A Workflow-Oriented System Architecture for the Management of Container Transportation

Sarita Bassil[1], Rudolf K. Keller[2], and Peter Kropf[1]

[1] Département IRO, Université de Montréal, C.P. 6128, succursale Centre-ville, Montréal, Québec, H3C 3J7, Canada
{bassil, kropf}@iro.umontreal.ca
[2] Zühlke Engineering AG, Wiesenstrasse 10a, CH-8952 Schlieren, Switzerland
ruk@zuehlke.com

Abstract. In this paper, we introduce a workflow-oriented system architecture for the processing of client requests (CRs) for container transportation. In the context of multi-transfer container transportation, the processing of CRs can be achieved by specific sequences of interdependent activities. These sequences need to be just-in-time created. They also need to be adapted to deal with unexpected events that may occur. Workflow technology is used to model and to manage the processing of CRs. The creation and the adaptation of activity sequences require first, an optimized scheduling of a limited number of resources (by also respecting CRs constraints); and second, a number of special workflow concepts and functionality to correctly manage activity sequences. Optimization models are involved to take care of the resource management and of the activity scheduling. Enhancements of workflow concepts and functionality for workflow management systems are investigated to deal with an activity sequence creation and adaptation. Finally, the proposed architecture includes a rule processing part to reduce the time-consuming manual interaction with the system.

Keywords: system architecture, workflow management system, process planning, flexible workflow, transportation application.

1 Introduction

Workflow technology provides an appropriate support for the planning of activities. It allows for the coordination and the follow-up of tasks explicitly defined. Domains such as transportation can take advantage of this technology, in particular if its underlying challenging aspects are accommodated.

Nowadays, a growing number of transportation companies are facing fleet management (FM) issues and are forced to cope with highly constrained environments while maintaining a satisfactory level of efficiency. A definition of FM is given in [7]: "FM covers the whole range of planning and management issues from procurement of power units and vehicles to vehicle dispatch and scheduling of crews and maintenance operations". This type of management can be tackled under various lengths of planning horizons and levels of detail: the strategic, the tactical and the operational level. The latter involves a short planning horizon where the level of detail is relatively high.

Among the sectors in which FM represents a particularly challenging issue, the *multi-transfer container transportation* (MTCT) – that could be extended to the multi-modal freight transportation [17] – has gained in interest in recent years [7]. In MTCT a container is moved from terminal to terminal with the possibility to shift it from vehicle to vehicle before delivering it to the final destination. In this paper, we focus on MTCT at the operational level, in which a close follow-up of activities must be achieved in order to ensure a good client request (CR) satisfaction.

In the context of MTCT management, it appears that the processing of a CR can be achieved by a specific sequence of interdependent activities: e.g., attach an empty container to a vehicle, move the empty container to the origin location, load the container, move the container to the final destination, unload the container. Moreover, MTCT requires to just-in-time create the sequence of activities, or to just-in-time adjust a basic sequence of activities, needed to accomplish a specific request. It also requires a high degree of adaptation of ongoing activities' sequences to deal with unexpected events (e.g., new request arrival, delayed vehicles, crew member desistance).

Our approach exploits the workflow technology [28] to model and to manage the processing of CRs. Specific features of Workflow Management Systems (WfMSs) can result in positive effects for the transportation domain. These features include new concepts and functionality.

We are aware of the fact that workflow technology in the transportation domain is usually used to manage logistic activities where documents and information are passed from one participant to another according to a set of procedural rules [12]. However, the central issue related to workflows in our approach is the focus on supporting *flow of work* and not on supporting flow of "documents" [1]. Furthermore, we adopt the idea of emergent workflows described by Jørgensen and Carlsen in [15]: "emergent workflows provide an integrated support for planning, coordination and performance of work". The workflow definition and enactment are intertwined.

Taking into account a proposed extension of the Workflow Reference Model (WfRM) [30] that covers the support of particular workflow concepts and functionality [5], we already introduced an original transportation system framework adapted to the MTCT application [4]. This framework is conceptually divided into two main layers: a *workflow layer* and a *coordination layer*.

The workflow layer essentially gathers a set of concurrently running workflow instances, each of them being associated with a specific CR. Knowing that a workflow instance is composed of a sequence of activities, and that the state of these activities is known at any time, it is hence possible to determine the set of resources used such as vehicles, containers and drivers. Since we are dealing with activities to be achieved by humans, the dispatching of the appropriate crews at the appropriate time plays an important role. We take advantage of the worklist concept to ensure this task.

The coordination layer is responsible for a certain number of tasks that ensure the efficient allocation of resources. It is responsible to receive the new requests, to ask the workflow layer to instantiate new workflow instances and to react accordingly to unexpected events by sending modification orders to the workflow layer. In brief, the coordination layer gathers a set of optimization algorithms that are used for the management of resources and for the scheduling of activities.

Taking into account this framework, we propose in this paper a workflow-oriented system architecture applied in the MTCT context. This system – that we call the MTCT system – enables the user (called "system administrator" in the rest of the paper) to efficiently track and monitor the progress of many CRs in process. Moreover,

the system allows crew members (called "drivers" hereafter) to identify at the right time their assigned activities and to transmit to the system administrator the state of each activity from its selection to its completion.

In the following, we explore the MTCT application (Section 2) and investigate workflow concepts and functionality for WfMSs to deal with this application (Section 3). We then present the architecture of the MTCT system (Section 4). This architecture is based on workflow technology, optimization engine technology and rule engine technology. Section 5 gives an example of a CR processing planning that illustrates the use and the characteristics of the developed architecture. Section 6 reports on the implementation of the MTCT system. Section 7 discusses related work and Section 8 concludes the paper.

2 The MTCT Application

Usually, a CR for container transportation gathers at least the following information: an origin location where goods are picked-up, a destination where goods are delivered, a pick-up and a delivery time window. To answer a CR, a number of activities of different duration are involved. These activities need to be performed in a certain order, and are scheduled within a given time window depending on the individual request information, on the resource availability and on the possible paths to follow.

We consider a set of transportation units that we call resources. This set is composed of a fixed number of containers with fixed wheels, trucks (i.e., vehicles) without loading space, crews and terminals. We suppose that the transportation company offers a full container-load, where one container carries at one time only merchandise related to one client. Besides the resources, we define a set of six activities, that we call *activity templates*. A composition of these activities provides a possible solution to satisfy a CR. An activity is assigned to a specific driver that becomes responsible for its execution within a specific time and by taking into account specific information that we call input attributes. Table 1 shows the six activity templates we defined.

Table 1. Activity templates involved in the processing of a CR for container transportation

	(1) Attach container to vehicle	**(2) Detach container from vehicle**	**(3) Move vehicle to location**	**(4) Load container**	**(5) Unload container**	**(6) Wait at location**
Input attributes	Container_ID Vehicle_ID Location_ID	Idem[1]	Container_ID Vehicle_ID O_location_ID[2] D_location_ID[3]	Container_ID Location_ID	Idem	Idem
Assignment	Driver_ID	Idem	Idem	Idem	Idem	Idem
Time	MinD/MaxD[4] WUT[5] EST/LST[6]	Idem	Idem	Idem	Idem	Idem

[1]The same as left, [2, 3]The ID of the origin (resp., destination) location (it does not necessary correspond to the origin (resp., destination) location of a CR), [4]The minimum/maximum duration, [5]The warm-up time (time when the driver is informed about the activity to carry out), [6]The earliest/latest starting time.

A possible composition of these activities to answer a CR could be the following sequence: (1)-(3)-(6)-(4)-(3)-(2)-(6)-(1)-(3)-(6)-(5)-(3). Note that a "wait at location" activity is sometimes necessary before going further in the processing of a request.

The composition of activities should be based on a transportation network in which a number of nodes (i.e., locations) and edges (i.e., paths) between these nodes are defined. As a first configuration, we consider a transportation network with a central depot or terminal where resources are located and where a transfer is possible. A transfer is defined as the action of shifting a container from one vehicle to another vehicle. As an example, the sequence "(2)-(6)-(1)" in the composition presented above, represents a transfer.

Taking into account this configuration, a number of path scenarios are possible for the management of CRs. The simplest scenario would be to consider that the satisfaction of a CR consists to ask a couple container/driver (c/d – We consider that each driver is associated with a specific vehicle.) to leave the depot P at a specific time, to pick up the goods at the origin location O specified by the request, to deliver the goods at the final destination D and then to go back to P. In other words, answering a request consists of accomplishing the path P-O-D-P ("simple scenario").

Another scenario would be to ask a couple c/d to leave P at a specific time, to pick up the goods at O and to go back to P with the possibility to make a transfer at P (i.e., to change the driver and the vehicle at P) before delivering the goods at D and then to go back to P (i.e., P-O-P-D-P). This represents a "transfer scenario". It can be motivated by the non-availability of drivers. In this case, we hence need to plan a path P-O-P when a driver is just available to make this portion of the whole path.

In the first two scenarios, c/d should return to P before answering a new request. We may however, consider that a couple c/d is free to answer a new request as soon as the goods are delivered at a specific destination (i.e., P-O_1-D_1-O_2-D_2-P, where O_1/D_1 are related to a specific request and O_2/D_2 are related to another request). We talk about a "round scenario". A combination of the transfer scenario and the round scenario is also possible. Here is an example: P-O_1-P-D_1-O_2-D_2-P.

The scenarios presented above take into account a transportation network with a central depot. This transportation network configuration could be extended to a more complex one that gathers a number of distributed depots. Considering this configuration, a "multi-transfer scenario" of the kind P_1-O-P_2-P_3-…-P_n-D (where $\{P_1, P_2, \ldots, P_n\} \in \mathcal{P}$, $\mathcal{P}$ being the partition of the set of depots) is possible.

An in-depth description of the MTCT application would involve the discussion of the different unexpected events (e.g., delayed vehicles, crew member desistance) that may occur, and their implication in our context. This however is beyond the scope of our paper. In the remaining of this paper, without loss of generality, only the "new request arrival" event will be considered to explain and discuss the MTCT System.

3 Workflow Concepts and Functionality for the MTCT System

The MTCT system we propose is workflow oriented and supports transportation processes. It should inform the drivers at the right point in time about the work to accomplish, while providing them the appropriate information. For transportation processes, it is sometimes impossible to fix all activity attributes as soon as a workflow in-

stance is created/launched. It is also impossible to predict all events that may occur and that may necessitate a workflow structural deviation or an activity attribute updating. Our workflow-oriented system should provide the appropriate workflow concepts and functionality to support all these aspects. Otherwise, as stated in [9], the system would have to be "bypassed", which would cause (besides other problems) the problem of missing documentation. The following presents the list of special concepts and functionality for WfMSs necessary to deal with the MTCT application:

The Activity Template Concept. In order to introduce a standard way for defining activities, it is useful to define a set of activity templates related to the container transportation management. Activity templates are used by the system to schedule the different activities in a workflow model/instance. Each activity template consists of an elementary task with three types of attributes:

- *Input attributes*, which specify the (material) resources needed to accomplish a task.
- *Assignment attributes*, which specify the (human) actor responsible of accomplishing this task. This is mainly used by the system to let the task appear in the worklist where it should appear.
- *Time attributes*, which specify the (min/max) duration of the task, its (earliest/latest) starting time, and its related warm-up time.

Table 1 gives examples of activity templates related to the MTCT management.

The Warm-Up Time Concept (WUT). "Time" plays a crucial role in the transportation domain. Therefore, time attributes should be defined for each activity. Temporal aspects such as the duration and the starting time (fixed calendar date) are discussed in literature. The ADEPT project for instance, treats these two aspects in detail [9]. A differentiation should however be done between (1) the *planned starting time* of an activity, (2) the *activation time* of an activity (i.e., when the activity is due, taking into account the control flow of the workflow), and (3) its *assignment time* to a worklist. Usually, within current WfMSs an activity is assigned to a worklist as soon as it is due within the flow. However, crew members should not be surprised by activities, and they should know in advance about the next activity to carry out. Hence, the assignment time of an activity to a worklist should depend on the planned starting time of the activity and on the necessary warm-up time. Eder *et al.* tackle a similar problem by working on future personal schedules [10]. Their work is motivated by the need to provide early information about future tasks (i.e., forecasting of tasks). Their approach is based on probabilistic time management.

The Dynamic Setting/Updating of Attributes at the Workflow Instance Level. In the MTCT system, activity attributes are provided by a "Solution Provider" component that is external to the WfMS (see Section 4). Hence, no data flow between activities exists, and input attributes of all activities can be logically linked to the "start node" output attributes. However, we should be able at any time to set/update input attributes of activities not yet in a "running" state. It should also be possible to dynamically (re-)assign such activities to a valid workflow actor (i.e., late binding of resources [16]), and to dynamically set/update the time attributes of these activities by always respecting the temporal constraints.

The Dynamic Insertion of an Activity at the Workflow Instance Level. This insertion should be based on previously defined activity templates. During insertion, temporal constraints should be respected and input attributes of the inserted activity should be linked to newly generated data elements. This is well discussed in [22]. The dynamic insertion of an activity could be extended to the dynamic insertion of a subworkflow. As an example from the MTCT application, the sequence of the two activities "detach container from vehicle" and "attach container to vehicle" should be inserted each time a container needs to be transferred from one vehicle to another.

The Dynamic Deletion of an Activity at the Workflow Instance Level. As an example, a deletion of the "move vehicle to the depot P" activity from a workflow instance should be possible in the special case where a round scenario is involved (refer to Section 2). In the context of the MTCT application, major verifications before permitting the deletion of an activity are related to the activity state (e.g., an activity in a "running" state cannot be deleted unless it is possible to preserve its context).

The Dynamic Management of Worklists. The reassignment or the deletion of an activity already assigned to a specific worklist should be complemented by a correct worklists management. Following a reassignment, the workitem that corresponds to the reassigned activity should be removed from its original worklist and it should appear in the appropriate worklist taking into account the new assignment (if not null). The workitem that corresponds to a deleted activity should be removed from its worklist. The updating of an activity input/time attribute should be complemented by a correct updating of the information provided by the worklists.

In short, there are a number of issues that arise from the list of workflow concepts and functionally just exposed. Insights from today's WfMS research projects [9, 10, 16, 22] are combined and contrasted, and an extension of the WfRM to accommodate these concepts and functionality is proposed in [5]. A discussion of the WfRM extension is beyond the scope of this paper.

4 Architecture of the MTCT System

We describe in this section the MTCT system architecture shown in Fig. 1, and give an overview of its underlying constructs. Two phases are distinguished in this system: the build-time phase and the run-time phase.

During build-time, a set of activity templates is defined using the Workflow Definition Tool. The latter is also used to design basic workflow models that capture the sequencing of the most likely required activities for the processing of a CR. Activity templates and workflow models are stored in the Workflow Repository as Workflow and Activity Template Definitions. Another component of the system is the Resource Definition Tool. It allows the definition of resources that make possible the accomplishment of the activities. The resources are stored in the Workflow Repository as Resource Definitions. The planned (fixed) availability of the human resources (i.e., shift) are defined via workflows using the Workflow Definition Tool. This will be de-

tailed in Section 4.1. A third component of the system is the Optimization Model Definition Tool. It allows for describing optimization models (OMs). These models are used to (re-)plan the processing of CRs. Refer to Section 4.2 for details. Another part of the definition deals with modification rules (MRs). These are usually defined using a rule editor (not shown in Fig. 1 for simplicity purpose). They go into the MR Repository. MRs and rule engines are discussed in Section 4.4. As a last part of the build-time phase, the Transportation Network Information is fixed within a specific database. It defines in particular locations/depots of the transportation network as well as the durations to move between two locations. This information, once it is specified, is rarely modified.

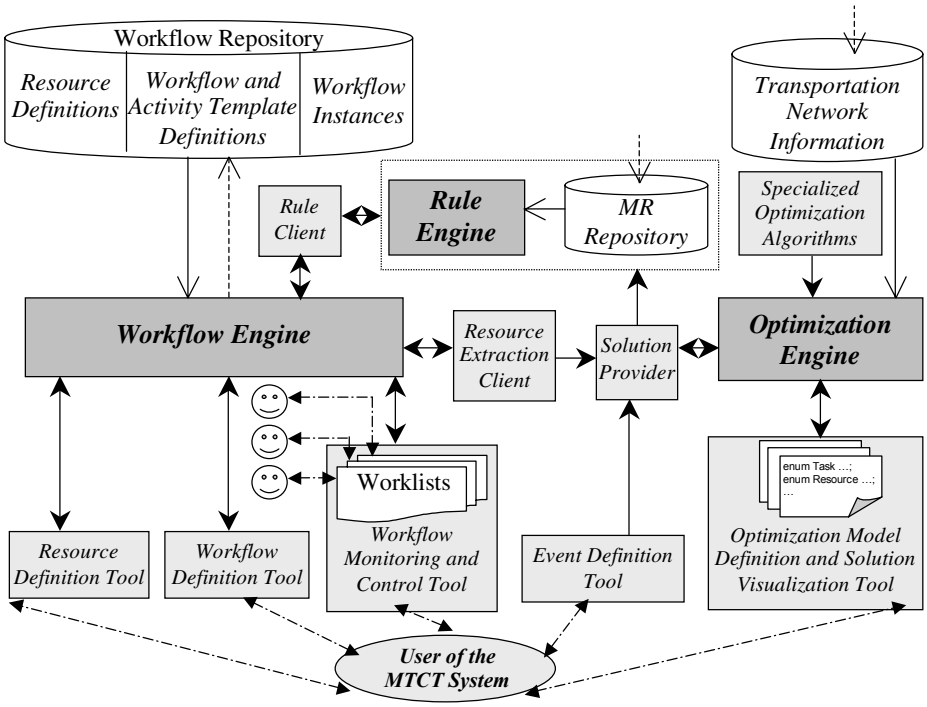


Fig. 1. Architecture of the MTCT System

At run time, when a new event appears, the system administrator of the MTCT system uses the Event Definition Tool to define this event (e.g., a new request arrival) as well as its related data. This triggers the selection of a specific OM. The Solution Provider module takes care of this selection. As long as no solution is found, a number of OMs may be solved. Specialized Optimization Algorithms are called by the Optimization Engine to solve a selected OM. Three data sources are used to initialize the OM:

1. The Event Definition Tool provides event information.
2. The Resource Extraction Client provides data related to the current reservation/unavailability of resources (reflected by the state of our workflow instances).

3. The Transportation Network Information database (already described in the build-time phase).

When an optimized solution is generated, it is interpreted and translated into a set of modifications that can be automatically applied, via the Rule Client, on the pool of currently running instances. We talk about modifying the pool of workflow instances when a new workflow instance is created as well as when a structural or an attribute modification is applied on an existing workflow instance. The interpretation of solution implications on this pool is the task of the Rule Engine and the MR Repository. The system administrator can also make manual modifications. Indeed, the optimized solution can be displayed to the system administrator via the Solution Visualization Tool, so that she can take decisions regarding the modification(s) to bring to the pool of instances. Manual modifications are applied from the Workflow Monitoring and Control Tool. Details about workflow management are given in Section 4.3. The Workflow Engine is responsible of applying modifications on the pool of workflow instances. It also executes the instances by enforcing the sequencing of the activities and by dispatching work at the appropriate time to the appropriate human resource. Worklists (which are part of the Workflow Monitoring and Control Tool) are used to show which activity needs to be carried out. Each human resource has her personal worklist to quickly identify her assigned activities.

4.1 Resource Management

The diagram of Fig. 2 describes the entities that are used for capturing the resource structure and the relations between them. A resource type (e.g., vehicle) gathers a set of resources (e.g., V101, V202). Unlike material resources, human resources (i.e., drivers) are not continuously available but only within their own shift. The planned unavailability (i.e., the complementary of the availability or shift) of the different drivers over a period of time is captured by a workflow with parallel branches. Each branch of the workflow corresponds to a specific driver and each activity of the branch defines a period of unavailability for this driver.

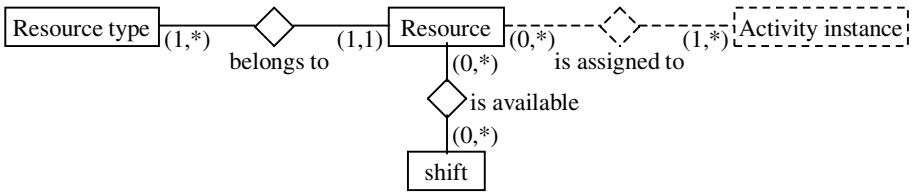


Fig. 2. Entity-relation diagram for the resource management

Resources can be assigned to activity instances. The tables corresponding to the dashed part of the entity-relation diagram (Fig. 2) are frequently updated. At a specific time, the reservation of the different resources is deduced from the set of activity instances where the state is different from “completed”, “deleted” or “skipped”.

4.2 The Optimization Part of the System

The need for an optimal management of resources when (re-)planning activities in the container transportation domain is well recognized [24] and can be answered by defining specific OMs. OMs can be defined as a data-independent abstraction of an optimization problem in which the aim is to find the best of all possible solutions. More formally, the goal is to find a solution in the feasible region (i.e., the set of all possible solutions), which has the minimum (or maximum) value of the objective function (i.e., a function which determines how good a solution is) [3]. In our context, we use OMs to plan the processing of CRs and to re-plan this processing when necessary. These OMs should assign resources to activities while satisfying the constraints of a CR as well as while respecting the information related to the transportation network. Our resource allocation problem is modeled as a constraint satisfaction problem that we resolve using constraint programming [27].

When modeling our problem, we leveraged the work reported in [26, 29]. Suitable strategies to answer a CR according to the different scenarios presented in Section 2 were developed. An example of a strategy consists of minimizing the duration of a request processing (i.e., minimizing the reservation of a set of resources). Taking into account this strategy, the following defines a model that picks an available resource and schedules the different activities to answer a CR according to the simple scenario:

- Given a set R of resources from a specific type.
- Given a set S of ordered triples $<r_i, st_{ij}, ft_{ij}>$, (st_{ij}, ft_{ij}) specifying a reservation block (starting/finishing time) for the resource r_i. Note that a number of reservation blocks can be associated with r_i at a specific time.
- Given information related to a specific request: origin location O, destination location D, pick-up time window (put_{min}, put_{max}) and delivery time window (dt_{min}, dt_{max}).
- Given transportation network information: duration(Move(P-O)), duration(Move(O-D)), duration(Move(D-P)) where P corresponds to the depot, and the durations of the specific operations at O and D: duration(Load), duration(Unload).
- We define the objective function Z as the duration of the request processing:
  Z = duration(Waiting_time(O)) + duration(Waiting_time(D)) + c
  Where c is a constant:
  c = duration(Move(P-O)) + duration(Load) + duration(Move(O-D)) + duration(Unload) + duration(Move(D-P))
- The problem to solve is the following:
  Minimize Z
  Subject to the following constraints (where t^* corresponds to the leaving time at P):
  (C1) $t^* + \text{duration(Move(P-O))} + \text{duration(Waiting\_time(O))} \geq put_{min}$
  (C2) $t^* + \text{duration(Move(P-O))} + \text{duration(Waiting\_time(O))} + \text{duration(Load)} \leq put_{max}$
  (C3) $t^* + \text{duration(Move(P-O))} + \text{duration(Waiting\_time(O))} + \text{duration(Load)} + \text{duration(Move(O-D))} + \text{duration(Waiting\_time(D))} \geq dt_{min}$
  (C4) $t^* + \text{duration(Move(P-O))} + \text{duration(Waiting\_time(O))} + \text{duration(Load)} + \text{duration(Move(O-D))} + \text{duration(Waiting\_time(D))} + \text{duration(Unload)} \leq dt_{max}$
  (C5) $\forall <r, st_j, ft_j> \in S$ where $r \in R$, $t^* > ft_j \vee t^* + Z < st_j$

When selecting a specific OM, such as the one defined above, the Solution Provider module provides to the Optimization Engine the necessary data to solve this model (i.e., the "given statements"). Once a solution is found, this module is also responsible of letting the Rule Processing part of the system "know" about this solution.

4.3 Workflow Management

The architecture of the MTCT system is based on WfMS modules that are compliant with the WfRM proposed by the WfMC [30]. We rely on workflow technology because it provides support in three broad functional areas [25]: (1) workflow definition: capturing the definition of the flow of work, (2) workflow execution: managing the execution of the workflow processes in an operational environment, sequencing the various activities to be performed, and (3) workflow monitoring: monitoring the status of workflow processes and dynamically configuring the runtime controller. We also rely on workflow technology because a number of today's WfMS research projects propose interesting and inspiring approaches to deal with dynamic modifications of workflow instances [2, 11, 18, 22, 23]. In the architecture of the MTCT system, the user can for instance adjust certain attributes or bring structural modifications to existing workflow instances at runtime. Examples include postponing the execution time of a specific activity, changing the driver responsible of an activity, adding a transfer to an already planned CR processing, and so on. Finally, once the execution of a workflow instance is completed, workflow technology allows for recording this instance as historical data (i.e., audit). Workflows are hence seen as providing a way to represent a blueprint of activities so that analysis becomes possible for the detection and for the prevention of bottlenecks at the operational level.

4.4 Modification Rules and Rule Engines

In the architecture of the MTCT system, we use rule engine technology to represent and exploit MRs. A rule such as: *"If a new request arrives, and if a solution is found when a specific optimization model is solved, and if a specific basic workflow model has already been defined, and if a workflow instance manager exists, then a new workflow instance related to the newly arrived request is instantiated from the basic workflow model"* can be nicely coded as a declarative statement [19]. The rules can be coded as standalone atomic units, separate from and independent of the rest of the application logic. This makes the rules easier to develop and maintain. Rule engines have already been applied for dynamic modification of workflows [21]. The idea is to use an automatic rule-based approach with a focus on cancer therapy workflow scenarios. The approach intends (1) to detect semantic exceptions, (2) to derive which instances and control flow areas are affected, and (3) to automatically adjust the affected areas. In the MTCT system, we intend to take advantage of this approach for the automatic structural modification of instances. At this level of our work, we only experimented with the automatic workflow instantiation and attributes setting.

5 An Example of a Client Request Processing Planning

In this section, we illustrate the different steps for answering a CR taking into account the simple scenario discussed in Section 2 and considered in Section 4.2. When a request is received, the system administrator uses a "request information" form (Fig. 3) provided by the Event Definition Tool to specify the related information. This infor-

mation, the availability of the resources and the transportation network information are used to generate a solution if any.

If a solution is found (as in our case), the system administrator uses the Workflow Monitoring and Control Tool to instantiate a basic workflow model that captures a sequence of eight activities defined between a "start" activity and an "end" activity: (S) start, (A1) attach container to vehicle, (A2) move vehicle to O, (A3) wait at O, (A4) load container, (A5) move vehicle to D, (A6) wait at D, (A7) unload container, (A8) move vehicle to P, (E) end. Since the solution shown in Fig. 3 does not specify a waiting time at O, the activity (A3) is then deleted from the instance. Note that in this case, the activities constitute a simple sequence of actions. Other examples may yield to activities whose control flow is best captured in a state-transition diagram.

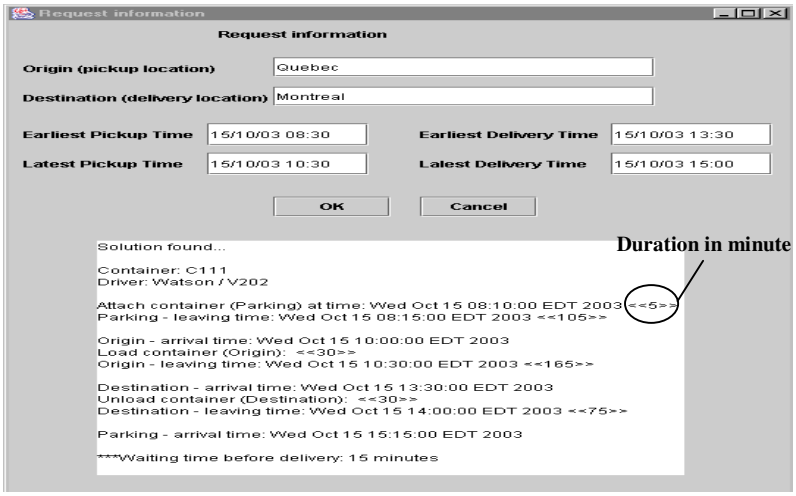


Fig. 3. "Request information" form

Two types of edges are used in our workflow model: the control edges and the time edges. The WfMS prototype we are using – ADEPT [8] – does not allow the specification of a fix calendar date for the activities' starting time. We use instead the "time edge" concept and we define a minimum and a maximum distance between the "start" activity (S) and each of the activities (A). The earliest and the latest starting time of (A) are specified taking into account the real starting time of (S).

The system administrator launches (S) to specify the five following output attributes (see Fig. 3): the CR origin location (Quebec), the CR destination location (Montreal), the central depot of our transportation network, and the container and vehicle IDs shown in the solution (C111 and V202). These attributes are given as input to the different activities of the workflow instance. The other elements of the solution (e.g., driver, starting time/duration of the activities) are used to set the assignment attribute and the time attributes for each activity.

The set of steps just accomplished by the system administrator (i.e., workflow instantiation, activity deletion, execution of (S) and attributes setting) can be automated so that time-consuming manual interactions with the system are reduced. For that reason, we need MRs such as the following, which applies to a workflow instantiation:

```
WHEN
  there is a RequestInformation called ?ri
  there is a OptimizationModel called om      such that OM_ID=1 and Solution_Found=true
  there is a ProcessTemplate called ?pt       such that PT_Name.compareTo("Simple")=0
  there is a ProcessInstanceManager called ?pim
THEN
  Apply ?pim
    so that assert(createProcessInstance(?pt, ?ri.Request_ID, "Standard", "Administrator"))
```

The notation used above stems from ILOG JRules [13]. It is based on an English-like syntax. Four class instances are involved in the rule shown above: RequestInformation and OptimizationModel are classes from our implemented application; ProcessTemplate and ProcessInstanceManager are classes provided by the ADEPT API.

6 Implementation of the MTCT System

Part of the presented architecture has already been implemented (MTCT System version 0.1). This version includes an extended WfMS and an optimization system.

We use ADEPT, a WfMS prototype developed at the University of Ulm [8]. Two main criteria were applied to retain this system among other WfMSs. The first and foremost criterion is its compliance with the basic WfRM, as well as its support for the "activity template" concept, for temporal aspects (except the WUT concept), and for two structural modifications (the insertion and the deletion of an activity). The second criterion refers to the availability of its API.

A *Mediator* component that extends the existing ADEPT API was implemented. This component provides functions for the dynamic setting/updating of attributes (input attributes, assignment attributes and time attributes) and for the dynamic management of worklists. The WUT concept is not supported yet.

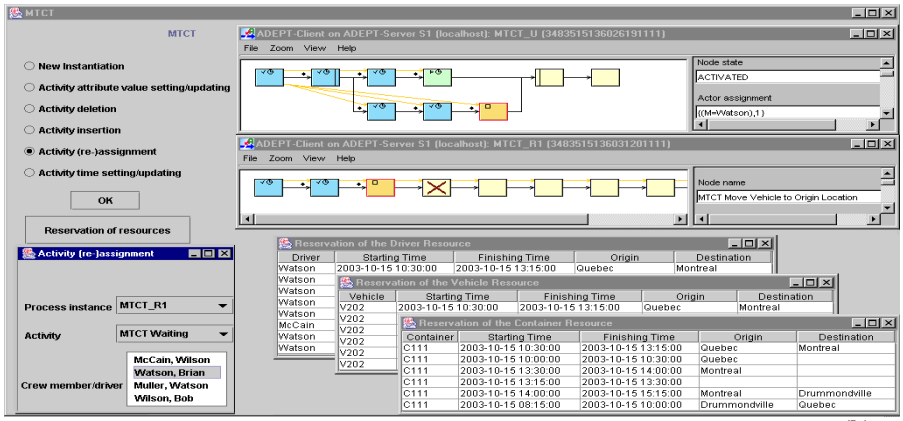
We use OPL Studio from ILOG [14] to define OMs that are solved using the CPLEX optimization algorithms. Since our implementation is based on ADEPT which is implemented in Java and which uses an Oracle relational DB, the advantage of OPL is twofold: (1) We can access its C++ API from Java code, relying upon the Java Native Interface (JNI). So, once a model is designed, compiled and tested in OPL Studio, it can be easily solved from a Java application by interfacing with OPL. (2) We can establish a connection to a database and initialize the model by reading the appropriate relational tables. Having this in mind, we implemented in Java the ADEPT Resource Extraction Client and the Solution Provider.

In a standalone fashion, we have incepted integrating rule engine technology (ILOG's JRules) into our MTCT system. The integration with the MTCT system will be accomplished at a later stage, once we are satisfied with the results of applying MRs on the pool of workflow instances. Since rules for the transfer scenario, the round scenario and the multi-transfer scenario are much more complex than those for the simple scenario, we will deal with them in later versions.

In Fig. 4, we present a screenshot of the MTCT system. The main window in (a) shows the Workflow Monitoring and Control Tool. It provides functionality the system administrator can use to modify the pool of the workflow instances. The first two windows (top right) are monitoring windows and show running workflow instances: a planned unavailability workflow instance, and one of the CR processing instances that

is going on. The three windows at the bottom right show the current reservation of the different resources. This information is automatically extracted and used by the Solution Provider component; however, the system administrator is also able to visualize it at any time. The last window here (bottom left) shows one of the possible windows the system administrator can access to make manual modifications to the pool of instances – the "Activity (re-)assignment" in this case. In fact, each time she chooses one of the six possible functionality options, the corresponding window is opened. The two windows in (b) show the worklists of two specific drivers. All necessary information is available for the execution of an activity related to a request processing instance. As we can see, activities related to a planned unavailability workflow instance are also communicated to drivers via their worklists.

(a)



(b)

Fig. 4. Screenshot of the MTCT System version 0.1. **(a)** The environment of the system administrator; **(b)** The environment of the drivers

As a final note in this section, the performance of the system shall be briefly discussed. A performance evaluation may be performed in terms of answering questions such as "how much time does it take to generate a solution using OMs?" and "how much time does it take to modify the pool of instances (e.g., to instantiate a new workflow instance, to update already planned/instantiated ones)?". Based on our current prototype implementation, we expect to encounter a performance problem which is mainly related to the continuously access to the database. In fact, some of the ADEPT API functions useful in our context are not implemented yet. Consequently, we sometimes had to manipulate the ADEPT database directly, especially when implementing the *Mediator* component. The performance of the system would be considerably enhanced if the functions of this component were inherently provided by the WfMS (e.g., ADEPT).

7 Related Work

The MTCT system is, to our knowledge, the first workflow-based system to support the processing of CRs for container transportation. Planner systems in the transportation domain usually use planning languages such as PDDL (Planning Domain Definition Language) [20] to describe a logistic problem and its solution, represented as a sequence of ordered activities (a plan). These systems do not however allow for plan monitoring and control during execution – an inherent characteristic of the workflow technology on which the MTCT system is based.

TeleTruck [6] is a prototype software system that is probably the closest related one to the MTCT system. This prototype has been developed for planning, optimizing, and monitoring of road haulage. The underlying approach is based on multi-agent technology: physical objects (e.g., trucks, drivers, trailers, load spaces) are modeled by intelligent agents. Those agents are able to reason and plan on the basis of their individual resources and means provided by the corresponding physical objects. For TeleTruck, the work is not described for a specific CR but for a specific vehicle (i.e., each vehicle's plan is represented separately). In the MTCT system, however, workflows are used to provide a complete description of the work process to be enacted for a particular request. Consequently, the processing of each request is separately documented and can easily be tracked. If necessary, the client can be quickly informed about the state of her request. The originality of the MTCT system in respect of the TeleTruck system stems from the fact that the reservation of the resources is reflected by the workflows, and that the support for dispatching drivers in their daily work is smoothly provided by worklists, a concept tightly associated with the workflow technology. In the TeleTruck system, the traditional timetable approach is used. Finally, in this system, the emphasis is put on the on-line re-planning to cope with highly dynamic environment, whereas in our system we take advantage of already existing optimization algorithms, by only defining simple OMs.

8 Conclusion

In this paper, we studied the *multi-transfer container transportation* (MTCT) application and we described an architecture with all its underlying concepts to deal with client requests (CRs) for container transportation. It is based on workflow management, optimization engine technology and rule engine technology. We claim that the processing of CRs and the management of human resources availability can be adequately and profitably dealt with using workflows. Since a workflow instance is composed of activities and the state of these activities is known at any time, the reservation of resources can be deduced. Optimization models (OMs) take care of the management of resources and the scheduling of activities following the occurrence of an unexpected event such as the arrival of a new CR. Solutions from the optimization part of the system are translated into a set of modifications that are applied on the pool of workflow instances. The use of the rule engine technology considerably reduces the time-consuming manual interactions with the MTCT system in an obvious way.

We believe that the MTCT system provides an environment that can easily help on the one hand, a user of a container transportation company to efficiently manage the

processing of CRs, and on the other hand, a crew member to just-at-the-right-time identify the work to do.

We feel that the architecture proposed can also be applied in the context of transportation applications other than the MTCT application. We may think about local express-mail services and dial-a-ride services where the planning of activities can be solved as a Pick-up and Delivery Problem. Moreover, production systems in which assembly lines are involved could take advantage of this architecture. Indeed, in such systems, two issues are interrelated: the management of limited (shared) resources and the management of processes.

As future work, we aim to further investigate several issues that are central to our system. Among them is the support of unexpected events such as delayed vehicles, crew member desistance and technical problems. The only event supported up to now by the MTCT system is the "arrival of a new CR". Another issue is the distributed worklists we would like to investigate in order to dispatch work on a network of several computers, which could be located at different terminals/vehicles. Finally, modification rules are an important research issue for us. We will have to come up with rules that would bring structural modifications to workflow instances. We will also have to define more complex OMs taking into account the different path scenarios (simple and complex scenarios). Solutions coming from these OMs will potentially be translated into structural modifications of workflow instances.

Acknowledgments. The completion of this research was made possible thanks to funding provided by the NSERC (CRD-224950-99), Bell Canada's support through its Bell University Laboratories R&D program, and support by the CIRANO. We thank Benoît Bourbeau from CIRANO, who has given valuable thoughts regarding the MTCT application.

References

1. Abbot, K.R. and Sarin, S.K., Experiences with Workflow Management: Issues for the Next Generation. In *Proceedings of the 5th Conf. on Computer Supported Cooperative Work (CSCW'94)*, 113-120, Chapel Hill, NC, Oct 1994.
2. Agostini, A. and De Michelis, G., Improving Flexibility of Workflow Management Systems. In van der Aalst, W., Desel, J., and Oberweis, A. (Ed.), *Business Process Management: Models, Techniques, and Empirical Studies. LNCS 1806 Springer*, 218-234, 2000.
3. Algorithms and Theory of Computation Handbook, *CRC Press LLC*, 1999. Appearing in the Dictionary of Computer Science, Engineering and Technology, *CRC Press LLC*, 2000.
4. Bassil, S., Bourbeau, B., Keller, R.K., and Kropf, P., A Dynamic Approach to Muli-transfer Container Management. In *Proceedings of the 2nd Int'l Workshop on Freight Transportation and Logistics (ODYSSEUS'03)*, Palermo, Sicily, Italy, May 2003. On-line at <http://www.unipa.it/Odysseus/Odysseus2003_file/odysseus-main_file/pdf/Programma.htm>.
5. Bassil, S., Rolli, D., Keller, R.K., and Kropf, P., Extending the Workflow Reference Model to Accommodate Dynamism. *Technical Report GELO-155*, Software Eng. Lab, University of Montreal, Quebec, Canada, Feb 2003. On-line at <http://www.iro.umontreal.ca/~bassil/>.
6. Bürckert, H.-J., Fischer, K., and Vierke, G., Transportation Scheduling with Holonic MAS – The TeleTruck Approach. In *Proceedings of the 3rd Int'l Conf. on Practical Applications of Intelligent Agents and Multiagents (PAAM'98)*, 577-590, London, UK, Mar 1998.
7. Crainic, T.G., Long-Haul Freight Transportation. *Handbook of Transportation Science*, R.W. Hall (Ed.), 2nd Edition, Kluwer Academic Publishers, 2002.

8. Dadam, P. and Reichert, M., The ADEPT WfMS Project at the University of Ulm. In *Proceedings of the 1st European Workshop on Workflow and Process Management (WPM'98) (Workflow Management Research Projects)*, Zürich, Switzerland, Oct 1998.
9. Dadam, P., Reichert, M., and Kuhn, K., Clinical Workflows – The Killer Application for Process-oriented Information Systems? In *Proceedings of the 4th Int'l Conf. on Business Information Systems (BIS'2000)*, 36-59, Poznan, Poland, Apr 2000.
10. Eder, J., Pichler, H., Gruber, W., and M. Ninaus, Personal Schedules for Workflow Systems. In *Proceedings of the Int'l Conf. On Business Process Management (BPM'03)*, 216-231, Eindhoven, The Netherlands, Jun 2003.
11. Ellis, C.A. and Keddara, K., A Workflow Change Is a Workflow. In van der Aalst, W., Desel, J., and Oberweis, A. (Ed.), *Business Process Management: Models, Techniques, and Empirical Studies. LNCS 1806 Springer*, 201-217, 2000.
12. E-Workflow – the Workflow Portal, Workflow Case Studies (Transportation). 1997-1998. On-line at <http://www.e-workflow.org/case_studies/transportation/index.htm>.
13. ILOG JRules. On-line at <http://www.ilog.com/products/jrules/>.
14. ILOG OPL Studio. On-line at <http://www.ilog.com/products/oplstudio/>.
15. Jørgensen, H.D. and Carlsen, S., Emergent Workflow: Planning and Performance of Process Instances. In *Proceedings of the 1999 Workflow Management Conf. – Workflow-based Applications (WFM'99)*, 98-116, Münster, Germany, Nov 1999.
16. Kammer, P.J., Bolcer, G.A., and Bergman, M., Requirements for Supporting Dynamic Adaptive Workflow on the WWW. In *Proceedings of the Workshop on Adaptive Workflow Systems at CSCW'98*, Seattle, WA, Nov 1998. On-line at <http://ccs.mit.edu/klein/cscw98/>.
17. Kinnock, N., EU Commissioner for Transport, Task Force Transport Intermodality Brochure, Oct 1995. On-line at <http://www.cordis.lu/transport/src/taskforce/src/intbrch2.htm>.
18. Kradolfer, M., A Workflow Metamodel Supporting Dynamic, Reuse-Based Model Evolution. *Doctoral thesis,* University of Zürich, Switzerland, 2000.
19. McClintock, C. and Berlioz, C.A., Implementing Business Rules in Java. In *Java Developers Journal*, 5(5):8-16, 2000.
20. McDermott, D., Ghallab, M., Howe, A., Knoblock, C., Ram, A., Veloso, M., Weld, D., and Wilkins, D., PDDL - The Planning Domain Definition Language. *Technical Report*, Department of Computer Science, Yale University, New Haven, CT, 1998.
21. Müller, R. and Rahm, E., Rule-Based Dynamic Modification of Workflows in a Medical Domain. In Buchmann, A.P., (Ed.), *Datenbanksysteme in Büro, Technik und Wissenschaft (BTW'99)*, Freiburg im Breisgau. Springer, 429-448, Berlin, 1999.
22. Reichert, M. and Dadam, P., ADEPTflex: Supporting Dynamic Changes of Workflow without Losing Control. *Journal of Intelligent Information Systems*, 10(2):93-129, 1998.
23. Sadiq, S.W. and Orlowska, M.E., Architectural Considerations in Systems Supporting Dynamic Workflow Modifications. In *Proceedings of the Workshop on Software Architectures for Business Process Management at CAiSE'99*, Heidelberg, Germany, Jun 1999. On-line at <http://www.itee.uq.edu.au/~shazia/Publications/CAISE99.pdf>.
24. Taleb-Ibrahimi, M., de Castilho, B., and Daganzo, C.F., Storage Space Versus Handling Work in Container Terminals. *Transportation Research B.* 27(1):13-32, 1993.
25. The Workflow Automation Corporation, Workflow Automation: New Opportunities for Dramatic IT Results. *White Paper*, 1998. On-line at <http://www.workflow.ca/>.
26. Trilling G., Génération automatique d'horaires de médecins de garde pour l'hôpital Côte-des-Neiges de Montréal. *CRT-98-05*, University of Montreal, Quebec, Canada, Jan 1998.
27. Tsang, E., Foundations of Constraint Satisfaction. *Academic Press*, London, 421 pp., 1993.
28. van der Aalst, W. and van Hee, K., Workflow Management: Models, Methods, and Systems. *The MIT Press*, 368 pp., 2002.
29. Weil, G., Heus, K., François, P., and Poujade, M., Constraint Programming for Nurse Scheduling. *Engineering in Medicine and Biology*, 14(4):417-422, 1995.
30. Workflow Management Coalition, The WfRM. *WFMC-TC-1003*, Version 1.1, Jan 1995.