Design and Implementation of Workflow Based Yard Management System

Dong Won Cho A, Young Hae Lee B and Hyun Jin Hwang C

A Hanyang University, S. Korea, e-mail: dwcmj@naver.com
B Hanyang University, S. Korea, e-mail: yhlee@hanyang.ac.kr
C Korea Institute for Defense Analyses, S. Korea, e-mail: sfsmile@empal.com

Abstract
In this paper, we address workflow management system for automatically operating yard management system which can fill up a gap between transportation and warehouse management system in supply chain management. But, it is difficult to find an optimal solution for yard operation problems because of its dynamic complexities. Instead, we propose workflow management system which describes the automation of yard’s operations, tasks and transactions. In the beginning, we describe and analyze yard management system and write down workflow pattern via worklist mechanism. Then, a business rule to efficiently and effectively operate dock allocation is proposed. Finally, workflow patterns are implemented by YAWL which is known as one among workflow specification languages. The proposed design and implementation can improve efficiency and visibility of yard management system.

Key words: Yard management system, Supply chain management, Workflow management system, YAWL

1. Introduction
In today’s fierce global business environment, companies are willing to invest in, and focus attention on, their supply chains in order to enhance their competitive edge. It has been reported that more than 6 million USD were invested in information systems concerned with supply chain management in 2008 [5]. Supply chain management has become a significant strategic tool for firms striving to enhance competitiveness, improve customer service, and increase profitability [13]. In other words, supply chain management has played an important role in innovatively improving organizational processes such as supplier relationship management, demand management, and order fulfillment.

However, interest in yard management has lagged compared with interest in other processes involved in supply chain management [10].

Therefore, yard management represents a supply chain bottleneck in terms of material and information flow. The Aberdeen Group recently surveyed the methods used by companies to manage their yard distribution and found that companies use one of three approaches. Fifty-eight percent of companies use the manual approach, involving clipboards and spreadsheets. Twenty-four percent of companies use the siloed approach, which involves using a commercial yard management system that is not integrated with the warehouse management system. Only 12% of companies use an integrated approach, in which the warehouse management system includes a yard management module, even though the integrated approach showed the best performance of all of the methods [1].

A yard management system, which can fill the gap between the transportation and warehouse management systems, is a dynamic component that can have broad positive impacts on supply chain management. Consequently, factors such as gate passage failure, dock misallocation, and container put-away delays can have a negative impact on supply chain performance. When process integration and collaboration with warehouse and/or transportation management systems are performed correctly, the use of yard management systems can allow companies to become formidable competitive entities, as customers can get exactly what they want within a shorter delivery time, thus benefiting everyone along the supply chain.

Identifying which yard management processes should be jointly managed is an important issue. To achieve successful yard management process integration and all of the associated benefits, decision makers associated with the yard management system must reach a shared understanding of the key yard management processes and construct a reasonable and efficient information system.

In this paper, the distribution operations of a warehouse yard are considered in terms of port operation management. Yard management systems involve gate management, dock management, and container management, as shown in Figure 1.

*Corresponding author
Components of yard management systems such as dock allocation, container pick-up, and gate management differ with port operation management. Generally, it is difficult to identify the optimal operation using mathematical programming because of the dynamic complexities of yard management systems. Recently, a real time location system for yard management was developed for a logistics yard, but gate and container management were not considered. Therefore, we design and implement a workflow management system (WfMS) to simplify and streamline current business processes by automating internal business operations, tasks, and transactions.

Workflow is the automation of a business process, in whole or in part, during which documents, information, or tasks are passed from one participant to another for action, according to a set of procedural rules [16]. Workflow management systems include some sort of software package for the implementation of a workflow system. The WfMS term is not a customized term for a specific business situation but instead refers to a universally applicable system. By configuring such a system, it can be modified to support a specific workflow. Therefore, unlike a workflow system, a workflow management system is a generic application [16]. Workflow management systems aim to automate the support and coordination of business processes to reduce costs and increase efficiency [12]. The key benefits of WfMS are improved efficiency, better process control, improved customer service, and business process improvement.

Workflow management systems have been applied to diverse industries including banking services, logistics, and manufacturing [15]. Ho et al. [6] proposed an intelligent information infrastructure that includes a two-step strategy that embraces the combination of online analytical processing and neural networks to support knowledge discovery; their infrastructure encompasses a rich collection of knowledge representation formalisms for dealing with a logistics information flow problem. Lau et al. [8] presented an intelligent logistics support system able to provide expert advice related to the airfreight forwarding business based on WfMS, thereby enhancing the logistics operations for relevant activities within the value chain of tasks. Dreiling et al. [4] proposed a method that aims to increase the efficiency of enterprise system implementation. These authors addressed the integration of existing process modeling and the configuration and elaboration of the enabling mechanisms. Leonardi et al. [9] proposed a service flow management system as a solution to information handling and communication requirement using WfMS YAWL (Yet Another Workflow Language). Jørgensen et al. [7] demonstrated how an informal task description of functional requirements for a new electronic patient record system could be translated into the YAWL workflow language.

In this paper, we design and implement a workflow management system for a yard management system. The developed WfMS can improve supply chain performance and customer service, provide better process control, and decrease work time. This paper is organized as follows. The next section describes the WfMS system developmental methodology and introduces the YAWL language. Section 3 describes and analyzes yard management processes and implements a yard management system using YAWL. Section 4 concludes the paper with a short summary.

2. WfMS developmental methodology and YAWL

2.1 WfMS development methodology
We apply rapid application development (RAD) to WfMS system developmental methodology. Rapid application development is a
method characterized by a cyclical, or iterative, development process [15]. In other words, the development phases are cycles repeatedly in small rapidly successive steps. Its main objectives are speed, cost reduction, and quality improvement with a high degree of user participation.

During the requirement planning phase, the intended goals of the project are established. Guidelines for the required functionality of the system are set, as are the requirements of the final products. Based upon the results to be achieved, the subsequent development roadmap is planned. During the architectural phase, the technical relationships between the individual components of the system are designed. During the user design phase, the system’s functionality is proposed. Its specifications are drawn up interactively with application design collaboration between the designer and the users. The users provide the input, which is recorded by the designer in the form of specifications. The designer can easily create specification prototypes with the aid of a program generator. The users can then test the specifications directly against the prototypes; this process is repeated several times. In traditional developmental methods, the design phase is clearly distinct from the construction phase, but this is not the case in RAD. During the construction phase, the details of the generated software are perfected and any elements that could not be produced automatically are made “by hand.” The users continue to validate the design during this phase. During the integration phase, the operations of the separate components are integrated and tested. This is a preparatory step that is mainly aimed at evaluating the technical compatibilities between the separate components before the system is handed over to the user for an acceptance test and enactment. During the delivery phase, the acceptance test is performed, and the system is prepared for production. This involves processes such as installation, any conversions that are necessary, and user training. For more extensive applications, a limited number of parallel designs and construction paths may be taken, keeping project management in mind.

2.2 YAWL

To implement a WiMS for a yard management system, we use YAWL, an open source workflow management system. The use of YAWL as a workflow language has two main merits [3,16]. First of all, YAWL is based on well-characterized Petri-net workflow patterns, which can easily express diverse processes. Second, YAWL allows for much more flexibility than do other workflow languages. Both expressiveness and flexibility are clearly important characteristics of a workflow language. Next, we briefly introduce the basic concepts in YAWL.

Despite the efforts of the Workflow Management Coalition, the workflow products developed over the past few years are not interactively compatible because the languages differ significantly in terms of concepts, constructs, and semantics. With the recent workflow patterns initiative, however, sets of the most relevant workflow patterns are now available, together with the relevant documentation. To overcome these limitations and make better use of the patterns, YAWL was developed through the rigorous analyses of existing workflow management systems and workflow languages, but it is a completely new language with independent semantics. Languages based on Petri-nets perform better with regard to state-based workflow patterns.

The YAWL system is highly expressive and flexible, provides direct support for all workflow patterns, has formal semantics, and offers graphical representations of many of its concepts [9]. The process editor in YAWL supports graphical modeling for processes and permits the visual editing of workflow definitions and the data they exploit, and the YAWL engine executes the workflows.

A graphical representation of YAWL’s modeling elements is provided in Fig. 3. A detailed workflow specification in YAWL is a set of extended workflow nets (EWF nets) that form a hierarchical and decomposite structure. Tasks may be atomic or composite, representing EWF nets at a lower level in the hierarchical structure. Each task can be instantiated multiple times using the concept of multiple instances for atomic or composite tasks. Conditions can be viewed as locations in a Petri-net. Every process definition starts with a unique input condition and ends with a unique output condition. Connections are possible between condition-task and task-task. AND, XOR, and OR split-and-join tasks are natively supported using the symbols shown in Fig. 3. Extensibility, which is an important feature of the YAWL system, allows for the interconnection of external applications and services using the workflow execution engine based on the service-oriented approach. In particular, the worklet dynamic process selection service for YAWL enables the replacement of a work item in a YAWL process with a dynamically selected worklet, thereby providing an established framework for workflow flexibility [9,18]. An extensible catalogue of worklets is maintained. Each time the service is invoked for a work item, a choice is made from the catalogue based on the data within the work item using a set of rules to determine the most appropriate substitution [9]. This service allows for dynamic exceptional change without having to resort to external off-system intervention or modification of the original process specifications.
3. WfMS-based yard management system

3.1 Yard management processes

A yard management system typically includes gate management, dock management, and container management. The hierarchical levels are designed...
and maintained to support yard managements of various complexities.

Figure 4 shows the yard management processes for a distribution or warehouse yard. The yard management system must proceed interactively between gate, dock, and container managements. Container management processes include algorithms that put-away or pick-up containers. Because dock allocation plays an important role in yard management systems, dock allocation algorithms need to be developed in line with the goals of yard management. In addition, business rules that result in an automatic decision concerning dock allocation after considering the dock situation need to be developed. These operation algorithms need to be included in the WfMS. In the remaining sections of the paper, we use the following notations:

\[(x, y, z)\] Container put-away/pick-up position in the yard,

\(n\) The number of docks with no waiting vehicles,

\(m\) The number of vehicles waiting for dock passage,

\(p_i\) Working time of vehicle \(i\) in dock \((i = 1, 2, \cdots, n)\),

\(q_i\) Waiting time of vehicle \(i\) in dock \((i = 1, 2, \cdots, n)\),

\(LB\) Lower bound of makespan for vehicles allocated in a dock,

\(d_j\) Makespan for vehicles allocated in dock \((j = 1, 2, \cdots, m)\),

\(T\) Set of no allocated vehicles in a dock,

\(TC\) Set of allocated vehicles in a dock,

\(A\) Set of docks belonging to \(d_j < LB\),

\(AC\) Set of docks belonging to \(d_j \geq LB\),

\(F\) Mean flowtime of vehicle in a dock,

\(QT\) Total waiting time of vehicles in a dock \((QT = \sum_i q_i)\).

### 3.2 Container put-away/pick-up algorithm

(1) Container put-away algorithm

Step 1. Find the positions that minimize \(x + y\).

Step 2. Put-away a container in a position among those selected in step 1 that minimizes \(z\).

(2) Container pick-up algorithm

Step 1. If the container to be picked-up is in the uppermost position, go to step 2 or step 3.

Step 2. Pick-up the container with the value \(z\). Go to step 6.

Step 3. If there exists another position into which a container can be put-away, go to step 4 or step 5.

Step 4. Pick-up the container with the \((x, y)\) value and put it away in the upper-most position. Go to step 6.

Step 5. After picking-up the container with the \((x, y)\) value and putting it away in the position equal to \((x, y)\), go to step 6.


### 3.3 Dock allocation algorithm

Because the majority of work in a transport distribution center occurs at the docks, dock allocation has a large impact on the efficiency of a yard management system. Thus, dock allocation needs to be efficient and flexible so that a yard management system can achieve its goals. Because dock allocation is NP-hard in the ordinary sense, several heuristics have been developed to solve this problem [11]. Several different objectives related to dock allocation can be considered. The three principle objectives are minimization of the makespan, total waiting time, and vehicle arrival order. As in the basic model, suppose there are \(m\) single-operation vehicles simultaneously available at time zero. Also suppose that there are \(n\) identical docks available for working, and that a vehicle can work at most one dock at a time.

(1) A-1 algorithm

Step 1. Construct a non-decreasing order for the working times of the vehicles.

Step 2. Calculate \(LB\) such that,

\[
LB := \left[ \max\left( \max_{i} \frac{\sum_{j} p_j}{m} \right) \right]
\]

Step 3. If \(T = \phi\), go to step 6. If \(T \neq \phi\) and \(A \neq \phi\), go to step 4. If \(T \neq \phi\) and \(A = \phi\), go to step 5.

Step 4. Allocate the first vehicle in the step 1 order into dock \(j\) and add the vehicle to \(TC\). Next, update \(d_j\). If \(d_j \geq LB\), add the dock to \(AC\). If \(T \neq \phi\), go to step 3.

Step 5. Allocate the vehicle to the dock with the minimum \(d_j\) among the docks belonging to \(AC\). Next, update \(d_j\). If \(T \neq \phi\), go to step 3.


For example, consider the following vehicle set in the case that \(n = 3\) docks are available.

Example 1. Schedule the vehicle by applying A-1 algorithm to Table 1.
(2) A-2 algorithm
Step 1. If $\phi = T$, go to step 3. If $\phi \neq T$, go to step 2.
Step 2. Allocate a vehicle to a dock based on the first-come, first-out rule. Update $d_j$. Then add the vehicle to $T^c$. If $\phi = \emptyset$, go to step 3. If $\phi \neq \emptyset$, repeat step 2.
Step 3. Stop.
Example 2. Schedule the vehicle by applying A-2 algorithm to Table 1.

(3) A-3 algorithm
Step 1. Construct an increasing order for the working times of the vehicles.
Step 2. Calculate $LB$ using equation 1.
Step 3. If $\phi = T$, go to step 6. If $\phi \neq T$ and $\phi \neq A$, go to step 4. If $\phi \neq T$ and $\phi = A$, go to step 5.
Step 4. Construct a non-decreasing order for $d_j$. Allocate the first vehicle in step 1 into the first dock in step 4 and add the vehicle to $A^c$ and go to step 3.
Step 5. Construct an increasing order for $d_j$ and a decreasing order for the working times of the vehicles. Allocate the first vehicle in step 5 to the first dock in step 5 and add the vehicle to $T^c$. Next, update $d_j$. Go to step 3.
Example 3. Schedule the vehicles by applying A-3 algorithm to Table 1.

Table 1. Vehicle arrival order and working time

<table>
<thead>
<tr>
<th>Vehicle number(VN)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Working time(WT)</td>
<td>10</td>
<td>12</td>
<td>5</td>
<td>8</td>
<td>7</td>
<td>3</td>
<td>5</td>
<td>15</td>
<td>12</td>
<td>77</td>
</tr>
</tbody>
</table>

Table 2. Result for applying A-1 algorithm to table 1

<table>
<thead>
<tr>
<th>dock 1</th>
<th>dock 2</th>
<th>dock 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>VN</td>
<td>WT</td>
<td>LB = 26</td>
</tr>
<tr>
<td>8</td>
<td>15</td>
<td>11</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>QT</td>
<td>94</td>
<td></td>
</tr>
</tbody>
</table>

Table 3. Result for applying A-2 algorithm to table 1

<table>
<thead>
<tr>
<th>dock 1</th>
<th>dock 2</th>
<th>dock 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>VN</td>
<td>WT</td>
<td>LB = 26</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>9</td>
<td>12</td>
<td>-3</td>
</tr>
<tr>
<td>QT</td>
<td>72</td>
<td></td>
</tr>
</tbody>
</table>

Table 4. Result for applying A-3 algorithm to table 1

<table>
<thead>
<tr>
<th>dock 1</th>
<th>dock 2</th>
<th>dock 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>VN</td>
<td>WT</td>
<td>LB = 26</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>23</td>
</tr>
<tr>
<td>8</td>
<td>15</td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>-2</td>
</tr>
<tr>
<td>QT</td>
<td>60</td>
<td></td>
</tr>
</tbody>
</table>

$QT$ is the total waiting time, and $F$ is the average waiting time.
3.4 Business rules for dock allocation

The selection of an appropriate algorithm for dock allocation depends on the choice of criterion. For this reason, dock allocation is perhaps as much the study of application methodologies as it is the study of algorithms. We will thus develop business rules for dock allocation to make this an automatic process with reasonable decision-making.

Business rules are loosely the rules or policies about a certain aspect of the business. They guide the activities of a business and govern changes in the state of the enterprise. Consequently, a business rule for dock allocation is a statement that defines or constrains some aspect of the dock. These rules are intended to impose business structure or to control or influence the behavior of the dock. Business rules can in general be divided into three major categories: terms, facts, and rules [18]. The terms and facts are the semantics behind the rules that will become the foundation for a logical data model and physical database, and perhaps for a business object model. A term is a noun or noun phrase with an agreed-upon definition, for example, a concept or a value, such as vehicle, dock, or working time. A fact is a statement that connects terms, through prepositions and verb phrases, into sensible, business-relevant observations, for example, ‘Vehicle arrives,’ ‘The order is for a line item.’ A rule is a declarative statement that applies logic or computation to information values. Terms and facts provide meaning and coherency to other kinds of requirements, especially rules. Therefore, rules build directly on terms and facts, expressed as if/then statements. The YAWL system uses XQuery or XPath instead of if/then statements to apply business rules to workflows. YAWL is the automation of a business process, in whole or part, according to a set of procedural rules. The business rules for dock allocation are shown in Fig. 5. The implementation procedure is as follows.

First, the waiting time of a vehicle for dock allocation is compared with \(WT\) arbitrarily decided. If the waiting time is more than \(WT\), apply A-2 algorithm. If the number of waiting vehicles is more than \(NWV\), apply A-1 algorithm. Otherwise, apply A-3 algorithm.

3.5 Implementation of a yard management system

YAWL is one of the few languages that completely relies on XML-based standards like XPath and XQuery. The process editor supports graphical modeling of the processes and permits visual editing of the workflow definitions and the data they exploit. The file that is stored in the process editor is an XML; this file performs the workflow execution in the workflow execution engine. During certain activities of a YAWL case, information for data elements that have been established in the XML specification may have to be supplied. The form submits the data back to YAWL only when all of the data in the forms are valid.

4. Conclusions

Although enterprises are willing to give attention to their supply chains, yard management systems are still relatively under-developed compared to other processes involved in supply chain management. Consequently, the yard management system is still a bottleneck process in the supply chain. The failure and delays of processes such as container pick-up, dock allocation for vehicles, and gate passage can have an adverse impact on supply chain performance. In effect, efficient process operation within a yard management system can enhance and improve enterprise competitiveness. In this paper, we designed and implemented a workflow management system for a yard management system using the workflow language YAWL. The
developed WfMS of the yard management system can improve the efficiency and visibility of the supply chain. In future studies, algorithms and business rules can be incorporated into yard management and implemented using YAWL. Furthermore, the process integration methodology between inbound and outbound traffic in yard management systems needs to be investigated.

References

Dong Won Cho is a PhD. Student in the Department Industrial & Management Engineering , Hanyang University, Korea. He obtained MS from Hanyang University, Korea, in 1999. His research interests include SCM, service science and discrete event simulation.

Young Hae Lee is a professor in the Department of Industrial & Management Engineering , Hanyang University, Korea, and a President of the Korean Society of Supply Chain Management. He received PhD from University of Illinois in 1986. His areas of interest interests are SCM, logistics and simulation optimization.

Hyun Jin Hwang is a research assistant in agency for defense development. He received Master Degree from Hanyang University in 2009. His interests are SCM and business processes engineering.