Boxcan: A Platform Realizing Fast Retrieval of Parent-Child Tree of Containers and Inner Objects Over EPCIS Events

Yuki Sato, Taisuke Sato, Jin Mitsugi
Auto-ID Laboratory Japan at Keio University
5322 Endo, Fujisawa, Kanagawa 252-0882 Japan
Email:{sat3, paina, mitsugi}@sfc.wide.ad.jp

***Abstract*—This paper introduces "Boxcan", an information retrieval platform for aggregated objects in ID-based object management system based on GS1 EPCglobal architecture framework. Boxcan platform receives an EPC of a container, then provides a tree structure of current parent-child relationship between the EPCs of the container and its inner objects. This paper proposes the system design of Boxcan platform and its applications. The proposed system design is evaluated with a field test of a practical object management system including Boxcan platform. A technical challenge to realize this function of Boxcan platform, fast retrieval of current parent-child relationship of EPCs from EPCIS, is also discussed in this paper. The effectiveness of the caching mechanism of EPC's current parent-child tree, which is a solution to this technical problem, is also evaluated by an experiment in a comparison with direct querying to EPCIS.**

***Index Terms*—Internet of Things, Supply chain management, Object management, GS1 EPCglobal architecture framework, EPCIS**

I. Introduction

ID-based object management is one of typical applications of Internet of Things. Objects are assigned their own unique identifiers (IDs), then information systems recognize these objects with using ID-based automatic identification technologies such as barcode and RFID. In addition to automatic identification technologies, recent enhancements of cloud computing and ubiquitous networks enable information systems to collect such real-space information in a real-time manner. Feedbacks by information systems to real-space based on such real-time real-space information can reduce losses in current object management processes, e.g. over-stocking and thefts by workers, thus contribute to sustainable society.

GS1 EPCglobal architecture framework [1] is a set of standards which focuses on such ID-based object management systems. In GS1 EPCglobal architecture framework, each objects are assigned an unique EPC, Electronic Product Code, as an ID. There are many prior works about object management system based on GS1 EPCglobal architecture framework, such as [2] and [3]. We also developed an object management system which interworks with an electronic commerce system [4], [5].

Not only individual management, objects are often put into and handled by a container. Identification of such aggregated objects is done by opening the containers and checking the all of inner objects individually, but it is a tough work when there are large amount of containers and objects. Even if we use RFID, which can recognize non-line-of-sight RF tags attached on objects in the container, it is difficult to guarantee that the all objects in the container are identified. In some ID-based object management systems, not only individual objects but also containers are managed by their own IDs. Retrieval of IDs of the objects in a container from container's ID is effective in such situations.

This paper introduces "Boxcan (box+scan)", which is developed as the part of the object management system based on GS1 EPCglobal architecture framework. Boxcan is an information retrieval platform which retrieves EPCs of objects from their container's EPC. Boxcan absorbs the difference between individual objects' EPCs and containers' EPCs, thus applications implemented on Boxcan platform do not need to be aware whether an EPC is assigned to an individual object or a container. A technical challenge to realize this function of Boxcan platform is fast retrieval of a tree structure of parent-child relationship of EPCs, which changes dynamically by adding/removing objects to/from a container. In GS1 EPCglobal architecture framework, such EPCs' parent-child relationship is expressed with AggregationEvents recorded in EPCIS [6], EPC Information Service. A caching mechanism which realizes fast retrieval of parent-child tree of EPCs based on AggregationEvents is also proposed in this paper.

This paper is organized as the following. Section II introduces the system design of Boxcan. The fast retrieval of the current parent-child tree of EPCs is discussed in Section III. Section IV evaluates the proposed system. Section V concludes this paper.

II. System design

This section introduces Boxcan platform. The first subsection briefly introduces EPCIS, which is the background of Boxcan platform. Then, the next subsection describes system design of Boxcan platform with one example of applications implemented on it.

A. Basis of EPCIS

EPCIS is a service which records information of objects attached their own EPCs as "EPCIS events". EPCIS 1.0.1

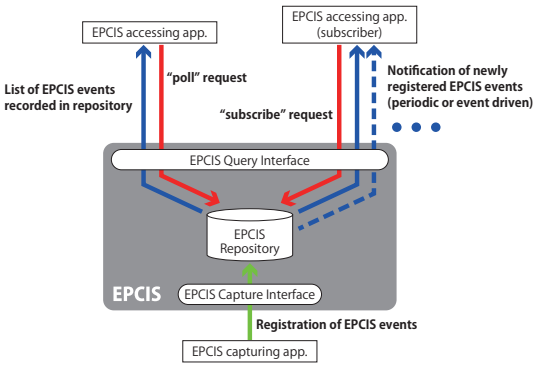

Fig. 1. Registration (capture) and retrieval (query) of EPCIS events

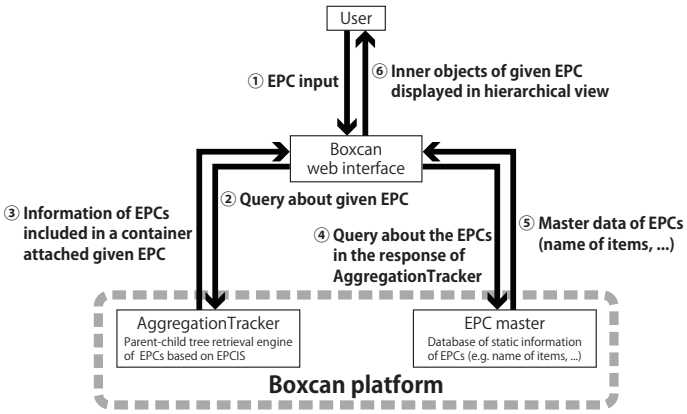

Fig. 2. Example transactions between Boxcan platform and its application (Boxcan web interface as an example)

specification [6] defines the following 4 types of EPCIS events.[1]

- An **ObjectEvent** records information of individual objects. The *epcList* field records a list of EPCs related to the event.
- An **AggregationEvent** records a parent-child relationship of objects, such as a relationship between a container and its inner objects. The values of the *parentID* field and the *childEPCs* field correspond to a parent and children of the relationship, respectively.
- A **QuantityEvent** records quantity information of objects. The *epcClass* field records EPC URI representing class of the object, and the *quantity* field records its quantity.
- A **TransactionEvent** records relationship information between objects and business transactions. The *parentID* field and the *epcList* field record EPCs related to the event.

The all of these event types have fields recording where / when / why an event has occurred.

EPCIS is composed of EPCIS repository, capture interface and query interface. EPCIS events registered by capturing applications through the capture interface are recorded by the EPCIS repository, and these events can be retrieved through the query interface. The query interface provides two event retrieval method, *poll* and *subscribe*. When the *poll* method is invoked, EPCIS responds with a list of recorded EPCIS events which match conditions given by an accessing application. The *subscribe* method registers subscription requests from accessing applications. After this subscription request via the *subscribe* method, EPCIS sends newly registered events to registered subscribers. Figure 1 shows the flow of registration (capture) and retrieval (query) of EPCIS events.

B. System design of Boxcan platform

This subsection explains the design of Boxcan platform with one example of its applications. This application, Boxcan web interface, displays a hierarchical list of objects contained in the container whose EPC is given by a user. It enables workers to grasp what objects are in a containers without opening it. Figure 2 shows an overview of transactions related to this application.

The EPC given to Boxcan web interface may be an individual object's one or a container's one containing some objects. Boxcan web interface sends this EPC to AggregationTracker, a current parent-child tree retrieval engine of EPCs based on EPCIS events. AggregationTracker returns parent-child tree whose root element is the given EPC. Quantity information of EPCs in this tree is also included in the response. These two kinds of information returned by AggregationTracker are based on AggregationEvents and QuantityEvents registered to EPCIS, respectively. The reason why we introduce AggregationTracker instead of direct querying to EPCIS is described in the next section.

AggregationTracker has the same query interface as EPCIS query control interface, i.e. AggregationTracker returns the above two kinds of information in the form of EPCIS events. A parent-child tree and EPCs' quantity information are expressed with AggregationEvents and QuantityEvents, respectively. Each AggregationEvent contains one container's EPC in the *parentID* field and its direct children's EPCs in the *childEPCs* field, and the whole of the tree is expressed with the same number of AggregationEvents as the number of containers. Each EPC's quantity information is expressed with a QuantityEvent whose *epcClass* field records that EPC. Note that these events in responses of AggregationTracker are different from EPCIS events recorded by the EPCIS repository. Figure 3 and Table I show an example of EPCs which compose parent-child relationship and AggregationTracker's responses in this example, respectively.

After the query to AggregationTracker, Boxcan web interface retrieves master data of the EPCs included in the response of AggregationTracker. Master data of EPCs means static properties of objects attached EPCs, such as name of objects, category etc. Boxcan web interface retrieves this information with accessing EPC master, a database recording the master data, then finally displays a list of the EPCs of inner objects

[1]The latest version of EPCIS standards is 1.1, but this description is based on the older version because we developed Boxcan platform for EPCIS 1.0.1. However, the idea of Boxcan platform can be applied on EPCIS 1.1.

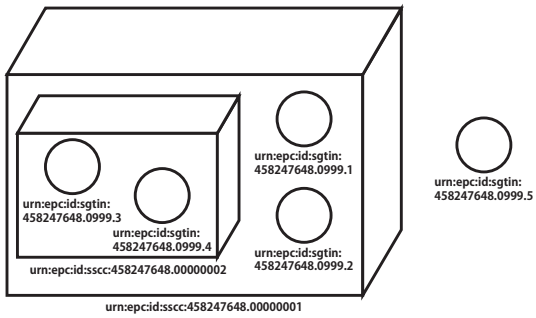

Fig. 3. Example of a parent-child relationship between EPCs (Circles and boxes represent individual objects and containers, respectively.)

TABLE I
AGGREGATIONTRACKER'S RESPONSES IN THE EXAMPLE OF FIG.3

Inputted EPC	Response (unordered list of EPCIS events)
urn:epc:id:sscc: 458247648.00000001	AggregationEvent - parentID=urn:epc:id:sscc:458247648.00000001 - childEPCs=urn:epc:id:sscc:458247648.00000002 urn:epc:id:sgtin:458247648.0999.1 urn:epc:id:sgtin:458247648.0999.2 AggregationEvent - parentID=urn:epc:id:sscc:458247648.00000002 - childEPCs=urn:epc:id:sgtin:458247648.0999.3 urn:epc:id:sgtin:458247648.0999.4 QuantityEvent - epcClass=urn:epc:id:sscc:458247648.00000001 QuantityEvent - epcClass=urn:epc:id:sgtin:458247648.0999.1 QuantityEvent - epcClass=urn:epc:id:sgtin:458247648.0999.2 QuantityEvent - epcClass=urn:epc:id:sscc:458247648.00000002 QuantityEvent - epcClass=urn:epc:id:sgtin:458247648.0999.3 QuantityEvent - epcClass=urn:epc:id:sgtin:458247648.0999.4
urn:epc:id:sscc: 458247648.00000002	AggregationEvent - parentID=urn:epc:id:sscc:458247648.00000002 - childEPCs=urn:epc:id:sgtin:458247648.0999.3 urn:epc:id:sgtin:458247648.0999.4 QuantityEvent - epcClass=urn:epc:id:sscc:458247648.00000002 QuantityEvent - epcClass=urn:epc:id:sgtin:458247648.0999.3 QuantityEvent - epcClass=urn:epc:id:sgtin:458247648.0999.4
urn:epc:id:sgtin: 458247648.0999.1	QuantityEvent - epcClass=urn:epc:id:sgtin:458247648.0999.1
urn:epc:id:sgtin: 458247648.0999.5	QuantityEvent - epcClass=urn:epc:id:sgtin:458247648.0999.5
urn:epc:id:sgtin: 458247648.0999.6	null list

contained in the container. When the container includes other containers, the multi-layered parent-child tree is displayed in a hierarchical view. The information associated with these EPCs, such as names and quantities, are also displayed.

The features of AggregationTracker and EPC master is required for not only this example application but also many other applications related to object management systems. Therefore, these subsystems are implemented as a platform which can be used by multiple applications, then we named this platform "Boxcan platform". In the object management system which we developed, not only the above-mentioned Boxcan web interface but the following applications are implemented on the Boxcan platform.

- Inventory management application
- Interworking function between non-EPC electronic commerce system and EPCIS

Boxcan platform enables these applications to manage individual objects in a container only with an input of the container's EPC.

III. FAST RETRIEVAL OF CURRENT PARENT-CHILD TREE OF EPCS

This section discusses on the retrieval of current parent-child tree of EPCs based on EPCIS event. At first, a problem of EPCIS related to this function is described, then a subsystem which solves this problem, i.e. detail of AggregationTracker, is introduced.

A. EPCIS's problem in retrieval of current status of objects

Each EPCIS event recorded in EPCIS repository represents an observation or an operation for objects attached EPCs. In other words, it is not necessary that each EPCIS event records these objects' complete current status at the time when the event is issued. As an example, EPCIS 1.0.1 specification [6] defines three types of meaning of AggregationEvent which are distinguished according to the value of its *action* field as the following.

- An AggregationEvent whose *action* field's value is "ADD" means addition of EPCs recorded in the *childEPCs* field to the parent-child relationship whose parent is recorded in the *parentID* field.
- An AggregationEvent whose *action* field's value is "OBSERVE" means observation of EPCs which compose a parent-child relationship. There is a possibility that the *parentID* and *childEPCs* fields do not record the all of EPCs belonging to this relationship.
- An AggregationEvent whose *action* field's value is "DELETE" means removal of EPCs recorded in the *childEPCs* field from the parent-child relationship whose parent is recorded in the *parentID* field.

These definitions are reasonable from the viewpoint of EPCIS capturing applications, which generate EPCIS events. It is impractical to check the all objects in a container and recognize these EPCs at each time of addition/removal of objects to/from the container, which are expressed as AggregationEvents whose *action* fields record "ADD" and "DELETE", respectively. As mentioned in the introduction, complete observation of EPCs belonging to an parent-child relationship is also a tough work.

Conversely, from the viewpoint of Boxcan, the above-mentioned meanings of AggregationEvent are troublesome

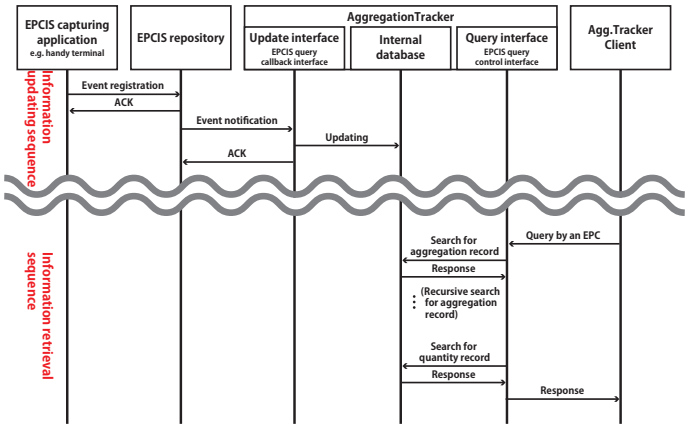

Fig. 4. Sequence of information updating and retrieval of AggregationTracker (information updating and retrieval processes are asynchronous each other)

because the all of them do not represent current and complete parent-child relationship of EPCs, i.e. a complete list of objects in a container. To retrieve this information by direct querying to EPCIS, it is required to process the all of AggregationEvents whose *parentID* fields record the container's EPC. If EPCs compose multi-layer parent-child relationships, e,g, objects in inner cases in a container, recursive repetition of the above process is required to retrieve complete parent-child relationship information because each AggregationEvent records information of only single-layer parent-child relationship. It is considered that this process causes long delay when the amount of events recorded in EPCIS repository becomes large. In addition, when there are multiple queries on the same EPC, it is required to repeat this process which is mostly (if there is no newly registered event, totally) same to that of the previous query.

B. AggregationTracker: improvement on retrieval delay by caching current parent-child tree

AggregationTracker whose function is introduced in Section II is developed to solve the problem mentioned in the previous subsection. A concept of AggregationTracker is caching current parent-child tree of EPCs. When new events are registered to EPCIS, AggregationTracker updates its cache by processing these events. AggregationTracker responds to queries based on the cache, thus realizes fast response in comparison with direct querying to EPCIS. Figure 4 is a sequence diagram of AggregationTracker.

AggregationTracker is a subscriber of EPCIS, which implements EPCIS query callback interface to receive notifications of newly registered events from EPCIS. EPCIS query callback interface can receive notifications from multiple EPCIS repositories, thus one AggregationTracker can manage information recorded in multiple EPCIS repositories. This extensibility is also an advantage of AggregationTracker over direct querying to EPCIS.

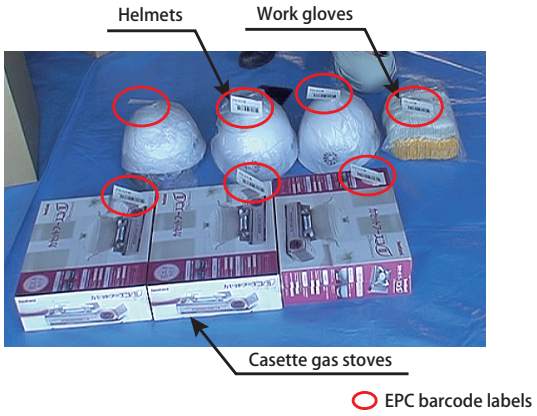

Fig. 5. Objects attached EPCs encoded in GS1-128 barcode labels

Fig. 6. A GS1-128 barcode label attached on a container

IV. Evaluation

This section evaluates functionality of Boxcan platform and performance of AggregationTracker. The evaluation is done by using the practical implementations of Boxcan platform in the object management system which is developed by us and now in practical operation. This system has 3 EPCIS repositories (and their interfaces) and one AggregationTracker applied on them. Fosstrak [7] is used as an implementation of EPCIS. AggregationTracker is implemented as a web service using Java servlet. EPCIS (fosstrak) and AggregationTracker are deployed on the same server. EPC master uses MySQL database with web front-end implemented with Ruby.

In the developed object management system, each object and container is attached a EPC encoded in a GS1-128 barcode as shown in Fig.5 and Fig.6. Boxcan web interface, one of applications of Boxcan platform, accepts an EPC expressed with this GS1-128 format, then converts it to pure identity URI [8], which is a format of EPCs used in information systems such as EPCIS. In a field test of the object management system, it is confirmed that Boxcan web interface works. Figure 7 is a screenshot of Boxcan web interface running on a tablet device. It is confirmed that other applications implemented on Boxcan platform, the inventory management application and the interworking function between non-EPC electronic commerce system and EPCIS, also work. These results prove that the system design of Boxcan is valid.

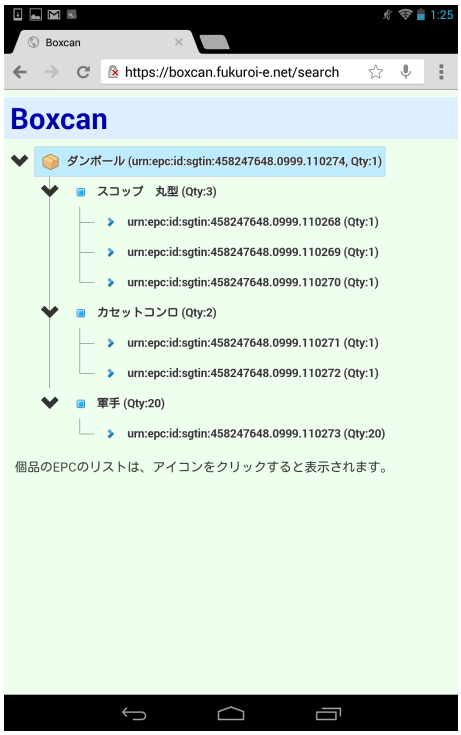

Fig. 7. Screen of Boxcan web interface : Names of objects (in Japanese) and corresponding EPCs and quantities are displayed in hierarchical view.

The effectiveness of caching EPCs' current parent-child tree by AggregationTracker is also evaluated with a comparison between performance of information retrieval by AggregationTracker and that by direct querying to EPCIS. In the experiment, the following two values are compared.

- Time duration which AggregationTracker takes to process one EPC which contains six other EPCs
- Time duration to retrieve the same information against the above EPC by direct querying to EPCIS

At the time of this experiment, the EPCIS repository recording 562 events, which is one of the above-mentioned three EPCIS repositories, is referred in the second case. AggregationTracker records 1994 EPCs' information collected from the three EPCIS repositories.

Figure 8 shows the distribution of time duration measured 100 times for each of the above two cases. The measured values includes time duration to retrieve master data of the EPCs, but it can be ignored in this performance comparison of parent-child tree retrieval because master data retrieval process is equally included to the two cases. From Fig.8, it is clear that AggregationTracker achieves shorter time duration than direct querying to EPCIS. In the average, the first case and second case take 0.07 and 0.34 seconds, respectively, and there is 0.27 second difference between the time duration of these two cases. This result proves the effectiveness of caching EPCs' current parent-child tree by AggregationTracker. Note that the direct querying to EPCIS will take much longer time to retrieve this information if implementing the absolutely same functionality to AggregationTracker, such as the application to multiple EPCIS repositories.

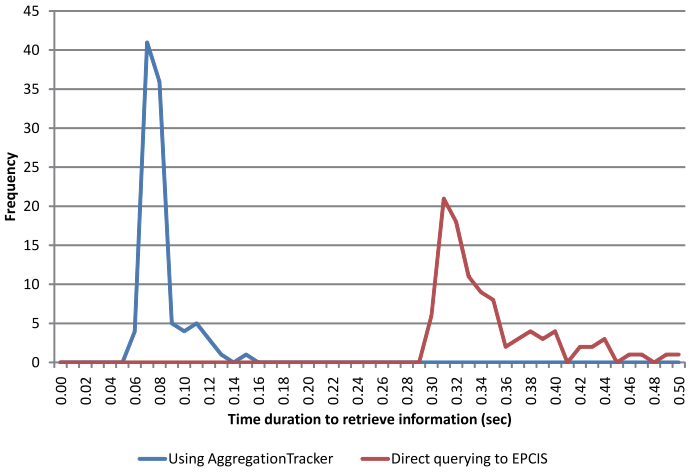

Fig. 8. Comparison of time duration to retrieve current parent-child tree of EPCs

V. Conclusion

Multiple applications which require information of EPCs current parent-child relationship can be implemented on Boxcan platform proposed in this paper. It is confirmed that the applications practically implemented on Boxcan platform works, thus the validity of the system design of Boxcan platform is proved. To realize Boxcan's functionality, it is a key technical challenge to realize fast retrieval of current parent-child tree of EPCs from EPCIS events. Caching EPCs' current parent-child tree is effective to solve this problem. In the experiment, the proposed caching method achieves approximately 5 times shorter delay to retrieve parent-child tree than the direct querying to EPCIS in the average.

References

[1] "The GS1 EPCglobal Architecture Framework," GS1 Final Version 1.5 Approved 23 March 2013.
[2] R. Wang and W. Gunthner, "Design and development of a traceability service for epc-enabled food supply chains," in *Software, Telecommunications and Computer Networks (SoftCOM), 2012 20th International Conference on*, Sept 2012, pp. 1–6.
[3] W. He, N. Zhang, P. Tan, E. Lee, T. Li, and T. Lim, "A secure RFID-based track and trace solution in supply chains," in *Industrial Informatics, 2008. INDIN 2008. 6th IEEE International Conference on*, July 2008, pp. 1364–1369.
[4] Y. Sato, M. Yoshida, K. Miyazaki, and M. Jin, "An information system for btoc e-commerce of agricultural products based on the epcglobal architecture," *The Transactions of the Institute of Electronics, Information and Communication Engineers D*, vol. 96, no. 10, pp. 2406–2417, 2013, (in Japanese).
[5] J. Mitsugi, Y. Sato, T. Yokoishi, T. Tashiro, T. Eda, K. Sato, and Y. Kishi, "Regional purchase orders dissemination and shipments aggregation of agricultural products with interworking epc network and edi system," in *RFID-Technologies and Applications (RFID-TA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1–6.
[6] "EPC Information Services (EPCIS) Version 1.0.1 Specification," errata Approved by TSC on September 21, 2007.
[7] "fosstrak - Open Source RFID Platform," https://code.google.com/p/fosstrak/.
[8] "EPC Tag Data Standard Version 1.5," august 18, 2010.