, the code generator can generate a skeleton for an application based on the task information. However, the data and control logic of the generated application still need to be implemented by site developers.

5 Binding Workflow Tasks to Applications

To enable business process execution in a workplace environment, tasks need to be bound to applications in the workplace. A task-to-application binding is specified when the workplace is created for the user. Figure 7 illustrates a workflow definition composed of four tasks. Task 1 is bound to Application A, and executed by Application A. Similarly, Task 2 is bound to Application B in user 1's workplace, whereas Task 3 and Task 4 are bound to application C in user 2's workplace.

At run time, the workflow management system initiates a workflow instance from a workflow definition. A workflow instance consists of several task objects. Each task object corresponds to one task specified in a workflow definition. Figure 8 shows a sequence of task objects of a workflow instance. When a task object needs to be executed, the workflow management system sends a request to the workplace. Consequently, the workplace application fetches the binding information between the task and its associated application. Furthermore, the workflow management system assigns the task to each user. The user accomplishes the task by executing the associated application in the workplace. An application can implement one or more tasks, such as Task 3 and Task 4, shown in Figure 8. In this case, if Task 4 is completed by Application 3, it also implies that Task 3 is also completed by Application 3.

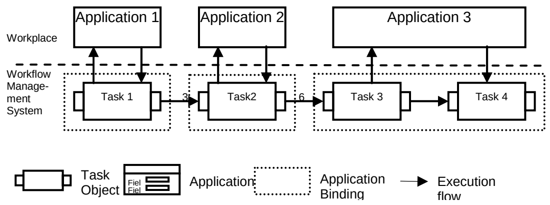


Figure 8: Coordination between workplace and workflow management system

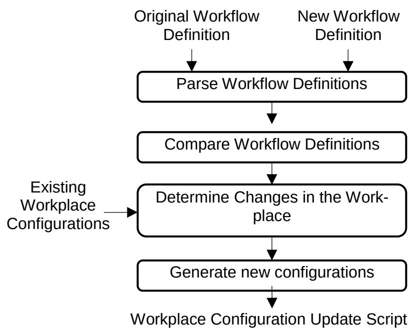

Original Workflow Definition
New Workflow Definition
Parse Workflow Definitions
Compare Workflow Definitions
Existing Workplace Configurations
Determine Changes in the Work-place
Generate new configurations
Workplace Configuration Update Script

Figure 9: Generating new configuration updates for a workplace

6 Dynamic Reconfiguration

In this section, we elaborate on our approach for detecting changes in workflow definitions, and generating an updated workplace configuration to incorporate these changes. A workplace configuration includes the bindings between tasks and their corresponding applications, interface definitions of the applications, and the workplace layout. The content and layout of each workplace is determined by each user's individual profile and configuration.

6.1 Change Detection and Processing

Changes in workflow definitions can be detected through exchanging information with the workflow management system. Typically, a workflow management system maintains workflow definitions, which are used to create workflow instances. Through the communication with the workflow management system, any changes to workflow definitions can be obtained.

In order to reconfigure the workplace application infrastructure, we utilize the workplace generation process to produce workplace configuration. Figure 9 depicts the process of generating new workplace configurations. We reuse the linkage established between business processes and workplace applications in our workplace generation framework. However, instead of generating a workplace application, the code generator generates configuration update scripts, which are applied to the workplace reconfiguration process.

As a result of changes in a workflow definition, new bindings between tasks and applications may be introduced. Certain applications need to be removed from the workplace application if they no longer bind to any tasks in the new workflow definitions; some applications may be added in the event that new tasks are added to the new workflow definitions.

6.2 Workplace Dynamic Reconfiguration

The objective of our proposed dynamic reconfiguration process is to maintain the work consistency when a workplace is switched from one workflow definition to an updated one while minimizing the disturbance to the users' current work within a workplace application. To accomplish this objective, we validate the affected workflow instances under the new workflow definitions, and determine an appropriate migration path for users to follow the new workflow definitions. Details of our reconfiguration process will be explained in the following subsections.

6.2.1 Validating Workflow Instances

Changes to workflow definitions can invalidate currently progressing workflow instances in a workflow management system. For example, adding a task to an existing workflow definition can cause the currently progressing workflow instances under the old workflow definition to become inconsistent with the new workflow definition. Therefore, each affected workflow instance needs to be verified for consistency when changes are made to workflow definitions. The research on the validation of workflow instances has been studied extensively in the literature, and various approaches have been proposed [6][8][21].

In general, a workflow management system provides several settings that allow a workflow management system to be reconfigurable to new workflow definitions. The reconfigurability settings of a workflow management system can be summarized in the following four cases.

Case 1: A workflow instance of an old workflow definition is set to continue its execution without being affected. However, newly initiated workflow instances of this business process must follow the new workflow definition.

In the case that a workflow definition is modified while the workflow instance from the old workflow definition is executing, the following two situations can occur in the workflow management system:

Case 2: Workflow instances of old workflow definitions can be adapted to the new definitions without being affected. This case is possible if the execution sequence currently completed by the workflow instance is valid under the new definition. In our example described in Figure 4, consider a workflow instance that has completed *Select product* and *Select payment method* task when the changes occur. Under the new definition, the execution sequence for these two tasks remains the same. Therefore, this workflow instance can adapt to the new definition without being affected.

Case 3: The workflow instance can be adapted to the new definition. However, the application to execute the affected tasks is required to terminate. This is the case when the associated tasks are changed in the new workflow definition. For example, a new task is added and an old task is removed; therefore, the task-to-application binding needs to be modified. Consequently, the workflow management system rolls back the workflow instance to a previous state. We define this difference in workflow definition as an inconsistency spot. For instance, in our example in Figure 4, consider a workflow instance that has completed *Select product*, *Select payment method,* and *Purchase approval* when the changes occur. Since the *Check customer credit* has not completed, the workflow management system will roll back the workflow instance to the *Select payment method* task.

Case 4: The workflow management system is set to terminate the progressing workflow instances under the old workflow definitions.

When the workflow definition of a currently progressing workflow instance is changed, our reconfiguration process retrieves the reconfigurability settings from the workflow management system and determines whether to immediately switch to the new workflow definition or wait until the old workflow instances are complete.

6.2.2 Life Cycle of an Application

In this section, we define the life cycle of an application that is bound to one or more tasks specified in workflow definitions. The life cycle of an application consists of five states: *Inactive, Active, Suspended, Completed,* and *Terminated.* The state transition diagram of an application is shown in Figure 10. The *Inactive* state indicates the tasks implemented by the application have not been initiated. Therefore, it is not activated in the workplace. The *Active* state indicates the user is current engaged in the application to perform tasks. *Completed* denotes the tasks enacted by the application are completed. *Suspended* state indicates that the application is initiated and temporarily idle; the user may resume work on it later. The *Terminated* state indicates execution of the tasks in the workplace has failed.

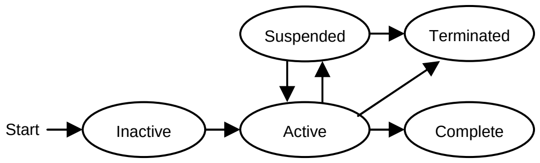


Figure 10: Life cycle of an application in the workplace

From a user's perspective, a task in a workflow instance is delivered to a user's workplace based on the execution sequence specified in the workflow definition. The task is temporarily queued in a waiting list of the workplace, and is shown to the user in the workplace. For example, the *Create Catalog* task listed in Figure 2 is task queued in the workplace. At any time, there can be many applications initiated or requested by other users; these applications are either in *Active* or *Suspended* state. At any time, a user can actively work on one application.

6.2.3 Executing Workplace Reconfiguration

Once the workflow instances of a changed workflow definition have been validated by the workflow management system, the reconfiguration process needs to take appropriate actions to terminate or continue the applications that are bound to the affected running workflow instances of old workflow definitions. In our approach, we identify reconfiguration criteria that determine which actions should be applied to the currently progressing workflow instances. The aim of defining reconfiguration criteria is to maximize the integrity of the users' current work in the workplace. We are concerned about active and suspended applications because these applications could cause inconsistency to the state of the workplace application after reconfiguration. An inactive application can be reconfigured according to the four cases in the previous section without being noticed by the user.

We apply a path analysis to all currently progressing workflow instances of the same workflow definition and compare the relative position of the active and suspended applications to the inconsistency spots in the workflow definition. These inconsistency spots represent the differences between the two versions of workflow definitions. Based on this relative position, we define the reconfiguration criteria by distinguishing among the following three scenarios:

- If the currently active application is in the upstream of all the inconsistency spots (i.e., no inconsistency spot has been reached yet), the reconfiguration process binds the affected tasks in the new workflow instance with the new application and updates the workplace configuration. In this scenario, the users' current work can carry on in the new workflow instance.
- If the currently active application is in the downstream of an inconsistency spot, the reconfiguration process updates the task-to-application binding information for the new workflow definition. The users' current work can continue through the unchanged tasks in the new workflow definition.
- If the currently active application is in the inconsistency spot, we can take two approaches. As described in Case 3 in Section 6.2.1, in which the workflow management system decides to adapt the currently progressing workflow instance to the new definition by rolling back, the active application is terminated, and the reconfiguration process updates the configuration for the new work-

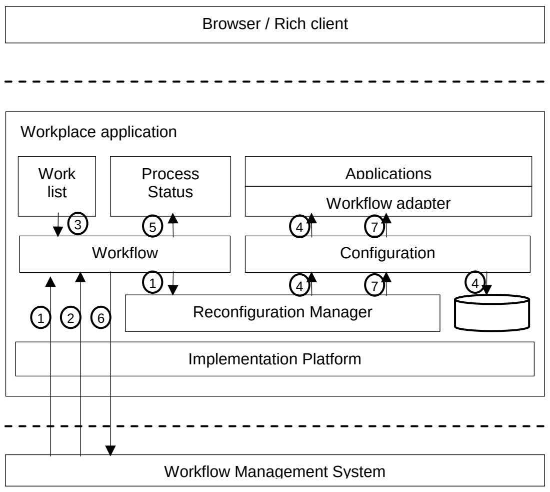


Figure 11: Architecture of our dynamic reconfigurable workplace

flow definition. However, the reconfiguration process can also follow Case 1 discussed in Section 6.2.1, which allows the old workflow instance to complete without changes.

Multiple workflow instances can exist in a user's workplace as the user is assigned tasks from different workflow definitions. The reconfiguration process is applied to workflow instances in which changes are introduced in the corresponding workflow definition. Other workflow instances under unchanged workflow definitions are kept intact. Once the reconfiguration criteria are determined for each individual affected workflow instance, the reconfiguration process can automatically modify the run time information of each workflow instance.

6.2.4 Workplace Reconfiguration Procedure

A summary of the architecture of our workplace application is shown in Figure 11. The *Workflow* component serves as the interface to the workflow management system. The *Configuration* component stores the configuration of the workplace, such as the user profile, page layouts, and application binding information in a database. The *Work List* component is an application responsible for delivering tasks of each workflow instance to the workplace. The *Progress Status* component displays the execution status of each workflow instance. The *Workflow Adapters* component is used to handle the events and data communication with the underlying workflow management system. We utilized the *Reconfiguration Manager* component to generate reconfiguration updates and perform the dynamic reconfiguration process. Furthermore, the reconfiguration process consists of the following 7 steps:

(1) Upon receiving a new definition from the back-end workflow management system, the *Workflow* component forwards the new definition to the *Reconfiguration Manager*, which generates workplace configuration update scripts, as mentioned in Section 6.1.
(2) The *Workflow* component retrieves the workflow reconfigurability settings from the workflow management system, as mentioned in Section 6.2.1.
(3) The *Workflow* component determines the reconfiguration criteria for each workflow instance and determines if active and suspended applications are affected, as discussed in Section 6.2.2.
(4) The *Reconfiguration Manager* performs the reconfiguration in the workplace. New applications are installed, and the binding information is updated by the *Configuration* component. The workplace also notifies the user about, and explains any changes to the tasks in the workplace. The progress of each affected workflow instances is also modified according to the reconfigurability settings of the workflow management system, as described in Section 6.2.1.
(5) The *Workflow* component updates the content of *Progress status* application and *Work List* application.
(6) The *Workflow* component notifies the workflow engine that reconfiguration is complete. After this point, both the workplace application and the workflow management system can resume their normal operations.
(7) In the case where an application is no longer required, the workplace application uninstalls this obsolete application when the workplace configuration and workflow instances are not depending on it.

We demonstrate our reconfiguration process using the motivating example in Section 3. Suppose there is a customer creating a new purchase order and is at the point of the *Select payment method* task when the workflow definition is changed. The workflow management system has decided to adapt the currently progressing workflow instances to the new definition. In this case, the change is detected by the *Reconfiguration Manager*, which generates and executes a configuration update script. This allows the *Credit application* to be installed to the workplace application. *Reconfiguration Manager* then verifies the workplace reconfiguration criteria and determines that the inconsistency spot (i.e., *Check customer credit* task) is in the downstream of the current active task (i.e., *Select payment method* task). The binding between the *Select payment method* task and the application, *Sales Website*, remains the same in the new workflow definition. In this context, the reconfiguration process that adapts the workplace to the new workflow definition will not affect the customer's interaction with the *Sales Website*. The *Reconfiguration Manager* then performs the dynamic reconfiguration. Once complete, the *Workflow* component notifies the workflow management system about the completion of the reconfiguration, and the workplace then resumes its normal operation.

7 Case Studies

We designed and developed our proposed framework to automatically generate a prototype workplace from workflow definitions. The workplace contains a dynamic reconfiguration process. Our prototype workplace is deployed on an IBM WebSphere Portal [18] platform. Moreover, we conducted case studies to verify our dynamic reconfiguration process. The results demonstrate that the prototype workplace can effectively adapt to changes to workflow definitions and dynamically reconfigure itself without manual intervention. In the following subsections, we explain the implementation and case studies in more detail.

7.1 WebSphere Portal Technology

The WebSphere Portal server is a Web application server that is responsible for providing contents of the Web portal to Web users. Figure 12 illustrates the architecture of the WebSphere Portal server. A portal server typically contains portlets, which are self-contained Web applications and perform specific functionality inside the portal. Typically, a portlet is represented as a mini-window on a portal Web page and can be installed,

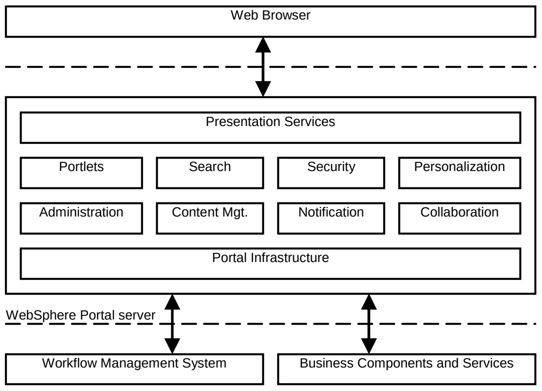


Figure 12: Websphere Portal Architecture

removed, or customized. In the context of our prototype workplace, a portlet provides a user interface for an application that implements one or more tasks specified in workflow definitions. The WebSphere Portal provides default portlets that implement mailbox, calendar, and document management. As shown in Figure 12, the WebSphere Portal also leverages the latest Web portal technology to offer the following features [18].

- Integrating information from distinct information and resources, through the content management module.
- Search and navigation services provide help for users to navigate to the required information.
- Notification service can inform users of events from the system or organization.
- A personalization module is capable of delivering customized content for a specific user.
- Collaboration services provide features to help users share and with to specific information.
- Security feature provides access control to information for different users.

The WebSphere Portal also provides libraries and toolkits for developing portlets. Since portlets are a type of Java™ servlet, designing a portlet application follows a similar procedure as designing a Servlet-based application. Moreover, WebSphere portal also provides additional functionality to customize portlets, such as state management, messaging, and portlet cooperation [28].

7.2 Automatic Workplace Generation

We parsed the workflow definitions, and generated a role model and a workplace model. In order to generate a workplace application for the WebSphere Portal, we first produced an XML configuration file that is used by the WebSphere Portal to automatically create the layouts of the workplace. Second, we generated source code for portlets for the user interface components, such as *Process List*, *Work Item*, and *Navigation*, required in the workplace. Third, we utilized a portlet messaging mechanism [28] provided by the WebSphere Portal Server to handle communications between portlets. This enables the workplace to provide a collaborative and integrated working environment for a user to accomplish their work. Lastly, the code is generated for a complete workplace application. A screen shot of our generated workplace application is shown in Figure 1.

7.3 Experiment

As an enterprise application, the workplace dynamic reconfiguration process can affect the business performance of the organization. Furthermore, it also impacts the user's perception of the working environment. Due to these reasons, we conducted an experiment to evaluate the performance of our workplace dynamic reconfiguration process. More specifically, we examined the execution overhead of the dynamic reconfiguration process and its impact on user's experience. In our experiment, we applied the prototype workplace to the business processes of a typical commerce application, which contains 26 workflow definitions. To test the performance of our reconfiguration technique, we deployed our generated workplace on an Intel® Pentium® 4 desktop computer at 2.66 GHz with 1 GB of RAM. We listed two example workflow definitions in Table 1.

In the case studies, we aim to determine the following performance characteristics in our system: (1) The time needed for each step in the reconfiguration process, (2) the effect of processing multiple workflow instances at run time. In the first example (*Workflow Definition 1* column in Table 1), we tested a workflow definition that involves two tasks that are bound to one application. We conducted two test cases for this workflow definition. In the first case, there was one

Table 1: Results of our workflow reconfiguration performance experiment

	Workflow Definition 1		**Workflow Definition 2**	
	Test Case 1	Test Case 2	Test Case 1	Test Case2
Number of workflow instances running	1	3	1	2
Number of running workflow instances affected	1	2	1	1
Reconfiguration Script Generation Time (ms)	4031	4094	4125	4344
Workplace Reconfiguration Time (ms)	53220	69392	568996	535963
Instance Reconfiguration Time (ms)	<1	<1	<1	<1
Work Items update Time (ms)	<1	<1	<1	<1
Experienced Deletion Time (ms)	<1	<1	<1	<1
Actual Deletion Time (ms)	6062	5422	72955	88830
Total Delay time (ms)	57251	73486	573121	540307

workflow instance running and it was affected by the reconfiguration process. In the second case, there were three running workflow instances, and two of them were affected by the reconfiguration process. In the second example of workflow definition (*Workflow Definition 2* column in Table 1), tasks are bound to five applications. The two test cases were conducted on the second workflow definition. In the first case, one workflow instance was affected by the reconfiguration process. In the second test case, two workflow instances are initiated and one workflow instance of them was affected by the reconfiguration process.

We found that the reconfiguration process takes the most amount of time (up to 568996 ms or 7.5 minutes) during our experiments, as shown in the row of *Workplace Reconfiguration Time* in Table 1. Deletion operations also take time to complete, as listed in the row of *Actual Deletion Time* in Table 1. However, the actual deletion is performed when the workplace is idle. Therefore, the experienced deletion time is less than 1 second, as depicted in the row of *Experienced Deletion Time* in Table 1. In addition, we observed that the instance reconfiguration time is negligible, as indicated in the row of *Instance Reconfiguration*

Time in Table 1. Hence the total delay time in each case is the sum of the time measurements for each step, not including the actual deletion time.

Uploading new configurations in a WebSphere Portal Server is an expensive operation, since it involves communicating with a back-end database. However, the reconfiguration process is conducted in the background, and only has impact on the user who is working on the affected workflow instances. The user can continue to work on other unaffected workflow instances during the reconfiguration process. We expect a performance increase by replacing the current desktop PC with a powerful enterprise-level server. Adapting a running workflow instance to the new workflow definition is not time-consuming, since we observed that handling multiple workflow instances does not increase the *Instance Reconfiguration Time*. The applications we used in our experiment are simple and do not access any back-end services. Furthermore, the workflow definitions studied do not involve large amounts of data; hence the workflow instance reconfiguration time is short. However, we expect longer delays for handling workflow instances for applications involving large amounts of database access. Nevertheless, we expect the delays not to be noticeable by the users of the workplace.

In summary, the results of case studies show that a workplace can be automatically generated from business processes. In addition, this workplace can perform dynamic reconfiguration process without interrupting users' work. This reduces the cost of system interruption, and eases the maintenance process for a frequently evolving business application.

8 Related Work

The concept of workplace application is new to the business domain. Most existing workplace products, such as IBM workplace [16], MySAP.com workplace [17], are developed in recent years. Many of these products are aim to support business processes in the workplace. However, as workflow definitions change over time, developing and managing a workplace application has revealed challenges to the developers.

Model-driven development [10] has been used in many application domains. The model-driven develop process is suitable for generating workplace application from business workflow model because workflow model can be considered as high level domain-independent model that captures the problem domain. Through a sequence of transformation, each step we provided more concrete design information, and eventually a workplace application can generated. The advantage of this methodology is that it imposes a systematic design process in the software development, and it also provides a high degree of automation.

Research issues relating to workflow validating and flexible execution in workflow management systems have been studied in the past decade. In [25, 26] a comprehensive study on flexibilities in workflow engine is given: In [6][7][8][9] the concept of workflow change primitives and workflow consistency criteria are proposed. In [24], the flexibilities are implemented in the workflow management system. However, these researchers only address the issues of maintaining workflow instance consistencies with respect to workflow definitions. No others have examined the workflow execution environment, such as a workplace, to be dynamically updated with changes from the workflow definitions.

Researches in dynamic reconfiguration [11] [12] have been predominant in recent years. The latest advancement in this field is a study of autonomic computing [27], which aims to design systems that are self-managing, that is, self-configuring, self-healing, self-protecting, and self-optimizing. Workplace applications share similarities with autonomic systems, since a workplace application has to detect the changes in the workflow definition and adapt itself to such change at run time. We believe our framework leverages features from dynamic reconfiguration and autonomic computing that allow the workplace to be dynamically configurable and adaptive to the evolving workflow definitions.

9 Conclusion

In this paper we presented a framework for designing and reconfiguring a workplace according to evolving business processes. We proposed a framework to automatically generate workplace applications from business processes. We further proposed a reconfiguration process that allows a workplace to automatically adapt to these changes at run time. This framework can achieve high automation in designing and configuring workplace applications. Our case studies demonstrated that the reconfiguration process can be automatically performed without interrupting users' work.

In the future, we plan to examine the performance and management issues of workplace reconfiguration, and refine the current approach to incorporate more usability, reliability and scalability features.

References

[1] Edward A. Stohr, J. Leon Zhao. Workflow Automation: Overview and Research Issues. *Information Systems Frontiers*, Volume 3 Issue 3, September 2001.

[2] WebSphere® Business Integration Modeler home page. http://www.ibm.com/software/-integration/wbimodeler

[3] Matthias Vering, Grant Norris, Peter Barth, James R. Hurley, Brenda Mackay, David J. Duray, Matthias Vering. *The E-Business Workplace*. John Wiley & Sons, Inc., June 2001.

[4] Anura Guruge. *Corporate Portals Empowered with XML and Web Services*. Elsevier Science & Technology Books., MA, USA, 2002.

[5] Ying Zou, Terence C. Lau, Kostas Kontogiannis, Tack Tong, Ross McKegney. Model-Driven Business Process Recovery. *Proceedings of the 11th Working Conference on Reverse Engineering (WCRE'04)* pp. 224-233, Delft, The Netherlands, Nov. 08 - 12, 2004.

[6] Casati, S. Ceri, B. Pernici, and G. Pozzi. *Workflow Evolution*. In Proceedings of ER '96, pages 438-455.

[7] Classen, I., Weber, H. and Yanbo Han. Towards Evolutionary and Adaptive Workflow Systems *In Proceedings of First International Enterprise Distributed Object Computing Workshop (EDOC '97)*, 24-26 Oct. 1997

[8] Shazia Sadiq. Workflows in Dynamic Environments – Can they be managed? *Proceedings of the Second International Symposium on Cooperative Database Systems for Advanced Applications (CODAS99)*, Woollongong, Australia. March 27-28, 1999.

[9] S. Rinderle, M. Reichert, P. Dadam. “Flexible Support of Team Processes By Adaptive Workflow Systems” Distributed and Parallel Databases, Kluwer Journal, 16, pp 91-116, 2004.

[10] S. Sendall, W. Kozaczynski. Model Transformation: The Heart and Soul of Model-Driven Software Development, *IEEE Software, Vol. 20, Issue 5*, pp. 42 – 45, September 2003.

[11] Joao Paulo A Almeida, Maarten Wegdam, Marten van Sinderen, Lambert Nieuwenhuis. Transparent Dynamic Reconfiguration for CORBA. *In Proceedings of the 3rd International Symposium on Distributed Objects and Applications*, pages 197—207, September 2001.

[12] De Palma N., Bellissard L., Riveill M., Dynamic Reconfiguration of Agent-based Applications. *European Research Seminar on Advances in Distributed Systems (ERSADS'99)*, Madeira, Portugal, April 1999.

[13] Business Process Execution Language for Web services version 1.1. http://www-128.ibm.com/developerworks/library/ws-bpel

[14] Workflow Process Definition Interface -XML Process Definition Language http://www.-wfmc.org/standards/XPDL.htm

[15] Workflow Reference Model. The Workflow Management Coalition Specification, http://www.wfmc.org/standards/docs/tc003v11.pdf

[16] IBM Workplace home page. http://www-.lotus.com/products/product5.nsf/wdocs/workplacehome

[17] mySAP.com Workplace Technology. http://-www.sapinfo.net/public/en/article.php4/com-vArticle-193333c63b4af4a922/en

[18] WebSphere® Portal for Multiplatforms http://www.ibm.com/software/genservers/portal/

[19] Eclipse Rich Client Platform. http://www-.eclipse.org/rcp/

[20] JSR 168: Portlet Specification. http://jcp-.org/en/jsr/detail?id=168

[21] Heinl, P., Horn, S., Jablonski, S., Neeb, J., Stein, K. and Teschke, M. A Comprehensive Approach to Flexibility in Workflow Management Systems. *Proceedings of the Inter-*

national Joint Conference on Work Activities Coordination and Collaboration, pp. 79-88, San Francisco, California, USA, February 22-25, 1999.

[22] Ginige, A., Re-Engineering Software Development Process for eBusiness Application Development. *Proceedings of the Software Engineering and Knowledge Engineering Conference (SEKE 2003)*, San Francisco Bay.

[23] K. Belhajjame, G. Vargas-Solar, and C. Collet. A flexible workflow model for process-oriented applications. *Proceedings of the 2nd International conference on Web Information Systems Engineering, WISE'2001*, Kyoto-Japan, December 2001.

[24] J.J. Halliday, S.K. Shrivastava, and S.M. Wheater. Flexible Workflow Management in the OPENflow System. *In Proceedings of the 4th International Enterprise Distributed Object Computing Conference (EDOC 2001)*, pages 82–92, Seattle, Washington, 4-7 September 2001,

[25] Halliday, J.J., Shrivastava, S.K., and Wheater, S.M. Implementing support for work activity coordination within a distributed workflow system. *In Proceedings of the Third International Conference on Enterprise Distributed Object Computing (EDOC '99)*, page 116-123, September 1999.

[26] L. Aversano, G. Canfora, A. De Lucia, S. Stefanucci, Automating the Management of Software Maintenance Workflows in a Large Software Enterprise: A Case Study, *Journal of Software Maintenance and Evolution: Research and Practice*, vol. 14, no. 4, pp. 229-255, 2002.

[27] IBM autonomic computing home page. http://www.research.ibm.com/autonomic/

[28] IBM WebSphere® Portal V5 A Guide for Portlet Application Development. http:-//www.redbooks.ibm.com/redbooks/pdfs/sg246076.pdf

Trademarks