Integrated GIS-Based Logistics Process Monitoring Framework with Convenient Work Processing Environment for Smart Logistics

Yeong-Woong Yu, Hoon Jung, and Hyerim Bae

In today's highly competitive business environment, most companies try to manage their logistics function strategically to satisfy their orders as cost-effectively as possible and maximize their present and future profits. In this environment, logistics process visibility is essential to companies wishing to competently track components, parts, or products in transit from the original suppliers to the final destinations. Thus, it is important to provide instantly and easily recognizable information about such visibility to all stakeholders, especially customers. To ensure a high-level geographical visibility of the logistics processes, in this paper, we propose a way of implementing a GIS-based logistics process monitoring environment and of integrating a performer's work processing environment on the monitoring system. Additionally, to provide more abundant monitoring information, we describe a procedure for creating and processing various monitoring information, which are represented as ECA rules in this work processing environment. Therefore, it is possible to provide intuitive and visible monitoring information regarding the logistics process and resource elements.

Keywords: GIS-based BPM, logistics process, process monitoring, process visibility, smart logistics.

I. Introduction

In today's highly competitive environment, many companies are aiming to gain a share of the global market and take advantage of higher production and sourcing efficiencies. A key determinant of business performance nowadays is the role of the "logistics function" in ensuring a smooth flow of the materials, products, and information throughout a company's supply chains [1]. Most businesses aim to provide their customers with products or services in an effective and timely manner. Therefore, logistics is very important to business. In logistics systems, items are produced at one or more factories, shipped to warehouses for intermediate storage, and then shipped to retailers or customers [2].

According to IBM Center, in today's world, by improving the visibility of inventories, asset management, and information, logistics has emerged as a key cost control area in many organizations' supply chain operations. One of the primary purposes to ensure such visibility is to improve the internal decision-making ability and operational performance. Through ensuring the visibility from the supplier to the end user, greater visibility can also provide a clearer view of their entire supply chain [3]. More recently, logistics has become more prominent and has been recognized as a critical factor to competitive advantage [4]. Since logistics is related to all areas of the supply chain, a higher visibility in logistics is finally able to provide greater recognition regarding the whole supply chain.

Therefore, to ensure a high visibility of the logistics process, in this paper, we introduce two phases for the implementation of a GIS-based logistics process monitoring environment. In

---

Manuscript received Aug. 5, 2014; revised Jan. 27, 2015; accepted Feb. 3, 2015.

This work was supported by the Postal Technology R&D program of MSIP, Rep. of Korea (10039146, Development of Implementation Technology for SMART Post).

Yeong-Woong Yu (corresponding author, hero@etri.re.kr) and Hoon Jung (hoonjung@etri.re.kr) are with the IT Convergence Technology Research Laboratory, ETRI, Daejeon, Rep. of Korea.

Hyerim Bae (hrbae@pusan.ac.kr) is with the Department of Industrial Engineering, Pusan National University, Busan, Rep. of Korea.

© 2015 ETRI

http://dx.doi.org/10.4218/etrij.15.2314.0056

this framework, it is possible to provide intuitive and visible monitoring information about the logistics process and resource elements such as the products, humans, and vehicles within the process. This visible monitoring environment is integrated with a work processing environment and provided to performers to make their logistics environment smarter.

The remainder of this paper is organized as follows. First, we briefly review the background and related works in Section II. Section III introduces our approach to implementing a GIS-based logistics process monitoring environment and integrating a work processing environment on it. Next, Sections IV and V describe the implementations of our integrated framework in detail. In Section VI, we introduce our implemented prototype system and its applications. Finally, some concluding remarks are presented in Section VII.

II. Background and Related Works

1. Supply Chain Management (SCM) and Logistics

SCM is defined as the systemic, strategic coordination of traditional business functions and the tactics across such business functions within a particular company and across businesses within the supply chain [5]. The focus of SCM is the management of relationships to achieve a more profitable outcome for all parties along the chain [6]. In contrast to SCM, logistics is the work required to move and position inventory throughout the supply chain. As such, logistics is a subset of, and occurs within, the broader framework of a supply chain [7].

According to the Council of Supply Chain Management Professionals (CSCMP), logistics management is the process of planning, implementing, and controlling the efficient, cost-effective flow and storage of goods, services, and related information from the point of origin to point of consumption. A functional logistical process also relies on the proper geographical location of all assets within the organization. When a company successfully coordinates these logistical processes, the company can track the process through the stages of production, consumption, storage, and disposal [8].

2. Supply Chain Visibility

Supply chain visibility is the ability of parts, components, or products in transit to be tracked from the manufacturer to their final destination [9]. A visible supply chain will eliminate or avoid problems, as well as improve opportunities within a company. In today's world, we know problems are going to occur regardless of how well we plan. Therefore, companies have to be flexible and smart enough to deal with such problems as and when they arise. In this paper, we do not distinguish clearly between the supply chain and logistics, because SCM is a wider concept than logistics, and logistics is a part of the supply chain.

3. Related Works

This type of research, which is a strongly coupled technical convergence with business process management (BPM) and GIS technologies, has not yet been dealt with by many researchers. There have been a number of partially similar researches related to vehicle monitoring in a GIS environment. However, the main objective in most of them is to identify where the vehicles are on their routes.

There are many variations to the pick-up and delivery problem in the literature, which are related to the vehicle routing problem, vehicle fleet management, delivery with a time-window, and so on. The main concept of most of these is to find solution(s) satisfying the constraints given using various approaches. A few of them only show the results in a GIS layout. Our research does not intend to find solutions to the above problems. In addition, there have been a few researches related to the monitoring and control systems for parcel delivery. Most of them describe the use of location information for capturing the position, speed, and direction of the vehicles, and for an optimization of the routing by GPS, GIS, and so on.

In [10], although they use GIS, GPS, and optimization technologies to control the parcel delivery service and deal with pick-up and delivery monitoring, the authors focused on delivery sequence planning and did not consider the BPM perspective at all. While the research in [11] proposes a real-time trace and tracking system implementation in a logistics environment, the authors focus on the implementation of RFID in postal logistics. In ESRI's white paper [12], the authors utilize GIS technology for insurance claims processing and describe how GIS technology can be part of a more expedient claims process as a case study. Although GIS is applied to the claims process, it handles the organization, analysis, and planning of the data to manage the process but includes no modeled processes based on the BPM methodology. In other cases, there are map-based monitoring solutions, such as UPS's ORION and FedEx's SenseAware. However, ORION mainly focuses on a routing optimization, and SenseAware deals with various information by a sensor-based logistics in a supply chain.

III. Integrated Framework Approach

1. Applying GIS to Logistics Process Monitoring

BPM is a systematic approach to making an organization's workflow more effective and efficient. A business process is

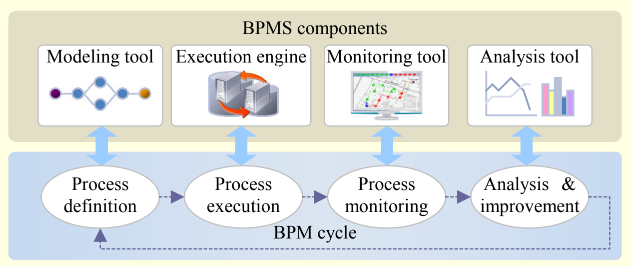


Fig. 1. BPM life cycle in BPMS.

an activity or set of activities that will accomplish a specific organizational goal [13]. A BPM system (BPMS) is one that can design, execute, monitor, analyze, and improve processes that are continuously managed in accordance with a BPM life cycle, such as the one shown in Fig. 1. Because the main objective of the process monitoring phase is to provide key information about the overall status of a process execution, by providing highly visible information, making users able to intuitively recognize and rapidly interpret the information are important factors of the process monitoring.

As previously mentioned, logistics is the one important function in business today. Raw material and products have always to be moved geographically. Considering these characteristics of the logistics process, we introduce a way of applying GIS technology into the logistics process monitoring environment not only to obtain geographical visibility of the logistics process itself but also a comprehensive flow of resources such as products, services, participants, and vehicles that move on the process.

GIS is a relatively broad term that can refer to a number of different technologies, processes, and methods [14]. Nowadays, through an integration of geographic/spatial information and existing diverse business information, the information is being widely used for a spatial-based analysis, providing information, and decision-making in many fields. For these reasons, GIS technology can be the foundation for many location-enabled applications.

2. Work Processing on GIS-Based Logistics Process Monitoring System

Next, we propose an integrated architecture that enables users to intuitively recognize the current status of a process and to process their tasks in the logistics process monitoring environment mentioned earlier. In a BPM environment, most of the monitoring information provided to companies or customers is acquired while users execute their tasks in a work list. A work list is a list of work items related to the process participants. Work items are represented tasks to be processed in process instances by the participants.

When a defined process is initiated, a process instance is created. After that, the process progresses through a repetition of the creation and processing of activity instances. Each activity instance is automatically assigned to a relevant performer's work list by a process engine and is then processed by that performer. The performers process the work items by starting and completing them. Thus, we introduce our system architecture, which is capable of providing a work processing environment in which the work items are able to be automatically identified and processed based on the performer's location through the mapping of geographical location information to each work item, similar to the way of ensuring geographical visibility of a process.

3. Problem Statements

The scopes of the supply chain and logistics are very wide. According to [6], there are a lot of complexities to the network, process, scope, products, customers, suppliers, business, and information within a supply chain. Considering these kind of complexities, we deal with the delivery process, one of the processes in the supply-chain operations reference model [15], in which geographical movement and the flow of elements such as products and resources are important.

This process is generally in the form of forward logistics. However, in the real world, reverse logistics can be included in the forward logistics, because if customers want to return their goods to their point of origin, then the goods should be withdrawn from them. Therefore, we focus particularly on the pick-up and delivery processes, which mainly consist of tasks such as delivery, pick-up, and movement. In addition, it has a structural characteristic in that the tasks are repetitive. To design a pick-up and delivery process, the process is conceptually formed through activities and links connecting the activities. Formally, the process definition is as follows [16].

Definition 1 (Process). A *process* p is defined as a tuple, $<A, L>$, where $A = \{a_p \mid p = 0, 1, 2, \ldots, I\}$ is the set of activities, and $L = \{l_{pq} = (a_p, a_q) \mid p, q \in I\}$ is the set of links formed by pairs of activities, each of which, for example, $l_{pq} \in L$ indicates that the pth activity immediately precedes the qth activity.

A link is a good representation of movement, because a link indicates a connection of activities and their precedence

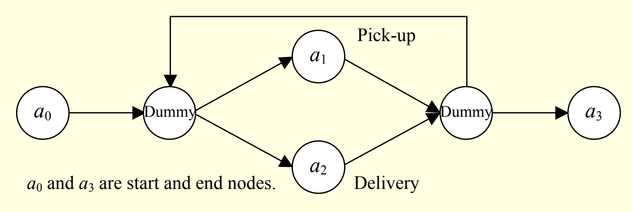
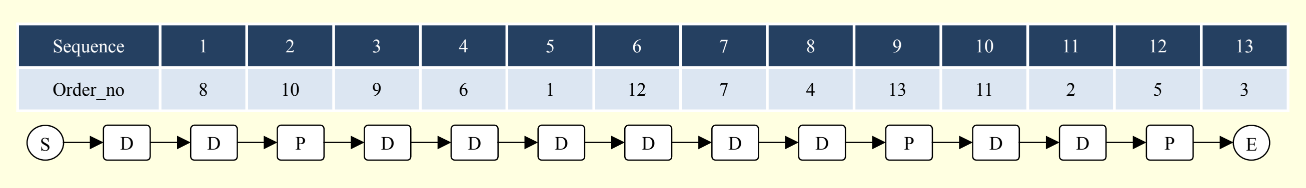


Fig. 2. Example of a pick-up and delivery process model.

Table 1. Example of an order list to be handled by a performer in a day.

No	custName	Address	orderType	Item	Coordinates	Sequence
1	Kim	39 Gyeryong-ro 16beon-gil, Yuseong-gu, Daejeon	Delivery	CD	POINT(x,y)	5
2	Lee	111 Yuseong-daero 668beon-gil, Yuseong-gu, Daejeon	Delivery	Kimchi	POINT(x,y)	11
3	Park	39 Gyeryong-ro 88beon-gil, Yuseong-gu, Daejeon	Pick-up	Toy	POINT(x,y)	13
4	Choi	41-7 Yuseong-daero 626beon-gil, Yuseong-gu, Daejeon	Delivery	Water	POINT(x,y)	8
5	Jang	59 Gyeryong-ro 88beon-gil, Yuseong-gu, Daejeon	Delivery	Clothes	POINT(x,y)	12
6	Bae	47 Yuseong-daero 694beon-gil, Yuseong-gu, Daejeon	Delivery	Watch	POINT(x,y)	4
7	Jo	57 World cup-daero 253beon-gil, Yuseong-gu, Daejeon	Delivery	Ticket	POINT(x,y)	7
8	Jeong	111 Yuseong-daero 694beon-gil, Yuseong-gu, Daejeon	Delivery	Note	POINT(x,y)	1
9	Yu	43 Gyeryong-ro 60beon-gil, Yuseong-gu, Daejeon	Pick-up	Chair	POINT(x,y)	3
10	Yu	43 Gyeryong-ro 60beon-gil, Yuseong-gu, Daejeon	Delivery	Book	POINT(x,y)	2
11	Kang	46 World cup-daero 289beon-gil, Yuseong-gu, Daejeon	Pick-up	Cup	POINT(x,y)	10
12	Han	69 Gyeryong-ro 18beon-gil, Yuseong-gu, Daejeon	Delivery	PC	POINT(x,y)	6
13	Moon	55 World cup-daero 253beonan-gil, Yuseong-gu, Daejeon	Delivery	Phone	POINT(x,y)	9



Sequence	1	2	3	4	5	6	7	8	9	10	11	12	13
Order_no	8	10	9	6	1	12	7	4	13	11	2	5	3

Fig. 3. Order sequences for order list and its process instance.

relation. Thus, we regard the movement tasks in the pick-up and delivery process to be links between two activities. Process modeling may vary based on the different points of view of the process designers. A simple example of the pick-up and delivery process modeled by the author is shown in Fig. 2.

Table 1 shows an example of an order list for a performer. The performer visits places to process the orders given and handles the pick-up or delivery tasks using a mobile device (PDA or smartphone). The results are then sent to the system. At this time, a performer moves according to the optimum (or shortest) path for visiting all locations and returns to the origin.

The label "Sequence" in Table 1 represents the visiting orders based on the path. The coordinates and sequence fields will be filled out after geocoding (in phase 1, later). If no problems occur while the process is being executed, then the actual process instance will be like that shown in Fig. 3.

A process instance is created by an actual initiation of the modeled process. In general process monitoring, the whole structure of a process instance is not determined until the completion of the process. In the pick-up and delivery process described in this paper, an optimal path to processing all orders assigned to a performer should be determined before the activities are begun. Because activity instances are created by a corresponding one-to-one relationship to the orders, and the performers execute their process alone, the process instance structure can therefore also be determined in advance before the pick-up and delivery process is executed.

If N is the number of orders to be processed by a performer in a day, then the process instance is defined as follows.

Definition 2 (Process Instance). A *process instance* is defined as a directed graph $p' = \langle A', L' \rangle$, where $A' = \{a'_i \mid i = 0, 1, 2, \dots, N+1\}$ is the set of activity instances, and $L' = \{l'_{ij} = (a'_i, a'_j) \mid 0 \le i \le N, j = i+1\}$ is the set of link instances formed by pairs of activity instances. In this definition, a'_0 and a'_{N+1} represent the start and end nodes in a process instance, respectively. Then l'_{01} and $l'_{N,N+1}$ represent movements from the origin to the first node, and from the last node to the origin, respectively. The ith activity instance immediately preceding the jth activity instance is represented by l'_{ij}. The jth means the next location to move to, after the ith activity instance is finished.

In graph theory, the number of incoming edges to a vertex is called the in-degree of the vertex, written as *in_degree*(v), and the number of outgoing edges from the vertex is its out-degree, written as $\mathit{out\_degree}(v)$. In the pick-up and delivery process in this paper, all activity instances included in a process instance are processed by a performer. Therefore, for any activity instance $a'_i \in A'$, its $\mathit{out\_degree}(a'_i)$ and $\mathit{in\_degree}(a'_i)$ are equal to one, except for the start and end activity instances. This means that all activity instances are processed sequentially with no split and merge structures. The subject of logistics process monitoring discussed in this study is focused on processing the instance exactly. Comparing Figs. 2 and 3, we can see that the process instance is more efficient than the process itself for monitoring the specific status of the process and providing monitoring information in detail.

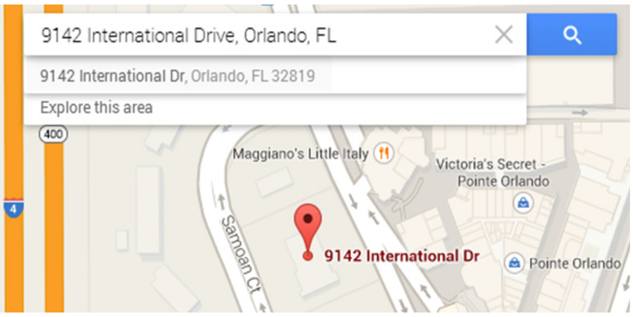
After a process instance is created and executed at a specific point in time, monitoring of the whole process and the status of the activities is important to both companies and customers. For orders to be processed in a specific area, companies need to know who executed it, what equipment is being operated, which task is being processed, and so on. Customers also want to know intuitively the current location of the performer and the time when they will receive their products. Delivery information from existing systems related to logistics is provided at a low level. Nevertheless, the overall status information, and in particular the geographic visibility, is not sufficiently high.

For an innovation of the poor and restricted logistics process monitoring environment, we ensure the geographical visibility for the logistics process and concurrently propose a framework that integrates the work processing environment with the monitoring environment. First, using the order data, we create activity instances corresponding to the orders and link instances, representing the connections between the activity instances as spatial data that can be represented in a GIS environment.

IV. Implementation of GIS-Based Logistics Process Monitoring Framework

Our proposed method of creating spatial data for process instances and projecting them on a map is a simple idea to geographically represent and visualize the status of the executed process instance. In the GIS field and spatial database systems, there are two basic spatial data types representing objects in the real world, raster and vector. In a vector data model, objects are simply represented as points, lines, or polygons. Vector data is stored and managed based on *geometry* type (in place of feature type) in a spatial database.

According to Definition 2, a process instance consists of sets of activity and link instances. By creating the geometry data of these two kinds of elements, the data can include location information. Therefore, each activity and link instance can represent an order's location and the path between two locations, respectively. The geometry data of this process instance can be projected onto a map. Every performer who deals with a pick-up or delivery task has their own area of responsibility, and every order occurring in the area has its address for the delivery or pick-up of the products. A specific location can be represented in various ways, such as by an address or coordinates. It is possible to map an address to the *x*- and *y*-coordinate representing the location in a geographical space, or to a building at the address.


9142 International Drive, Orlando, FL

9142 International Dr, Orlando, FL 32819

Fig. 4. Example of geocoding on Google Maps.

Geocoding is the process of assigning an *x-y* coordinate pair to the description of a location by comparing the descriptive location-specific elements to those in the reference data. The geocoding process is defined as the steps involved in translating an address entry, searching for the address in the reference data, and delivering the best candidate or candidates as a point feature on the map. Positional accuracy indicates how close each geocoded point is to the "true" location of the address [17]. Figure 4 illustrates an example of finding the location of a specific address. In this paper the quality of the geocoding is not a focus, and we assume that the results of geocoding are accurate. The locations for given orders are determined, and then the visiting sequence should be also determined. To find the optimal path to visit all locations and return is essentially the same type of problem as the traveling salesman problem, for which various solutions exist.

In this study, when the locations as geocoded results are determined according to the addresses for orders given, its optimal path is assumed to be given. PostGIS [18] supports ways to achieve this linestring geometry to visit all nodes. The following functions for geocoding and ordering of a sequence are used in the succeeding sections:

- geometry geoCoding(address): A function to transform an address into *x-y* coordinates. It returns a geometry that represents a proper location including a point geometry.
- array sortOrders([orders]): A function to sort orders in sequence by the optimal path. It returns a sorted list, O^S.

The steps for the creation of the sets of activity and link instances and of the geometry data are described below.

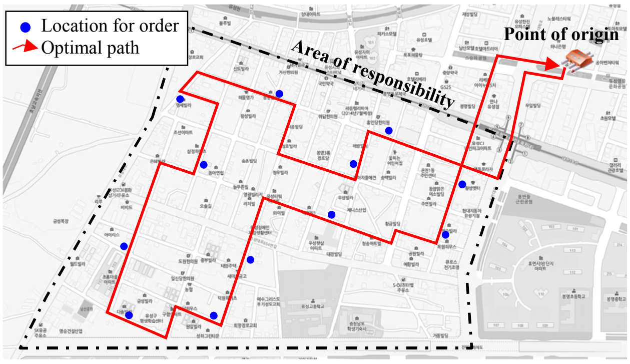


Fig. 5. Order locations and corresponding optimal path.

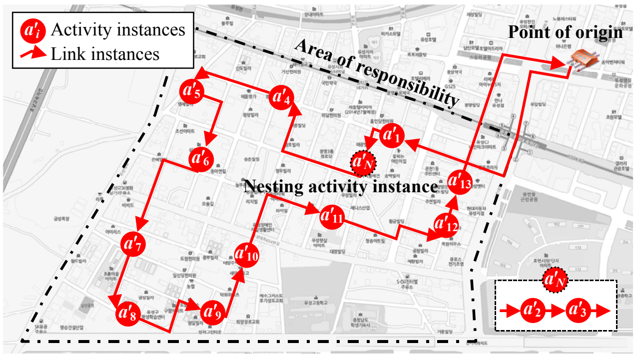


Fig. 6. Projecting an example of a process instance on a map.

1. Phase I: Creation of Activity Instance Geometry

A. Transforming Address to Coordinates Using Geocoding

If the number of given orders is N, then an order list, O, is created. This order list is an array of length $N+2$. Each order in O has a respective address. The result of geocoding using addresses in the given order list $O = [1, 2, \ldots, 13]$ can be obtained as follows. The geographic coordinates of the ith order (that is, of $O[i]$.coordinates) are transformed by geocoding and are represented as the geometry data of a form of POINT(x, y).

$$O[i].\text{coordinates} = \text{geoCoding}(O[i].\text{address}) \qquad \text{for all } i \in N.$$

B. Determining Visiting Sequence for Orders

After determining all locations for all orders in O, the sequence of visiting should be determined. The sorted order list O^S = sortOrders(O) according to the visiting sequence can be obtained. The route by this sequence is the optimal path, optPath, and it is assumed to be given. To process orders, a performer moves along optPath in order. This visiting sequence is the same as the processing sequence of its process instance. Figure 5 illustrates the locations for all orders given and optPath to visit the locations and return to the origin. The number of locations is 12, which is less than 13 orders because both orders 9 and 10 were for the same location. However, the two orders overlap each other on the map.

C. Creating Activity Set by Sorted Order List

Each order in O^S is transformed into an activity instance to be processed during the pick-up and delivery process. Thus, the set of activity instances corresponding to O^S is created to compose a process instance, as illustrated in Fig. 6. Because an order represents an activity instance, the number of elements in the set of activity instances $A' = \{a'_0, a'_1, a'_2, \ldots, a'_{14}\}$ is $N+2$ including the start and end activity instances a'_0 and a'_{14}. The ith order $O^S[i]$ corresponds to a'_i, and the coordinates of a'_i indicate the address of $O^S[i]$. The 13 orders consist of ten delivery and three pick-up activities. Every order has an orderType for a delivery or pick-up. Therefore, the activity types can be distinguished by this orderType. Each instance in A' is labeled by orderType and can be created as follows:

$$a'_i.\text{actInstName} = O^S[i].\text{orderType} \qquad \text{for all } a'_i \in A'.$$

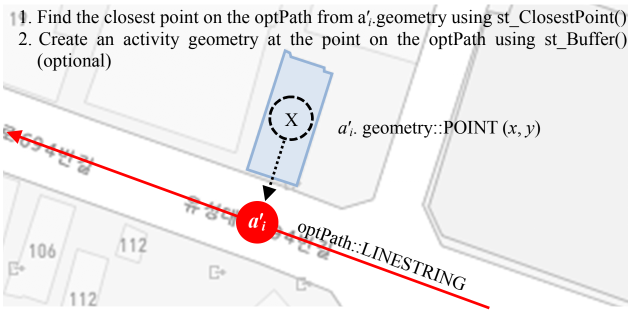
D. Creating Activity Instance Geometry

The set of activity instances A' was created using the order information. Because each activity instance should be created as spatial data to be represented on a map, we create their point-type geometries. The coordinates of a'_i to be located on a map are from the geocoding result, $O^S[i]$.coordinates. The point-type geometry can be suitably adjusted using a Styled Layer Descriptor to control the visual portrayal of the geospatial data. Regardless of map scale, a fixed size of the geometry can be also created by using the st_Buffer() function.

$$a'_i.\text{geometry::POINT} = O^S[i].\text{coordinates} \qquad \text{for all } a'_i \in A',$$
$$a'_i.\text{geometry::POLYGON} = \text{st\_Buffer}(a'_i.\text{geometry, size}).$$

2. Phase II: Creation of Link Instance Geometry

In this phase, we create the geometries of the link instances connecting the activity instances in the previous phase. The geometry of link instance l'_{ij} representing movement between two locations can be created using the optPath. Using the location information of two activity instances a'_i and a'_j on the optPath, l'_{ij} between a'_i and a'_j can be extracted by the segmentation of optPath. A specific line segment from a geometry of linestring type is able to be extracted using the st_LineSubstring() function with two points (these two points must be placed on the line). However, the geographical coordinates of the orders $O^S[i]$.coordinates and geometries of its activity instances a'_i.geometry are not exactly placed on the optPath, because the geocoding result generally indicates a building at an address, as shown in Fig. 4. Therefore, a step used to adjust the space between the building and path is needed by moving a'_i.geometry onto the optPath.



1. Find the closest point on the optPath from a'_i.geometry using st_ClosestPoint()
2. Create an activity geometry at the point on the optPath using st_Buffer() (optional)

Fig. 7. Adjusting location of an activity instance onto optPath.

A. Location Adjustment of Activity Instance Geometry

The a'_i.geometry is indicated as "X" in Fig. 7. Thus, to move the geometry on the building onto the optPath, we find a point on the optPath that is closest to "X" using the st_ClosestPoint() function. Strictly speaking, a new geometry of a'_i is created at the point where a'_i needs to move.

a'_i.geometry = st_ClosestPoint(optPath, a'_i.geometry) for all $a'_i \in A'$.

If activity instance geometries are newly created on the optPath, then they can better represent a process structure. The location-adjusted result is shown in Fig. 7. Through moving a'_i.geometry to the closest point on the optPath being perpendicular to the point, the new a'_i.geometry can be well-positioned on the optPath. In addition, these can be used as the basis points to create l'_{ij}.geometry.

B. Creating Link Instance Geometry

The geometries of a set of link instances L' are able to be created using the geographically adjusted activity instance geometries. Every l'_{ij}.geometry from optPath can be extracted, as previously mentioned.

l'_{ij}.geometry = st_LineSubstring(optPath, startfraction, endfraction) for all $l'_{ij} \in L'$.

Here, two arguments of the above function, startfraction, denoted as *sfraction*, and endfraction, denoted as *efraction*, indicate ratios related to the locations of the start and end points of l'_{ij} on the optPath. The two arguments take values between zero and one, where such values are known as *float* values. As shown in Fig. 8, to extract and create l'_{01}.geometry, we first need to know the values of both ratios regarding the location of its starting and ending points for the total length of optPath.

The sfraction of l'_{01} is zero, and the efraction of it is the ratio that is the length from the starting point to a'_i divided by the total length of the optPath. Hence, the fractions from the starting point of optPath to each activity instance a'_1, a'_2, and a'_3 are needed to create four l'_{ij}, as shown in Fig. 8. If optPath.length is the total length of the optPath and the length between two specific locations of a'_i and a'_j is denoted by

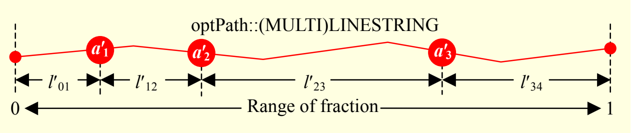


Fig. 8. Illustration of extracting line segments for link instance geometry.

length(a'_i, a'_j), then the length from the starting point to each a'_j can be calculated as follows:

$$\text{length}(a'_0, a'_j) = \sum_{i=0}^{j-1} \sum_{j=i+1}^{j} l'_{ij}.\text{length}\,. \tag{1}$$

Thus, $\forall l'_{ij} \in L'$, the *sfraction* and *efraction* of l'_{ij} can be written as follows:

$$l'_{ij}.\text{sfraction} = \begin{cases} 0 & i = 0, \\ \dfrac{\sum_{i=0}^{i-1} \sum_{j=i+1}^{j-1} l'_{ij}.\text{length}}{\text{optPath.length}} & \text{otherwise.} \end{cases} \tag{2}$$

$$l'_{ij}.\text{efraction} = \begin{cases} 1 & i = N, \\ \dfrac{\sum_{i=0}^{i} \sum_{j=i+1}^{j} l'_{ij}.\text{length}}{\text{optPath.length}} & \text{otherwise.} \end{cases} \tag{3}$$

Here, an ending point of any link is equal to the starting point of the succeeding link; this is denoted by the following:

$$l'_{ij}.\text{efraction} = l'_{j,j+1}.\text{sfraction}. \tag{4}$$

Therefore, to create four geometries, it is possible to use only the efractions of l'_{01}, l'_{12}, and l'_{23}, with "0" and "1" in Fig. 8 respectively representing the start and end points of optPath. Both fractions can be conveniently computed using the spatial function st_LineLocationPoint(), as below.

l'_{ij}.sfraction = st_LineLocationPoint(optPath, a'_i.geometry),
l'_{ij}.efraction = st_LineLocationPoint(optPath, a'_j.geometry).

Thus, all geometries of l'_{ij} can be created as follows:

l'_{ij}.*geometry* =
st_LineSubstring(optPath, l'_{ij}.sfraction, l'_{ij}.efraction).

We created the geometry data for the sets of activity and link instances through the two phases above. Because the activity and link instances representing the given orders and routes can be laid on a map in place, it is possible to intuitively recognize the whole process.

3. Visualization of Process Instance Projection on Map

Through the two phases above, we created spatial data for the sets of activity and link instances A' and L'. These can be projected onto a map. The results are shown in Fig. 6.

V. Integration of Work Processing Environment on GIS-based Process Monitoring System

We next describe an integrated framework with a work

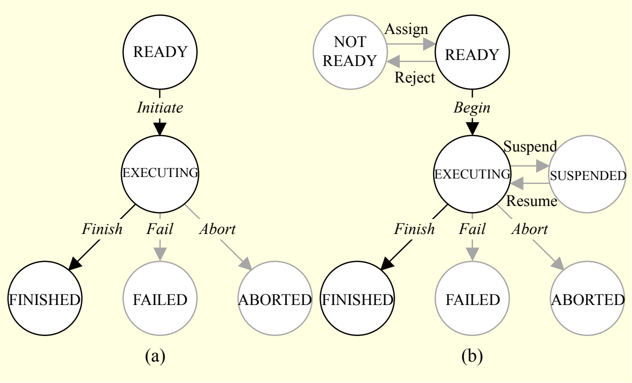


Fig. 9. State-transition model [21]: (a) process state-transition model and (b) activity state-transition model.

processing environment on the monitoring system that is capable of processing tasks directly. Performers can access their work items and handle them conveniently by clicking the activity instances on our monitoring screen. The procedure for processing the activities by directly accessing the work items is introduced using a state-transition model. The procedure can be easily understood by describing the creation and processing of various monitoring information using the ECA rule [19]–[20] connected with this state-transition model.

1. State-Transition Model for Process and Activity Instances

The process and activity (that is, the principal BPMS-managed objects) change their states while business processes are executed. Figures 9(a) and 9(b) illustrate the state transitions of the process and activity, respectively.

A process, after being designed and deployed, remains in a READY state, and an authorized user can execute it by generating an *Initiate* event and transitioning the process into an EXECUTING state. If the process is completed, then it enters a FINISHED state. Or, as a result of errors, the process can be in a FAILED state or be ABORTED by users.

The state-transition model of an activity is more complex than that of a process. Let us consider a task in a process. When all of its preceding tasks are complete, it is assigned to a user. If the user accepts the task, then an EXECUTING state is entered. After the execution, the activity state model enters one of three completed states, which are the same as those for the process.

The events mainly occurring from the principal objects process and activity are described in [22]. Our research focuses on events related to normal execution states such as READY, EXECUTING, and FINISHED.

2. ECA Rules for Monitoring Information

In our GIS-based logistics process monitoring environment, we attempt to integrate the performer's work processing environment with the monitoring system. Various monitoring information is created and controlled while a performer is executing a process instance. To provide plentiful monitoring information, we describe the procedure of creating and controlling the monitoring data using the ECA rule.

When a process instance is created, a variety of values may be set before a real execution starts. Every activity has its expected execution time (EET) by its type, and it can be calculated using the average execution time of the activities that have already been executed. In addition, every link has its expected movement time (EMT), which can be computed by its distance and average speed. Using both the EET and the EMT, the expected arrival time (EAT) can be calculated and updated according to the current time at the intervals of the beginning or finishing of the activities. These types of monitoring information are important, and EAT is particularly useful for customers.

EET of a'_i = average execution time by activity types,
EMT of $l'_{ij} = l'_{ij}$.length / average speed,
EAT from current location a'_c to a'_i =

$$\text{current time} + \sum_{i=c+1}^{i-1} a'_i.\text{EET} + \sum_{i=c}^{i-1} \sum_{j=c+1}^{i} l'_{ij}.\text{EMT}.$$

When a process instance is created, a performer departs from an origin to process the performer's own tasks. In addition, the performer returns to the origin after finishing all tasks, and the process instance is then finished. The departure and arrival times from/to an origin are the beginning and finishing times of a process instance and are denoted as initiatedTime and finishedTime, respectively. Similarly, every activity instance has its beginning and finishing time denoted as startTime and finishedTime, which indicate the arrival time from a'_{i-1} to a'_i and departure time from a'_i to a'_{i+1}, respectively. These times can be immediately captured using available functions for the time value processing, denoted as currentTime::timestamp=now(), when related events occur. The occurrence times of the beginning and finishing events of the process and activity instances can be represented using the ECA rule as follows.

Rule (R-A: timestamping currentTime)	
ON	p'.Initiate ∨ p'.Finish ∨ a'_i.Begin ∨ a'_i.Finish
IF	for each event
DO	{p'.initiatedTime, p'.finishedTime, a'_i.startTime, a'_i.finishedTime } = currentTime
EndRule	

Initially, the EET of the activity instances and EMT of the link instances are set in advance before a process instance is initiated. However, not every task is executed according to

http://dx.doi.org/10.4218/etrij.15.2314.0056

these expected times in the real world. Therefore, when a performer arrives to process a'_i at a location and a'_i.Begin occurs, the movement time from the previous (i–1)th location to the ith, $l'_{i-1,i}$.movement_time, can be computed exactly. In addition, if the execution of the current activity instance is finished and a'_i.Finish has occurred, then we are able to know the actual execution time, a'_i.execution_time.

Rule (R-B1: storing actual movement_time from a'_{i-1} to a'_i)	
ON	a'_i.Begin
IF	$l'_{i-1,i}$.state = 'PASSED' $\wedge$ a'_i.state = 'EXECUTING'
DO	$l'_{i-1,i}$.movement_time = currentTime − a'_{i-1}.finishedTime
EndRule	

Rule (R-B2: storing actual execution_time for current activity a'_i)	
ON	a'_i.Finish
IF	$l'_{i-1,i}$.state = 'ONGOING' $\wedge$ a'_i.state = 'FINISHED'
DO	$l'_{i-1,i}$.movement_time = currentTime − a'_i.startTime
EndRule	

These setting times at the beginning of a process, EET and EMT, are frequently changed to times consumed during actual executions and movements while a performer handles their tasks and moves repeatedly. Therefore, when the performer arrives at the location of a'_i or leaves a'_i, then EAT should be newly updated. No matter when a performer arrives, leaves, or both, the times can be updated. The problem then is when to update EAT. This is a problem that is related to the frequency or accuracy of providing the monitoring information. The updating rules can be represented as follows.

Rule (R-C1: updating new EAT when a performer arrives at a'_c)	
ON	a'_c.Begin
IF	a'_i.state = ¬'FINISHED' $\vee$ ¬'EXECUTING', $\forall i, i > c$
DO	$a'_i.\text{EAT} = \text{currentTime} + \sum_{i=c}^{i-1} a'_i.\text{EET} + \sum_{i=c}^{i-1} \sum_{j=c+1}^{i} l'_{ij}.\text{EAT}$
EndRule	

Rule (R-C2: updating new EAT when a performer departs at a'_c)	
ON	a'_c.Finish
IF	a'_i.state = ¬'FINISHED' $\vee$ ¬'EXECUTING', $\forall i, i > c$
DO	$a'_i.\text{EAT} = \text{currentTime} + \sum_{i=c+1}^{i-1} a'_i.\text{EET} + \sum_{i=c}^{i-1} \sum_{j=c+1}^{i} l'_{ij}.\text{EAT}$
EndRule	

The difference between rule R-C1 and R-C2 is whether the EET of the current activity is included when calculating EAT. This means that the accuracy of EAT in locations closer to the performer is higher. Finally, the process is finished and the last movement_time is computed, as in rule R-B1.

Rule (R-D: completing a process instance)	
ON	p'.Finish
IF	p'.state = 'FINISHED'
DO	**Rule R-B1**
EndRule	

VI. Implementation of Prototype System

1. System Architecture and Result

A GIS-based monitoring environment is applied to the pick-up and delivery process, and the work processing environment was integrated into our proposed monitoring system. We need BPMS to manage the processes adopted with an actual BPM methodology, including a process modeling tool to design the process models and a process execution engine to actually carry out the modeled process. However, in this study, we assume that BPMS exists and that our implementation scope is limited to a logistic process monitoring system and work processing environment.

The overall structure of the implemented system is shown in Fig. 10. We used PostgreSQL (v9.1) and PostGIS (v1.5) as a spatial database and GeoServer (v2.1) as a map/feature server. The prototype was implemented using JavaScript and Openlayers (v2.12) and operated in a web browser. An example screen shot of our prototype is shown in Fig. 11.

When the general process instance shown at the top of Fig. 11 is projected onto the map shown in the same figure, we can see that it is an excellent monitoring environment with geographical visibility. While the processes are being executed and controlled automatically by a process engine, a monitoring system provides information related to its executions and various states. In Fig. 11, the execution states of a process instance based on a performer's current location can be briefly distinguishable by three colors.

When there are no exceptional situations during the process execution, its states are classified according to the activity states. Activity instances to be conducted are first red, and when executed, they turn to blue, and then green after finishing their execution normally. The path is also expressed in red or green for NOT PASSED and PASSED, respectively. The path that a performer is following is shown by a blue dotted line.

2. Rich Monitoring Information and Applications

In our monitoring environment based on geographical visibility, more abundant information can be provided along with general information. Plentiful monitoring information from our proposed framework is classified according to the perspectives of the enterprise, customer, and performer and is thus more meaningful and useful. In addition, we can consider

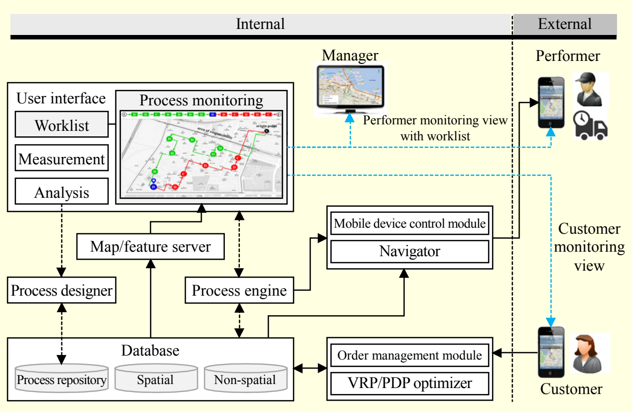


Fig. 10. Prototype system architecture.

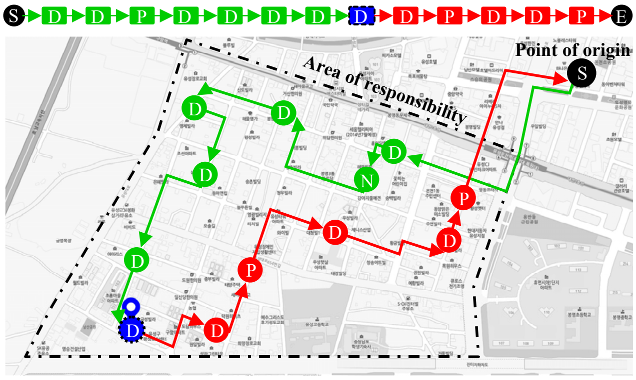


Fig. 11. Example of states of a process instance.

Fig. 12. Example of a window of mobile device.

several ways to obtain a lot of useful monitoring information. For example, applying a relational spatio-temporal data model [23] in our framework may be a good application for managing various spatial events.

Additionally, it is possible to utilize our innovative framework as the foundation to various applications. One example is to integrate a mobile navigation system into our framework. Figure 12 shows an expanded shot of our monitoring screen. If the navigation function is integrated in this screen, then a lot of useful information can be provided to the performers. This function can continuously provide such useful information until the performers arrive at their next locations. An additional advantage is that an unnecessary loss of time and effort, such as checking the next location and the path to arrive there, can be removed.

VII. Conclusion

To achieve a smart logistics in logistics process monitoring, we introduced an integrated GIS-based logistics process monitoring framework that included a work processing environment. With this main contribution, we first ensured the geographical visibility of the pick-up and delivery process, thereby making it possible to provide higher-level monitoring information. In addition, the integrated work processing environment is represented through ECA rules that can make performers identify, access, and process their work items in an easier and smarter way according to their location.

Although it is not easy to quantitatively measure the efficiency and availability of our GIS-based monitoring environment, the effect is clear in a logistic environment and in those processes that need to consider the location information related to a geographical movement and flow. In a broad sense, it is possible to establish a systematic logistics process management environment and an agile supply chain to identify predictable exceptional and risky problems and to respond and react to them properly in our environment. Therefore, smart logistics may be realizable in our framework owing to improvements in both the logistics competitiveness of enterprises and the ability to utilize monitoring information as a basis for fast decision-making. Finally, *visualization* in the Internet of Things (IoT) is critical, because it allows users to better interact with the objects in their environment [24]. In summary, our research is a good example of technical convergence with BPM and GIS, and as the IoT environment changes over time, the processes that were used in our framework will be able to visually interact with a wider variety of objects.

References

[1] C.-C. Sum, C.-B. Teo, and K.-K. Ng, “Strategic Logistics Management in Singapore,” *Int. J. Operations Production Manag.*, vol. 21, no. 9, 2001, pp. 1239–1260.

[2] D. Simchi-Levi, X. Chen, and J. Bramel, “*The Logic of Logistics: Theory, Algorithms, and Applications for Logistics and Supply Chain Management*,” 2nd ed., NY, USA: Springer-Verlag, 2010.

[3] IBM Center for The Business of Government, *Smart Logistics Part 1 of 3: Visibility-The Importance of Seeing Your Complete Supply Chain.* Accessed May 25, 2014. http://www.businessofgovernment.org

[4] D.J. Bowersox and D.J. Closs, “*Logistical Management: The Integrated Supply Chain Process*,” NY, USA: McGraw-Hill Companies, 1996.

[5] J.T. Mentzer et al., “Defining Supply Chain Management,” *J. Business Logistics*, vol. 22, no. 2, 2001, pp. 1–25.

[6] M. Christopher, "*Logistics and Supply Chain Management*," 4th ed., NJ, USA: FT Prentice Hall, 2013.

[7] D.J. Bowersox, D.J. Closs, and M.B. Cooper, "*Supply Chain Logistics Management*," NY, USA: McGraw-Hill Education, 2012.

[8] CSCMP, *CSCMP's Definition of Logistics Management*, Council of Supply Chain Management Professionals. Accessed May 25, 2014. http://www.cscmp.org/about-us/supply-chain-management-definitions

[9] M. Rouse, *Supply Chain Visibility*, SearchManufacturingERP, 2009. http://searchmanufacturingerp.techtarget.com

[10] H. Jung, K. Lee, and W. Chun, "Integration of GIS, GPS, and Optimization Technologies for the Effective Control of Parcel Delivery Service," *Comput. Ind. Eng.*, vol. 51, no. 1, Sept. 2006, pp. 154–162.

[11] J.-H. Park and J.-H. Park, "Postal RFID Application Model and Performance," *ETRI J.*, vol. 28, no. 3, June 2006, pp. 405–408.

[12] ESRI, *GIS for the Insurance Claims Process: Five Steps for an Effective Workflow*, White paper, California: ESRI, Apr. 2012.

[13] M. Rouse, *Bus. Process Manag. (BPM)*, SearchCIO. Accessed May 25, 2014. http://searchcio.techtarget.com

[14] V. Maliene et al., "Geographic Information System: Old Principles with New Capabilities," *Urban Des. Int.*, vol. 16, no. 1, 2011, pp. 1–6.

[15] SCC, *Supply Chain Operations Reference (SCOR) Model Overview-Version 10.0*, Supply Chain Council, 2010.

[16] H. Bae and Y. Kim, "A Document-Process Association Model for Workflow Management," *Comput. Ind.*, vol. 47, no. 2, Feb. 2002, pp. 139–154.

[17] P.A. Zandbergen, "A Comparison of Address Point, Parcel, and Street Geocoding Techniques," *Comput., Environment, Urban Syst.*, vol. 32, no. 3, May 2008, pp. 214–232.

[18] OSGeo, *PostGIS 1.5.3 Manual*, Open Source Geospatial Foundation. http://www.osgeo.org

[19] C.W. Tan and A. Goh, "Implementing ECA Rules in an Active Database," *Knowl.-Based Syst.*, vol. 12, no. 4, Aug. 1999, pp. 137–144.

[20] A. Goh, Y.-K. Koh, and D.S. Domazet, "ECA Rule–Based Support for Workflows," *Artif. Intell. Eng.*, vol. 15, no. 1, Jan. 2001, pp. 37–46.

[21] M. Rusinkiewicz and A. Sheth, "Specification and Execution of Transactional Workflows," in *Modern Database Systems*, NY, USA: Addison-Wesley Publishing Co., 1995, pp. 592–620.

[22] H. Bae and Y.-W. Yu, "Auto-Triggering of RFID-Based Logistic Process in Inter-Workflow Using Complex Event," in *Dynamics in Logistics*, Berlin, Germany: Springer Berlin-Heidelberg, Jan. 2011, pp. 329–339.

[23] J.-Y. Lee, "Integrating Spatial and Temporal Relationship Operators into SQL3 for Historical Data Management," *ETRI J.*, vol. 24, no. 3, June 2002, pp. 226–238.

[24] J. Gubbia et al., "Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions," *Future Generation Comput. Syst.*, vol. 29, no. 7, Sept. 2013, pp. 1645–1660.

Yeong-Woong Yu received his BBA and MS degrees in e-Business from Dong-Eui University and Pusan National University, Busan, Rep. of Korea, in 2005 and 2007, respectively. Since 2007, he has been working toward his PhD degree in industrial engineering at Pusan National University. Since 2011, he has been a researcher at the Electronics and Telecommunications Research Institute, Daejeon, Rep. of Korea. His research interests include BPM; process mining and modeling; and GIS-based process execution and monitoring.

Hoon Jung received his BS degree in industrial engineering from Kyunghee University, Seoul, Rep. of Korea, in 1989 and his MS and PhD degrees in industrial engineering from Iowa State University, USA and the University of Missouri, MO, USA, in 1997 and 2001, respectively. Since 2002, he has been working for the Electronics and Telecommunications Research Institute, Daejeon, Rep. of Korea, where he is now a principle researcher. His main research interests include supply chain management, logistics information systems, and logistics strategy.

Hyerim Bae received his BS, MS, and PhD degrees in industrial engineering from Seoul National University, Rep. of Korea. From 2002 to 2003, he worked for Samsung Credit Card Corp., Seoul, Rep. of Korea. Since 2005, he has been a professor with the Department of Industrial Engineering, Pusan National University, Rep. of Korea. He is interested in information system design; cloud computing; business process management systems; and process mining and big data analytics for operational intelligence. His current research activities include analyzing huge volumes of event logs from port logistics and shipbuilding industries using process mining techniques.