IEMS Vol. 9, No. 3, pp. 285-293, September 2010.

Performance Modeling of an EPC Information Service System

Sojung Kim
Department of Industrial and Systems Engineering
Dongguk Univ-Seoul, 26, Pil-dong 3-ga, Jung-gu, Seoul 100715, South Korea
Tel: +82-2-2260-3809, E-mail: sojungkim@dgu.edu

Yongshin Kang
Department of Industrial and Systems Engineering
Dongguk Univ-Seoul, 26, Pil-dong 3-ga, Jung-gu, Seoul 100715, South Korea
Tel: +82-2-2260-3809, E-mail: yskang@dgu.edu

Kyungwon Son
Department of Industrial and Systems Engineering
Dongguk Univ-Seoul, 26, Pil-dong 3-ga, Jung-gu, Seoul 100715, South Korea
Tel: +82-2-2260-3809, E-mail: kwson@dgu.edu

Yong-Han Lee†
Department of Industrial and Systems Engineering
Dongguk Univ-Seoul, 26, Pil-dong 3-ga, Jung-gu, Seoul 100715, South Korea
Tel: +82-2-2260-3809, E-mail: yonghan@dgu.edu

Jongtae Rhee
Department of Industrial and Systems Engineering
Dongguk Univ-Seoul, 26, Pil-dong 3-ga, Jung-gu, Seoul 100715, South Korea
Tel: +82-2-2260-3809, E-mail: jtrhee@dgu.edu

Sung Jo Hong
Department of Industrial and Systems Engineering
Dongguk Univ-Seoul, 26, Pil-dong 3-ga, Jung-gu, Seoul 100715, South Korea
Tel: +82-2-2260-3809, E-mail: sjhong@dgu.edu

Received, March 31, 2010; Revised, June 26, 2010; Accepted, July 25, 2010

Abstract. To obtain visible and traceable information from the supply chain, HW/SW standards for the EPC global network, which process electronic product code (EPC) data read from Radio frequency identification (RFID) tags, are regarded as the de facto industry standard. Supply chain participants install information service systems and provide logistics information to partners by following the EPCglobal architecture framework. Although quality of service (QoS) is essential for providing dependable and scalable services as pointed out by Auto-ID Lab, only a few models for the performance analysis of QoS-related work have been developed in the context of EPC information service systems. Specifically, doing so allows alternative design choices to be tested in an easy and cost-effective manner and can highlight potential performance problems in designs long before any construction costs are incurred. Thus, in this study we construct a model of an EPC information service system for the purposes of performance analysis and designing a dependable system. We also develop a set of building blocks for analytical performance models. To illustrate how the model works, we determine the characteristics of an EPC information service system and then select a combination of these proven modeling concepts. We construct a performance model that considers the response time and shows how to derive meaningful performance values. Finally, we compare the analytical results to measurements of the EPC information service system.

Keywords: EPCIS, Performance Models, RFID, EPCglobal Network, QoS, Queuing Theory

1. INTRODUCTION

Radio frequency identification (RFID) technology is considered to be a core component in a new generation of technologies applied in various fields such as information technology, logistics, distribution, supply

† : Corresponding Author

chains, transport, and the environment. For visible and traceable information in the supply chain, HW/SW standards for the EPCglobal network, which processes electronic product code (EPC) data read from RFID tags, are regarded as the de facto standard in the industry. EPC is a group of coding schemes that has been created as an eventual successor to the barcode and it uses RFID technology.

The EPCglobal network is a complete, complex, and scalable network of EPC subscribers. The EPC subscribers publish EPC information about their business and are interested in EPC data. The manufacturer assembles products and attaches RFID tags with a unique EPC for each product. Through the EPCglobal network, the manufacturer can track the product in their supply chain, which is comprised of other subscribers such as distributors and resellers (Assiotis *et al.*, 2006).

Therefore, RFID data related to logistics and the supply chain are stored by specific data formations defined by the EPCglobal architecture framework standards of EPC information service (EPCIS), Object Naming Service (ONS), and discovery service (DS). Thus, it is important to understand and utilize the EPCglobal standard.

Although the number of EPCglobal subscribers is increasing, Auto-ID Lab has stated that the EPC technology adoption level is still in its infancy compared to barcode technology. Thus, to realize a global RFID data exchange infrastructure, problems such as scalability, overlapping of various standards, master data management, economic evaluation of data sharing, data ownership and cost distribution, and security and privacy of information have to be solved (Thiesse *et al.*, 2009).

To solve the scalability problem, we developed an EPCIS performance model using analytical methods. EPCIS plays a key role as a gateway to the EPCglobal architecture framework because it accepts EPC information through an interface and distributes the same to other subscribers. EPCIS determines the visibility and traceability of information in a supply chain as it manages EPC information (Assiotis *et al.*, 2006).

The analytical performance model is widely used because it can easily forecast when a system is broken and prevents loss of data and the associated cost. In several cases, simulation methods have been used for performance modeling, but these simulations are expensive in terms of both programming and computational time. Furthermore, parameter space is not comprehensively covered, and sensitivity analyses are not as thorough as desired (Nicola *et al.*, 2000). For example, Mukkamala (1989) shows that even for simple evaluations of distributed databases, the simulation time may be 1024 times that of analytical evaluation, especially in the early stages of design. Born (1996) reported his experience that the design, implementation, and quality assurance of a reliable simulation model for a distributed system costs at least an order of magnitude more than the corresponding analytical model.

In this study, we surveyed a database system performance model that is similar to the EPCIS system and present the EPCIS characteristics in section 2. Using results from related work, we developed an EPCIS performance model and calculated the response time, which is a performance measurement parameter, as described in section 3 and 4. Section 5 presents our evaluation of the performance results and the validation of the performance model through a comparison with the results from an EPCIS benchmark test. Section 6 presents the conclusions.

2. RELATED WORK

Before the EPCIS performance model design, in this section we present a survey of EPCIS and related performance models. First, we found EPCIS characteristics and event types that are determined by the supply chain process. Second, we studied a performance model of the database system because it has similarities to the EPCIS repository, which is an EPCIS component that stores EPC data and provides EPC information. Through performance model analysis, we then designed a performance model for EPCIS.

2.1 EPC Information Service System

EPCIS is a gateway for the EPCglobal architecture framework called the EPCglobal network. The EPCglobal network has a set of standards to enable data sharing of EPC-related information within and between enterprises. Figure 1 shows the roles and interfaces in the EPCglobal network. First, an RFID reader reads and parses tags according to the EPC tag data specifications and tag protocols; through a reader protocol, the reader then provides data into RFID middleware for filtering and collection. Internal applications can then capture application layer events (ALE) for product information through an ALE interface. These events and related business information are then delivered by the EPCIS capture interface to Enterprise Repository. The information receives RFID tag information from middleware such as ALE to generate state or track information for an object, e.g., product, box, pallet, etc. The state or track information is saved to a local repository and managed by EPCIS. Thus, EPCIS serves like a hub of a system and is an important element of the EPCglobal architecture framework (EPCglobal, 2007). EPCIS also provides an information sharing service of object information between partners for visibility and traceability.

2.1.1 EPCIS Characteristics

EPCIS manages object information, which can be separated into two categories: static and dynamic. Static information is the nature character data of physical objects. It consists of class-level information that does not change (e.g., product name, code, manufacture, etc.) and instance-level information (e.g., manufacturing data, period of circulation, etc). Thus, it provides homogeneity and is unique to a physical object.

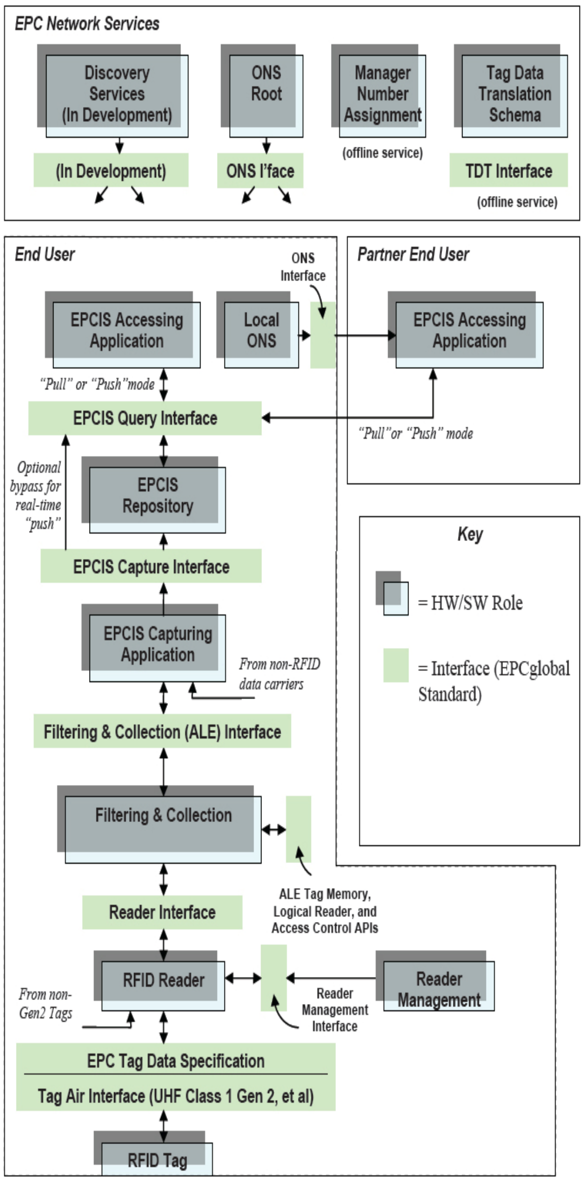
EPC Network Services
Discovery Services (In Development)
ONS Root
Manager Number Assignment
Tag Data Translation Schema
(In Development)
ONS I'face
(offline service)
TDT Interface
(offline service)
End User
ONS Interface
EPCIS Accessing Application
Local ONS
Partner End User
EPCIS Accessing Application
"Pull" or "Push" mode
EPCIS Query Interface
"Pull" or "Push" mode
Optional bypass for real-time "push"
EPCIS Repository
EPCIS Capture Interface
EPCIS Capturing Application
From non-RFID data carriers
Filtering & Collection (ALE) Interface
Filtering & Collection
ALE Tag Memory, Logical Reader, and Access Control APIs
Reader Interface
RFID Reader
Reader Management
Reader Management Interface
From non-Gen2 Tags
EPC Tag Data Specification
Tag Air Interface (UHF Class 1 Gen 2, et al)
RFID Tag
Key
= HW/SW Role
= Interface (EPCglobal Standard)

Figure 1. EPCglobal Network.

Dynamic information changes depending on physical object movement and state variation (e.g., supply chain location information of product, inside or outside the warehouse, sales information, etc.). EPCIS saves events generated by ALE according to business steps such as observations, parent and child relationships, quantities, transactions, etc.

EPCIS consists of an EPCIS capture interface, EPCIS repository, and EPCIS query interfaces. Following a business event, the EPCIS capture application changes the data on the EPC level through the filtering and collection interface.

The EPCIS capture interface saves data to the EPCIS repository in real-time when it receives EPCIS event data from the EPCIS capture application. The EPCIS repository provides a permanent storage space for saving EPCIS event data and EPC data to reply to queries sent from the EPCIS accessing application. The EPCIS query interface provides a real-time response to continuous queries from the EPCIS accessing application or a partner's accessing application. The EPCIS query interface provides two services. First, the "on-demand" and "synchronous" modes are provided by an EPCIS query control interface to immediately respond to a query result. Second, clients can reserve a response time period and receive data through the EPCIS query callback interface, which has "standing request" and "asynchronous" modes (EPCglobal, 2007).

2.1.2 EPC Events

EPC was created as a low-cost method for tracking goods using RFID technology. Barcodes only classify to the product level, but EPC can distinguish individual items with the same product level. The EPC structure consists of a header, company prefix, item reference, and serial number. Because EPC has a serial number for an individual item, we can track and trace each EPC in real-time (EPCglobal, 2007).

EPCIS records EPC data using four event types that are determined by the business step in the supply chain. First, an object event simply represents an event that happens to an EPC-tagged entity. It encapsulates information about the event's time, location, entities involved, etc. A detailed explanation of the fields follows below:

- *eventTime*: time the event occurred.
- *recordTime*: optional field; encapsulates the time the event is recorded into the system.
- *epcList*: list of EPC codes for all EPC-tagged entities described in the event.
- *action*: describes what happened to the entities in question. For example, the observe action can be used to describe the observation of some objects at a certain location.
- *bizStep*: *BusinessStepID* is used for business purposes to track the business step an event was part of.
- *disposition*: *DispositionID* is another field used to build business logic dependencies.
- *read point*: point at which an event took place; this information is supplied by the physical reader.
- *bizLocation*: business location where the event occurred. This differs from the read point as it is not associated with the exact physical location; instead, it refers to a business location.

Second, an aggregation event contains relationships between groups of tagged objects contained in another one. For example, an aggregation event can be "at time *t*, these items were added to container *c*." A notable difference is that the *epcList* field is now *childEPCList* and the addition of *parentEPCID*. Third, a quantity event contains an inventory report about the number of instances a specific object class is counted in a certain business context, but it does not collect data on the EPCs of the entities themselves. Fourth, a transaction Event associates or dissociates tagged objects in business transactions. All events are temporal with two time fields: event time, when events occurred; and record time, when they are recorded in repositories (Assiotis *et al.*, 2006).

Although there are just four primary event types, each event type has different content because each process has various roles and specific process times in the supply chain. Thus, various transactions can be generated in EPCIS.

2.2 Performance Models for the Database System

To evaluate or simulate database system performance, many studies have commonly used queuing theory and Kendall's classifying notation for queuing systems. Most database systems are open queuing networks, which means that arrival transactions are not limited (Nicola *et al.*, 2000). Some performance models assume a closed queuing network for easy analysis (Carey *et al.*, 1995; Liang *et al.*, 1996). However, it is unreasonable that there is no limitation to the transaction generation quantity in the real world.

Early performance models using queuing theory are defined as M/M/m/FCFS systems (Bacelli *et al.*, 1983; Coffmann *et al.*, 1981). M/M/m/FCFS denotes a Poisson arrival process, exponential distribution service time, *m* servers, and first-come-first-serve. The read transactions are processed by *m* servers in parallel, while writing transactions occupy all *m* severs during their service time. However, these models only consider reading transactions even though the database system also processes writing transactions.

Many performance models use the M/M/1 model, which has an exponential distribution service time for transactions to evaluate performance, but it is also unrealistic because not all transactions have the same size and type (Nicola *et al.*, 2000). Banerjee *et al.* (1994) and Hwang *et al.* (1996) used the M/G/1/FCFS and M/G/1/RR systems. These models assume that service time follows a general distribution such as normal distribution. Because they have the same problem as the M/M/1 model assumption, Gallersdörfer *et al.* (1995) and Leung *et al.* (1997) used an M/H_2/1 model, which reflects two transaction types such as reading and writing information. Nicola *et al.* (2000) applied the M/H_r/1 model to consider various transaction types with different service times. However, those models are limited in their applicability to EPCIS performance model design because EPCIS has its own characteristics, such as various transaction arrivals and multitasking by several sessions; the M/G/1 model performs better than the M/G/c model if the M/G/1 model service rate is *c* and the M/G/c model service rate is 1. Therefore, it has a higher probability of failure in modeling the EPCIS performance.

Generally, database system performance models assume that the service time follows an exponential distribution because the service time is largely affected by the disk and storage data quantity (Son *et al.*, 1990). Another reason is that transactions mostly take a short amount of time, and few transactions have long service times. The service time follows a geometric distribution and can be approximated as an exponential distribution in a continuous environment. Thus, we can assume that the service time follows exponential distribution (Nicola *et al.*, 2000). Because EPCIS also has these characteristics, we used this assumption to design the performance model.

3. MODEL DESCRIPTION

In this study, we designed the EPCIS performance model to use queuing theory; the model is composed of an EPCIS capture interface, EPCIS repository, and EPCIS query interfaces. Before modeling, we assumed that each message follows a Poisson arrival process and takes an exponential distribution service time. A session is regarded as a server because it independently processes messages at the same time through multitasking. When comparing buffer size, the message size is relatively small, and our performance model only considers the steady state of a system. Thus, the EPCIS system uses an M/G/c queuing system as in Figure 2.

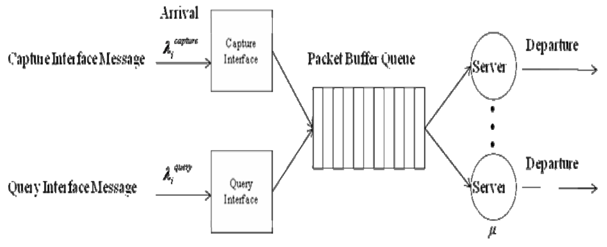

Figure 2. EPCIS system model.

There are four types of capture events, and the message size and resource requirements are determined by the supply chain process. For example, an aggregation event has different message sizes because the first aggregation, or an item in a box, and second aggregation, or a box in a pallet, have different capacities and EPC lists. They generate differently sized messages, so we regard each EPC tag read point as having its own message type.

Another interface is the query interface, which receives messages from other subscribers. Each message also differs in terms of resource requirements and message content. Let *r* be a number of message types; if message *i* has an arrival rate λ_i, then the total arrival rate λ is expressed as in formula (1).

$$\lambda = \sum_{i=1}^{r} \lambda_i \tag{1}$$

Because messages have different service times, the service time of message i is expressed by a hyper-exponential distribution with a weighted probability. The weighted probability is the arrival rate of message i divided by the total message arrival rate λ.

$$f(t) = \frac{1}{\lambda} \sum_{i=1}^{r} \lambda_i \cdot \mu_i \cdot e^{-\mu_i t} \tag{2}$$

Thus, we can express the EPCIS system as M/H$_r$/c, which is an M/G/c model. H$_r$ denotes r message types following a hyper-exponential service time.

4. PERFORMANCE MEASURES

In this section, we present two aspect sequence rules of EPCIS. First is first-come-first-serve (FCFS) in the EPCIS buffer, which is the current reference model. This model is the priority rule between the capture and query messages.

4.1 First-Come-First-Serve EPCIS

We considered the average response time as a performance measurement. This is the summation of the message transmission time and wait time in the EPCIS buffer. The response time is defined as:

$$R_i = W_q + S_i + t_i^{send} + t_i^{return} \tag{3}$$

This is a summation of the average waiting time W_q, service time S_i, and transmission times t_i^{send} and t_i^{return}. The average response time is shown below:

$$\bar{R} = \frac{\sum_{i=1}^{r} R_i}{r} \tag{4}$$

To calculate the response time, the average waiting time in the EPCIS system should be obtained. In the M/H$_r$/c queuing system, all messages have the same average waiting time. This is because each message waits in a shared buffer until it gets a service opportunity. Because the service time is a hyper-exponential distribution, one and two-moment service times are shown below:

$$E[S] = \int_0^{\infty} t f(t)\,dt = \frac{1}{\lambda} \sum_{i=1}^{r} \frac{\lambda_i}{c\mu_i} \tag{5}$$

$$E[S^2] = \int_0^{\infty} t^2 f(t)\,dt = \frac{2}{\lambda} \sum_{i=1}^{r} \frac{\lambda_i}{(c\mu_i)^2} \tag{6}$$

where μ_i is the message service time and λ_i is the arrival rate of type i. According to Tijms (1994), the M/G/c model has the same average waiting time as the M/G/1 model, which has a $c\mu$ service rate. Thus, we can use the average waiting time of the M/G/1 model, which is computed by the Pollaczek-Kinchine mean waiting time formula.

$$W_q = \frac{\lambda E(S^2)}{2(1-\rho)} = \frac{\sum_{i=1}^{r} \frac{\lambda_i}{(c\mu_i)^2}}{\left(1 - \sum_{i=1}^{r} \frac{\lambda_i}{c\mu_i}\right)} \tag{7}$$

The message service time is determined by the message size or load. Specifically, the capture message has a different message size following business processes and is generated at the RFID tag read point because all messages have various EPC codes. An object event generally has one EPC code compose a message. But for aggregation events, messages are composed by many EPC codes because one box has several items. Thus, the message service time is shown below:

$$\mu_i^{capture} = \frac{\mu}{c \cdot size_i^{capture}} \tag{8}$$

where c is the session number and $size_i^{capture}$ is the message size. The weight mean correction coefficient modifies differences in message size, service rate, etc. EPCIS message transmission uses simple object access protocol (SOAP) messaging. Thus, the message size becomes larger than the original data. We also assumed that the encoding and decoding times include the service time because it is short compared with the data search time. However, the query message weight should consider not only the size but also the data search time, which is affected by storage devices such as a hard disk and indexing strategies on the EPCIS repository. Thus, we use the search time weight d to reflect the data quantity in a hard disk.

$$\mu_i^{query} = \frac{\mu}{c \cdot d} \tag{9}$$

If the product process follows a Poisson process, the capture message arrival rate can be expressed by the produce rate as shown below:

$$\lambda_i^{capture} = w_i^{arrival} \lambda_p \tag{10}$$

where $w_i^{arrival}$ is the weight of the produce rate λ_p. For example, if four items are packed in one box, then the item's weight is 1 and the box weight is 0.25. However, the query message has an arbitrary value because it does not have a clear relationship with the production rate.

The transmission time is another element of the response time that is determined by the message size and network transmission speed (Nicola *et al.*, 2000). There are two categories: send and return. Capture message is an unnecessary response message, but query message considers two aspects because send and return have different sizes and requirements. The message send and return are shown below:

$$t_i^{return} = \frac{8 \cdot w_i^{transmit} \cdot size_i^{return}}{bps} \tag{11}$$

$$t_i^{send} = \frac{8 \cdot w_i^{transmit} \cdot size_i^{send}}{bps} \tag{12}$$

4.2 Priority EPCIS

Although current EPCIS specifications do not deal with message priority, we must consider this because messages have different values. For example, if a capture message was rejected by a query message when the capture and query messages enter the EPCIS at the same time, this can cause a second problem because EPCIS may then provide the query massage with wrong information for the location, quantity, etc. Thus, we assume that a capture message has higher priority than a query message.

As discussed in Section 4.1, the average response time and relative formulas are same. We only considered the average waiting time following the message priority. Let K be the total number of priority layers; for EPCIS, K is two: capture and query messages. In layer k, the average load is $\rho_k = \lambda_k E(S_k)$ in unit time, and the busy probability ρ at an arbitrary point of view is the summation of ρ_i. Assuming that $T_{a(1)}$ is the average time of layer 1, the average waiting time is shown below:

$$\begin{aligned} W_{q(1)} &= E\left(T_{q(1)}\right) = \rho E\left(T_{q(1)} \mid busy\right) \\ &= \frac{\sum_{k=1}^{2} \lambda_k E\left[S_k^2\right]}{2\left(1-\rho_1\right)} \end{aligned} \tag{13}$$

The idle probability is zero because a waiting situation does not occur when the server is idle. $E(T_{a(2)}|busy)$ has a total expected service time of existing messages between layers 1 and 2 in queue of $T_{s(1,2)}$, expected residual service time of messages in server of S_R, and expected service time of new arrival message in higher layers of S_A. An arbitrary layer's probability of service when a server is busy is ρ_i/ρ. $E(T_{a(2)}|busy)$ is shown below:

$$\begin{aligned} E\left(T_{q(2)} \mid busy\right) &= E\left(S_R\right) + E\left(T_{S(1,2)}\right) + E\left(S_A\right) \\ &= \frac{1}{\rho}\sum_{k=1}^{2} \frac{\lambda_k E\left[S_k^2\right]}{2} + \sum_{k=1}^{2} \frac{\rho_k W_{q(k)}}{\rho} + \frac{\beta \rho_1}{1-\rho_1} \\ &= \frac{\beta}{1-\rho_1} \end{aligned} \tag{14}$$

where β is as shown below:

$$\beta = \frac{1}{\rho}\sum_{k=1}^{2} \frac{\lambda_k E\left(S_k^2\right)}{2} + \sum_{k=1}^{2} \frac{\rho_k W_{q(k)}}{\rho} \tag{15}$$

We can express the average waiting time of layer k because each layer's waiting time increases regularly as shown below:

$$\begin{aligned} W_{q(2)} &= \rho E\left[T_{q(2)} \mid busy\right] = \rho \frac{\beta}{\left(1-\rho_1\right)} \\ &= \frac{\sum_{k=1}^{2} \lambda_k E\left[S_k^2\right]}{2\left(1-\rho_1\right)\left(1-\rho_1-\rho_2\right)} \end{aligned} \tag{16}$$

5. PERFORMANCE RESULTS

In this section, we compare the response time of the performance model and the simulation result. Before setting the parameters, we define the experiment scenario reflecting supply chain environment. Following the scenario, the parameter setting, EPCIS simulation, and response time calculation were conducted to verify our EPCIS performance model.

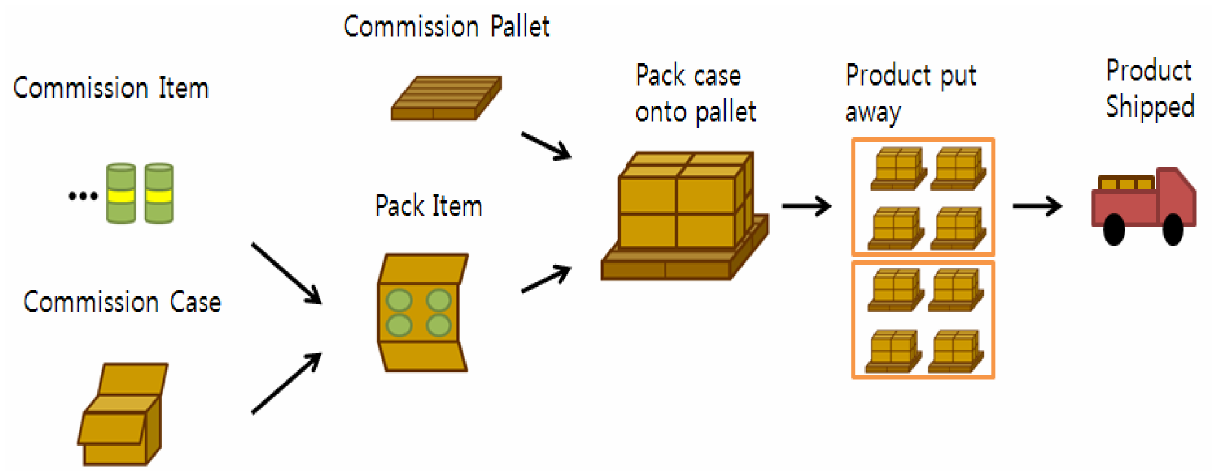

Figure 3. Manufacturing process of HLS.

5.1 Scenario and Parameter Settings

EPCIS is largely affected by the supply chain environment. We assumed that the supply chain follows the HLS Track and Trace Straw model, which consists of commission item, commission case, pack item, commission pallet, pack case onto pallet, product put away, and product shipped components (EPCglobal, 2007). There are seven EPC tag read points, boxes with a capacity for four items, and pallets with a capacity for eight boxes. In addition, a query message uses bills of material (BOM) query to confirm the content of a pallet, and capture messages for object and aggregation events are generated by each business step. For example, packing processes generate aggregation events that include the packing case and boxed items, and commission processes generate object events that register EPC codes to EPCIS. Figure 3 shows the supply chain process.

Table 1 shows the parameter values in our experiment. Each read point generates different message content and size. By measuring each message size, we determined the parameter values. We considered one BOM Query message with an index number of eight ($i = 8$). r is the number of business steps. We controlled the simultaneous process session number to 20. To determine d, we conducted several benchmark tests by changing the number of EPC events in the EPCIS repository. When the EPCIS repository stores more than 100 000 messages, d was 2.45 in our benchmark test environment. $w_i^{arrival}$ was determined by the packing number for each package in our scenario. Because SOAP messaging transforms ASCII into XML, $w_{1,2,3,4,5,6,7,8}^{transmit} = 4$. After fixing the parameter values, we calculated μ by comparison with the benchmark test results.

Table 1. parameter.

Parameter	Base setting	Parameter	Base setting
r	8	$size_{1,2,4}^{capture}$	0.79
λ_8^{query}	1	$size_3^{capture}$	1.01
μ	21.0	$size_5^{capture}$	1.18
c	20	$size_{6,7}^{capture}$	1.15
d	2.45	$size_8^{send}$	0.54
$w_1^{arrival}$	1	$size_8^{return}$	0.71
$w_{2,3}^{arrival}$	0.25	$w_{4,5,6,7}^{arrival}$	0.03125
$w_{1,2,3,4,5,6,7,8}^{transmit}$	4		

The benchmark test used one EPCIS operation server (ProLiant DL 360 G5 Intel Xeon E5430 2.66 GHz) with a buffer size of 8 MB and eight client computers (Inter Core2 Duo CPU E7400 2.66GHz). The network was connected at 10 Mbps WAN.

5.2 Response time

Each second, a BOM query message is generated; we controlled the production rate at 0-45 per second. Like a database system, the EPCIS response time had a radical change when the production rate was 35 per second, and our queuing network model showed a similar simulation result. The reason for the radical change in response time is that the production rate affects the EPC message generation rate twice. Seven read points each generate messages, and they are determined by a production rate according to a production schedule. If the transmission distance is very short such as formula (5) in Section 4, the response time is largely affected by the service time. The service time is determined by the message size, generation rate, and service rate. Figure 4 shows the benchmark test result for the average response time. The arrival rate for two messages has a positive relation with the average response time. The effect of the query message is higher than that of the capture message because the query message has a relatively larger size and longer search time.

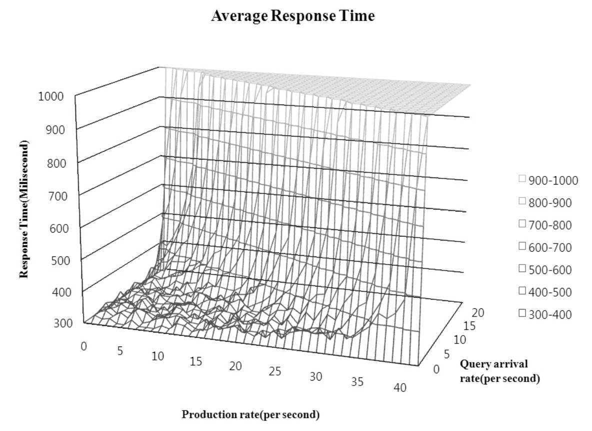

Figure 4. Benchmark test result.

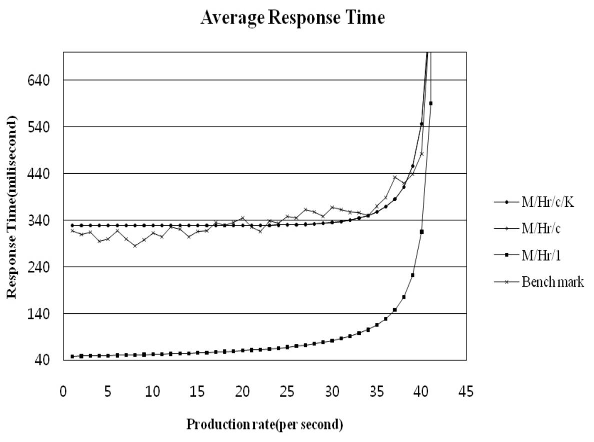

Figure 5. Comparing performance models.

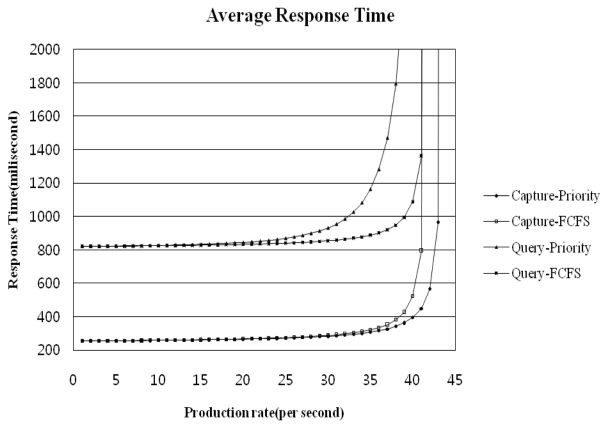

Figure 6. Comparing priority models.

Figure 5 show the goodness of fit test results for our model. We performed a paired-t test between our performance model and the benchmark test results. There was little difference, but the paired t-test results show that there was no difference in the mean (p-value = 0.64839). Following queuing theory, one server can service faster than a distributed server when the total service rate is the same. Thus, the M/hr/1 model response time is much shorter than the benchmark result. A proper analytical model was not used to measure the EPCIS performance. And also M/G/c/K model which was proposed by Nozaki-Ross (1978) presented similar results as our study because *K* is relatively large over each message size.

Finally, we compared the priority EPCIS model with the FCFS EPCIS model, and the result is shown in Figure 6. Because the capture message has a higher priority than the query message, the former reduces the average response time and increases the critical value for utilization to 1. However, each query message increases the average response time and reduces the critical value. These are natural results in that the priority rule raises the performance of high-priority messages.

6. CONCLUSION

EPCIS has a key role as a gateway in the EPCglobal architecture framework; it determines the visibility and traceability of information in a supply chain as it manages EPC information. However, too many users and EPC data can cause EPCIS to fail and produce a scalability problem. To solve the scalability problem, we developed an EPCIS performance model using analytical methods.

Through EPCIS characteristic analysis, we found that the production rate and business steps of subscribers are important to EPC event generation because these influence the capture message generation rate. Specifically, business steps determine the EPC message size or load. EPCIS also provide a query message service that is largely affected by the EPC data quantity in the EPCIS repository. Thus, we designed an EPCIS performance model that considers the characteristics of two interfaces and verifies them.

Our performance model properly implements the EPCglobal architecture framework with a reliable forecast of the system performance. Applying a priority rule to EPCIS can satisfy each subscriber's requirements through service level agreement (SLA) and reduce the capture message loss.

However, our performance model only considers EPCIS based on our developed system. Furthermore, we did not conduct various case studies. Thus, we should develop a performance model that considers all elements of the EPCglobal architecture framework for accurate forecasts applicable to various EPCIS systems.

ACKNOWLEDGMENT

This study was partially supported by UPLUS and SCCM projects, both sponsored by MKE (Ministry of Knowledge Economy) in Korea.

REFERENCES

Assiotis, M. and Mavrommatis, P. (2006), The EPC Global Network: A formal specification of EPC, *http://mavrommatis.googlepages.com/report.pdf.*

Bacelli, F. and Coffmann, E. G. (1983), A Database Replication Analysis Using an M/M/m Queue with Service Interruptions, *Performance Evaluation Review*, **11**, 102-107.

Banerjee, S., Li, V. O. K., and Wang, C. (1994), Performance Analysis of the Send-on-Demand: A Distributed Database Concurrency Control Protocol for High-Speed Networks, *Computer Comm.*, **17**, 189-204.

Born, E. (1996), Analytical Performance Modelling of Lock Management in Distributed Systems, *Distributed Systems Eng.*, **3**, 68-76.

Carey, M. J. and Livny, M. (1995), *Conflict Detection Tradeoffs for Replicated Data, Performance of Concurrency Control Mechanisms in Centralized Database Systems*, Prentice Hall, Inc., Upper Saddle River, NJ.

Coffmann, E. G., Gelenbe, E., and Plateau, B. (1981), Optimization of the Number of Copies in a Distributed System, IEEE Trans. *Software Eng.*, **7**, 78-84.

EPCglobal (2007), Business Requirements and Process Flows, HLS Track and Trace Interest Group.

EPCglobal (2007), EPC Information Services (EPCIS) v. 1.0 specification. *http://www.epcglobalinc.org/standa-rds/epcis*.

Gallersdörfer, R. and Nicola, M. (1995), Improving Performance in Replicated Databases through Relaxed Coherency, *Proc. 21st Conf. Very Large Databases*, 445-456.

Hwang, S. Y., Lee, K. S., and Chin, Y. H. (1996), Data Replication in a Distributed System: A Performance Study, *Proc. Seventh Int'l Conf. Database and Expert Systems Applications*, **1134**, 708-717.

Leung, K. K. (1997), An Update Algorithm for Replicated Signaling Databases in Wireless and Advanced Intelligent Networks, *IEEE Trans. Computers*, **46**, 362-367.

Liang, D. and Tripathi, S. K. (1996), Performance Analysis of Long-Lived Transaction Processing Systems with Rollbacks and Aborts, *IEEE Trans. Knowledge and Data Eng.*, **8**, 802-815.

Mukkamala, R. (1989), Measuring the Effects of Data Distribution Models on Performance Evaluation of Distributed Database Systems, *IEEE Trans. Knowledge and Data Eng.*, **1**, 494-507.

Nicola, M. and Jarke, M. (2000), Performance Modeling of Distributed and Replicated Databases, *IEEE Transactions on Knowledge and Data Engineering*, **12**, 645-672.

Nozaki, S. and Ross, S. (1978), Approximations in Finite-Capacity Multi-server Queues with Poisson Arrivals, J. *Appl.Prob*, **15**, 826-834.

Son, S. H. and Haghighi, N. (1990), Performance Evaluation of Multiversion Database Systems, *Proc. Sixth Int'l Conf. Data Eng.*, 129-136.

Thiesse, F., Floerkemeier, C., Harrison, M. Michahelles, F., and Roduner, C. (2009), Technology, Standards, and Real-World Deployments of the EPC Network, *IEEE Internet Computing*, **13**, 36-43.