Australian Journal of Basic and Applied Sciences, 7(7): 555-562, 2013
ISSN 1991-8178

Object-oriented Modelling for Module-based Production Logistics Inventory System

[1]Khabbazi M.R., [1]Hasan M.K., [1]Shapi'i A., [2]Sulaiman R., [3]Keshavarz Y., [4]Mousavi A.

[1]Department of Industrial Computing, FTSM, National University of Malaysia, Bangi, Malaysia.
[2]Institute of Visual Informatics, FTSM, National University of Malaysia, Bangi, Malaysia.
[3]Department of Marketing, Faculty of Economic and Management, University Putra Malaysia, Malaysia.
[4]Departmentof Electronic & Information Engineering, Universitat Degli Studi Di Salerno, Salerno, Italy.

Abstract: This paper proposes module-based object-oriented data models for inventory system focusing on the production logistics business processes. It expounds the methodology and modelling procedure to provide the inventory system requirements. Through warehousing business process analysis for production logistics and based on the object-oriented technique in a modular basis, the domain and entity class diagrams are modelled. Through identifying all the required and realizing system interfaces, the system is able to be integrated with other back office systems. The model is able to manipulate all warehousing operation data including receiving, storing, retrieval, allocations and traceability, and load balancing for actual inventory stock keeping units in real-time to support quick decision making with minimum efforts and or errors.

Key words: computer-aided systems, inventory management system, production logistics, object-oriented design, UML.

INTRODUCTION

Almost every business from small to large need to practice some kind of inventory management to keep track of the supplies they provide for internal or external use (Lee and Wu, 2006, Sana, 2012). The role of the responsive information system has always been considered as the final solution and seen dramatically beneficial while it is integrated with other parts of enterprise systems to real-time control leading to timely decision makings(Rabinovich *et al*., 2003, Vries, 2007, Yao *et al*., 2007). An inventory management system that can be integrated with other back-office management systems provides a competitive edge. It provides abilities to plan effectively, execute predictably with customers, and minimize labour costs and errors associated with manual reconciliations. An efficient inventory management system developed based on the business process analysis and accurate data modelling effort provides the valuable fundamental input data for numerous inventory control applications such as determining safety level (SL) to calculate safety stock (SS) (Ruiz-Torres and Mahmoodi, 2010), demand forecast, or re-order point (ROP) which is vital in enterprise design and performance evaluation (Razi and Tarn, 2003, Hung *et al*., 2006, He *et al*., 2002). Moreover, data models plays a key role at providing considerable amount of required data for applications like stock out, stock loss (Kang and Gershwin, 2005) or new product development, lean environment, and etc. to respond quickly at changes or redesign in short time (Harding and Yu, 1999).

Maintaining the optimum level of inventory in production logistics is of the main objectives of any inventory system as abundantly concerned in severalresearches(Shapiro and Wagner, 2009). Both periodic or continues alternative reviews of inventory control policies (Boute *et al*., 2007) basically demand a perpetual inventory system in which tracks the receipt and use of inventory and calculates the quantity-on-hand is a fundamental requirement to efficiently address such need. Despite plenty of devoted researches on inventory system analysis and general database design methodologies for integrated information systems, still more precise and coherent models are ambitiously desirable(Halsall and Price, 1999, Dean *et al*., 2007, Khabbazi *et al*., 2011).

As a part of larger effort on conducting an extensive modellingformodule-based inbound and outbound e-logistics system at the supply chain level,this paper representsthe development of the object-oriented inventory information system focusing on the production logistics (i.e. inbound logistics) section at the highest domain level.The main contribution of this paper is within the concept of modularity paradigm. Using the object-oriented design all the provided and required interfaces in inventory information systems in related to the other back office production logistics information systems such as manufacturing information system and quality information system are identified. Moreover, the explanatory technique in data modelling including the business processes, and providing domain and entity meta-modelsusing class diagrams describe the structure and

Corresponding Author: M.R. Khabbazi, Department of Industrial Computing, Faculty of Information Science and Technology, UKM, Bangi, Malaysia.
E-mail: mrkabbazi@gmail.com

behaviour of the system extensively. The proposed object-oriented models are considered referential for further development in lower abstract levels and integration capabilities in an enterprise.

The reminder of this paper is constructed in five other sections. This introduction is followed by inventory system and requirements at section two. Section three presents the datamodelling methods. Section four presents data modelling for inventory system including the adopted procedure, domain class diagram, and entity class diagram. Conclusion is presented at section five.

MATERIAL AND METHOD

Inventory System and Requirements:

Effective inventory management is all about knowing what is on hand, where it is in use, and how much the finished product will be exactly produced. Clearly, the information management plays a key role such as the decision-making at planning & control applications and redesigns and improvements. Here, the concept of inventory is within the information domain through modelling of storing data regarding the elements and requirements of an inventory system. Inventory model proposed by Heizer (2004) emphasizes the controlling available resources and supplying resources to the Shopfloor. The designed system should address: (1) reception, (2) replenishment, (3) inventory allocation (i.e. assigning available stock to customer orders), (4) order assignment (i.e. assigning resources to work orders/stations), (5) load balancing and picking control and (6) packing, labelling and consolidation.Moreover, inventory management mainly takes into account the following issues:

***Item Types and Classifications*:**

There are different frameworks with purposes to categorize inventory items. ABC classification as an instance is basically based on importance of inventory items used in frequency audit, physical security and process control over criterion schemes such as single or multi-criteria (Rezaei and Dowlatshahi, 2010). The item types are controversially categorized into four groups as raw material, work-in-progress, rework/consumable supply, and finished goods (Ruiz *et al*., 2011).

***BOM*:**

Bill of Material (BOM) describes the dependency demand between items. Inventory management and material resources planning (MRP) determine quantity and timing of dependent demand items basically based on BOM (Yeh, 1994).

***Replenishment Policies*:**

There are generally two main model types: Continuous, and Periodic (Boute *et al*., 2007). Continuous models are either follow fixed order-quantity policies such as economic order quantity (EOQ), production order quantity (POQ) and quantity discount (QD) or probabilistic policies which allow demand to vary and consider service levels and safety stocks (Ruiz-Torres and Mahmoodi, 2010). Periodic models are fixed order-period models where orders are placed at fixed intervals periodically such as order-up-to policy.

***Traceability*:**

Assigning the resources to operational process is not enough where inventory system should address the real-time inventory amounts within lots/batches as well. Numerous studies and models been carried out at traceability such as (Jansen-Vullers *et al*., 2003, Khabbazi *et al*., 2010b, Khabbazi *et al*., 2010a) to name a few.

To establish an efficient inventory system, considering identified concepts and requirements are essential and hence based on the recognised elements, the development of inventory system modelling is carried out.

Data Modelling Methods:

Data model is often used as the first visual plan in designing a database and Object-oriented programming as well (Stair and Reynolds, 2006). Data modelling methods are formalizing data entities and relationships between entities. There generally are four data modelling methods including Richard Barker methodology (Barker, 1990), IDEF1x (IDEF1x, 1993), Entity-relationship model (ERM) (Chen, 1976), and unified modelling language (UML) (Booch *et al*., 2000). All of these methods define a set of notations to express data entities and relationships among them (Xiaoqing and Yun, 2008). Unified Modelling Language (UML) and Entity-Relationship Model (ERM) are the most common ways to describe data structures (Rönkkö, 2006).

***Unified Modelling Language*:**

UML offers a standard way of object-oriented design to visualize a system's architectural blueprints, including Datamodelling and Database Schema. UML model serves as the foundation of most of the works on Object-oriented analysis and design methodologies and Semantic Web. It combines techniques from data

modelling (i.e. ERDs), business process modelling, object modelling, and component modelling. UML is adopted for data modelling for several reasons: First and most important of all is the comprehensiveness. All the constraints, triggers, schemes, indexes, store procedures and more can be modelled with UML. Secondly, it is well suited for all purposes of high level-data model including Relational, Dimensional, Business perspectives, and Application perspectives. Lastly, its applicability has been well verified and employed in a huge number of scientific research and applications (for example: Tsai and Sato, 2004, Yang *et al*., 2009, Xiaoqing and Yun, 2008).

Adopted Procedure for Data Modelling:

The initial step is to identify the main classes in inventory system which is made by analysing the inventory business process through case studies observations as the fundamental inputs. All classifiers and instance classes at the domain highest level of the system are identified. Some of the identified classes are basically the main classes in other systems that are associated with the main classes in inventory which thereof are referred through the attribute data types of the classes. Next, data modelling is continued with adding the relationships to the main classes at the running inventory system. As the result, the Domain class diagram is generated. Providing a general structural view of the system through Domain class diagrams is significant. However, one of the significant usages of domain class diagrams are on spotting the dynamic repository classes called as the Entity classes.Next, the entity classeswill be specifically elaborated. The scopes, attributes, data types, and possible methods/operations of identified entity classes as well as the multiplicities of their relationships are conceptualized. As the result,the Entity class diagram which isused as the database blueprint on developing the prototype information system is generated.

Inventory System Domain Class Diagram:

Inventory system assumably is run at the Warehouse division and Warehouse Staff are responsible for all the activities in the system. However, the otherwise is of no consequences as long as system authoritative is defined and preserved. It supports all process and sub-processes pertaining to activities such as docking, store and retrieval, initial lot issuance and lot termination, and returning rejected purchases through governing a variation of sent and received messages and notifications, confirmation/rejection/verification decisions, and lookup requisitions of other co-operative systems.Inventory business processes and data flow diagramof the system are initially employed to analyse the system behaviour.

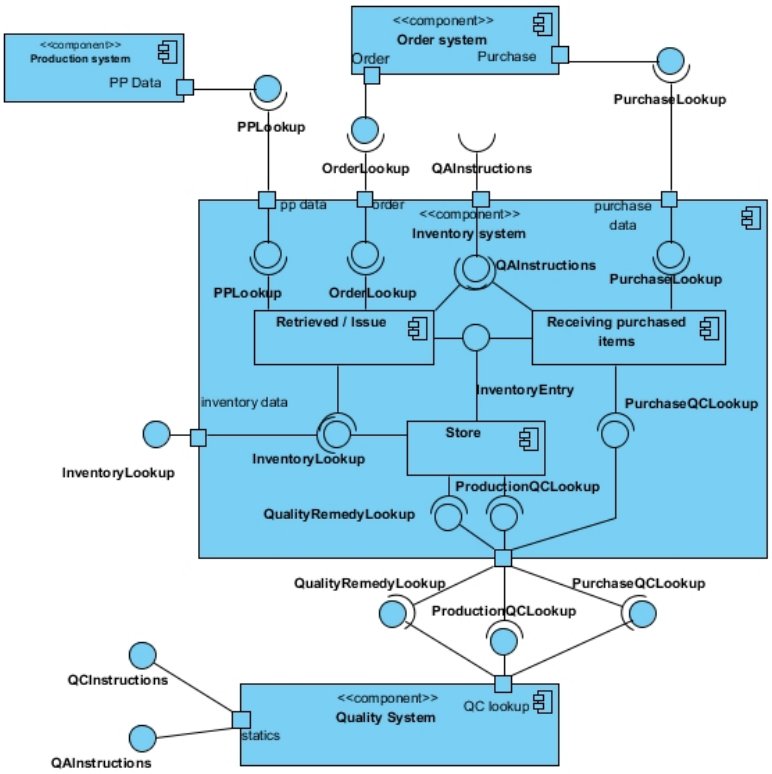

Fig. 1: Inventory System Component Diagram.

Inventory system is composed of three general internal components of retrieve/issue, receive, and store. As is illustrated in Fig. 1, the required and provided internal and external interfaces are identified through which the system can work properly and efficiently. External interconnections are to acquire detail information about production planning, purchase, order, and quality results from the Production, Order, and Quality systems. These identified interfaces determine the dependency (i.e. required) and realizing (i.e. provided) information from and for the external systems that might be included either as the other integrated production logistics modules (components) or as the external back office systems.

Table 1 displays a classification of all involving identified classes at Inventory System including Actors, Departments classes, Interfaces, Received Notifications, Primitives & Enumerations, Commons, and Entity classes. Inventory system has *cooperative* relationships with several Department classes, *Realizes* InventoryLookup while it has "Dependency" relationship with the remaining listed Interface classes. Condition class is identified as <<primitive>> data type, ActionBound and Issuer as <<Enummeration>> classes. Common classes are those of which appear in some other modules and same as Entity classes listed at the last column instantiate new instances at running system and are the blueprint of the Database *Tables* to store dynamic data.

Next the Cardinality of the class diagram is explained. "one" Ware house Staff instantiates Docked class with "one-to-many" new instance upon PurchaseOrder arrival from either Supplier or Bidder using QA Instructions. A new instance of PAN class is generated to notify purchase arrival to Sales Dept and Quality Dept accordingly.

Table 1: Identified Inventory System Classes.

Actors	*Departments*	*External Interfaces*	*Notifications Received*	*Primitive & Enummeration*	*Commons*	Entity
WarehouseStaff	Warehouse	InventoryLookup	PurQCR	Condition	Lot	Docked
Bidder	SalesDept	PPLookup	RetPurN	ActionBound	Relation	PAN
Supplier	ProductionDept	OrderLookup	PPD	Issuer	Retrieved	ReturnPurchase
	Shopfloor	PurchaseQCLookup	SSIN		Item	Stored
	QualityDept	PurchaseLookup	SFGN			Retrieved
		ProductionQCLookup				Issued
Legend						
PPLookup	Production Plan Lookup					
PurQCR	Purchase Quality Control Result					
PPD	Production Plan Document					
SSIN	Store Segregated Item Notification					
RetPurN	Return Purchase Notification					
PAN	Purchase Arrival Notification					
SFGN	Store Finished Goods Notification					

Ware house Staff later updates "one-to-many" Docked instances with referring to the related received "one-to-many" Pur QCR instance from Quality Dept using Purchase Lookup and Purchase QC Lookup.

In case of quality issues with the purchased items where "zero-to-many" RetPurN from SalesDept is received, "one" Warehouse Staff instantiates Return Purchase class with "zero-to-many" new instances which triggers transferring docked items to be returned which modelled as "zero-to-many" Docked instances are associated with "zero-to-one" Return Purchase instance. Eventually, "one" Docked instance is associated with "zero-to-many" Stored instances at Warehouse. All Zeros in this scenario indicate *Normal* docking with no quality problems therefore no instance for Return Purchase class is instantiated.

Stored class is instantiated with new instance generally in two identified cases either storing docked items, or storing lots containing items with different conditions such as considering as available stock or as particular required actions bound to them. "one" Ware house Staff instantiates Stored class with "one-to-many" new instances upon:

- Receiving "one" SFGN instance from Shopfloor and uses Order Lookup and Production QC Lookup.
- Receiving "zero-to-many" SSIN instances from QualityDept and uses OrderLookup and Quality Remedy Lookup. Zero indicates the cases of no quality issues and therefore no received SSIN instance and no Stored class instantiation.

In both mentioned cases of Stored class instantiation, "one" Lot is stored by "one-to-many" instances of Stored class. Therefore consequently, the life of "zero-to-one" instance of Relation class which registers the biography of each lot is ended up by storing process and by updating "zero-to-one" instances of Stored class. Zero indicates the relations between living lots (e.g. the consumed parent lots of which they are not ended up at string).

"one" WarehouseStaff receives "one-to-many" PPD instances from the Production Department. Retrieval process of stored items either as raw material or action bound items for received production plan triggers the instantiation of "one-to-many" instances of Retrieved class using PP Lookup, Order Lookup, and Inventory

Lookup itself too. Another possible retrieval instantiation is at Shipping System module triggered by pick and pack order. “one” Retrieved instance associates with “one-to-many” new instances of Issued class that instantiate new Lot instance for each.

Issued class is the classifier of new Lot instance generation and registers the issuer which might be triggered by of one of the three following modules:

- Issued by Inventory System (i.e. caused by Retrieved instance),
- Issued by Quality System (i.e. caused by QualityRemedy instance),
- Or issued by Production System (i.e. caused by Operation instance).

Fig. 2 illustrates the final view of Domain class diagram for Inventory system data-model. Even though, a Domain class diagram does not demonstrate the sequential events or processes like business process models nevertheless all business processes derived from them are perceivable. As described, some of the Entity classes are notification or a message such as PAN (Purchase Arrival Notification) class which is generated right after registration of new record in Docked class and is sent to SalesDept and QualityDept. Some of these messages like PPD (Production Plan Document) are still categorized as a message or notification which are received by other modules and departments and in this case by ProductionDept invoking Retrieved class to instantiate new instances. In Inventory System Domain class diagram, several kinds of Inventory Lookups are set to be realized by the Stored class providing the real-time information about the amounts of SKUs at the lots as well as the stock-on-hand detailed with the time, date, and responsible staff.

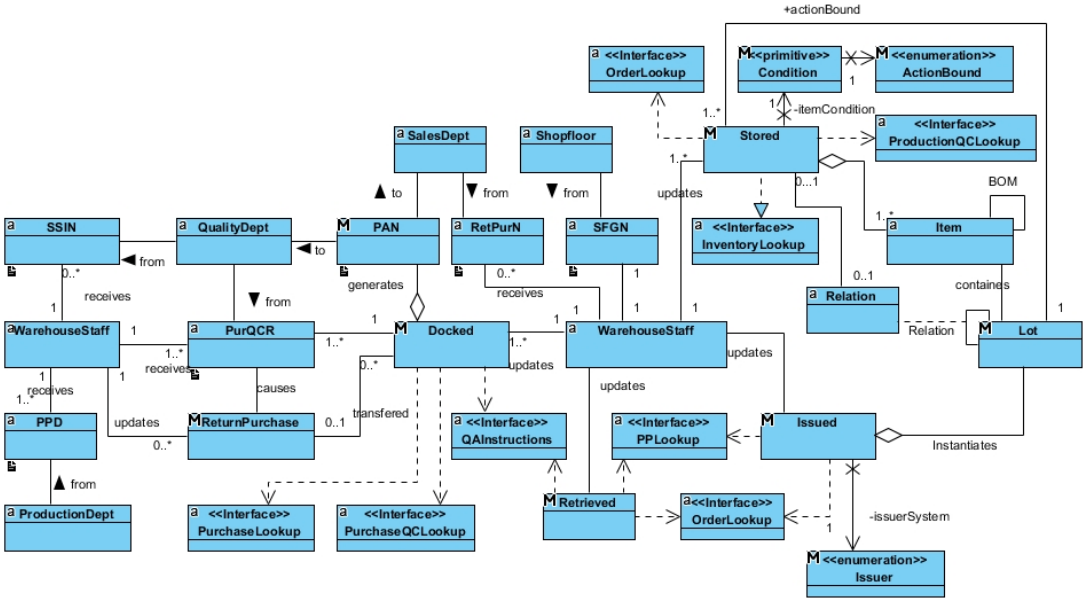

Fig. 2: Inventory System Domain Class Diagram.

Inventory System Entity Class Diagram:

Docked, Stored, Retrieved, Issued, and ReturnPurchase are identified as the main Entity Classes in which the essential dynamic Inventory System data are meant to be stored. However, Retrieved class is considered as common in other systemssuch as shipping system. PAN class is the notification Entity Classes helping the system work responsively at clerking affairs. Lot and Relation are to register the beginning and the end of the generated lots which are issued and terminated at Inventory System. Condition is <<Primitive>>*data type* class and Issuer and ActionBound classes are predefined <<Enummeration>> Classes serving as various purposes such as guidelines, human error prevention, efficient operability, etc. These Classes are elaborated next with their attributes, data type scopes, and relationships with each other.

Docked class specifies the collected data for each particular purchase arrival and docking operation. One specific instantiated instance of Docked class might be updated at different phases of docking process. As instance, the result of QC tests for docked items is updated upon receiving them after some time.

Any identifiable actual storing process at warehouse is classified as a new instance in Stored class to register its associated data. Through some designed automatic computing, the dynamic amount of items becomes achievable in real-time. Updating the actual item amount should be carried out at retrieval. Therefore similarly, one specific instance of Stored class also might be updated at different points of time accordingly.

Retrieved class specifies retrieved items from warehouse and their associate data. Retrieval process is triggered by receiving either production plan to provide required resources for production or by pick and pack order for the shipment. “0…1” instance of the Retrieved class is associated to “1…*” instance of Issued class.

Zero in here implies those cases of retrieval for shipment which would not trigger any instance initiation of the Issued class.

Issued class specifies all information about new lot instantiation and has "Aggregation" type association with Lot class meaning that for each instance of Lot class there would be a referring instance in the Issued class respectively. Fig.3 displays the final view of the Entity Class diagram for Inventory System.

A single instance of every Entity Class has a unique identification integer number displayed as *": int"* in data type scope section and it is listed at the top attribute compartment. These attributes are considered as Primary keys in developing the database step. Some of attributes are for associating purposes. These attributes will be transformed into the Foreign keys at database development. The identified Primary and Foreign keys for each Entity classes are listed at Table 2. Next, all attributes for each class are discussed.

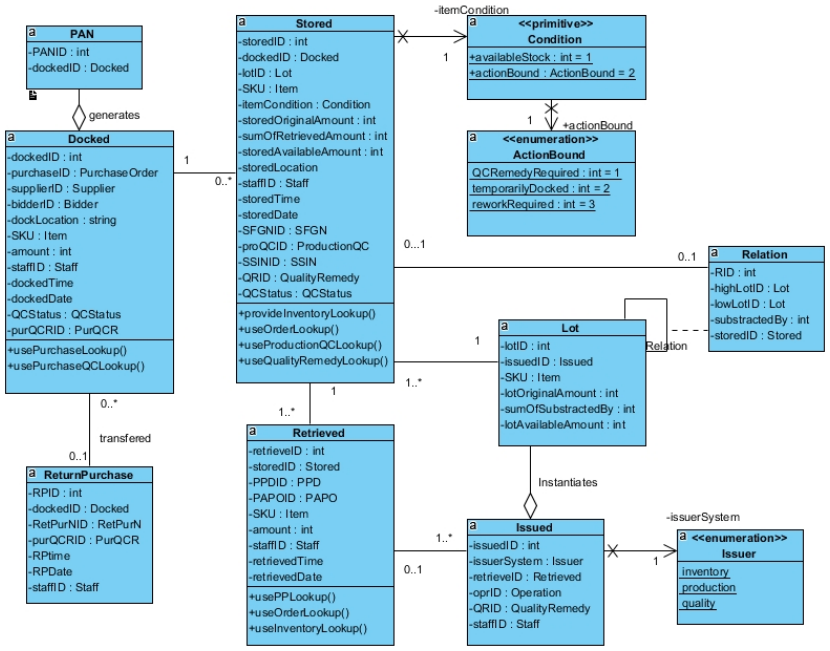

Fig. 3: Inventory System Entity Class Diagram.

Table 2: Primary and Foreign Keys at Inventory System Entity Classes.

Entity Class (Name)	*Unique identifier (primary key)*	*Referenced identifier (foreign keys)*
Docked	dockedID	purchaseID, supplierID, bidderID, SKU, staffID, QCStatus, purQCRID
PAN	PANID	dockedID
Stored	storedID	dockedID, lotID, SKU, itemCondition, staff, SFGN, proQCID, SSIN, QRID, QCStatus
Condition	-	-
ActionBound	-	-
Retrieved	retrievedID	storedID, PPDID, PAPOID, staff
Issued	issuedID	issuerSystem, oprID, QRID, staff
Issuer	-	-
Lot	lotID	issuedID, SKU
Relation	RID	highLotID, lowLotID, storedID
ReturnPurchase	RPID	dockedID, RetPurNID, purQCRID, staff

Docked (dockedID: int) is composed of information about the purchase and the vendor such as purchaseID: PurchaseOrder, supplierID: Supplier, or bidderID: Bidder; Docked items specifications, location, and dates such as dockLocation: string, SKU: Item, and amount: int, dockedTime, dockedDate; Responsible staffID: Staff; and referring data to quality control operations according to received results such as QCStatus: QCStatus, and PurQCRID: PurQCR.

PAN (PANID: int) contains information about related dockedID: Docked.

ReturnPurchase (RPID: int) is composed of information about related dockedID: Docked; Related received notification of retPurNID: RetPurN from SalesDept; Related QC result of purQCRID: PurQCR; Responsible staffID: Staff; and information about the date and time of purchase return such as PRTime, and PRDate.

Stored (storedID: int) is composed of information about related dockedID: Docked; Associate lotID: Lot;

Information about the item, its condition, its location, time and date, and its dynamic amount at ware house such as SKU: Item, item Condition: Condition, stored Location, stored Original Amount: int, sum of Retreived Amount: int, stored Available Amount: int, stored Time, and stored Date; Responsible staffID: Staff; it also concludes information about referring production operation and quality operation notifications and links which initiated the Stored class instance such as SFGNID: SFGN, pro QCID: Production QC, SSINID: SSIN, QRID: Quality Remedy, and QC Status: QC Status.

Conditionis defined as primitive public data types of +available Stock: int=1, and +Action Bound: Action Bound=2.

Action Boundis enumerated by QC Remedy Required: int=1, temporarily Docked: int=2, and rework Required: int=3.

Retrieved (retrievedID: int) is composed of information about related storedID: Stored; Related Pick and Pack order PAPOID: PAPO; Related production plan of PPDID: PPD; Responsible staffID: Staff; and information about the item, amount of retrieval and dates such as SKU: Item, amount: int, retrievedTime, and retrievedDate.

Issued (issuedID: int) is composed of information about the retrievedID: Retrieved; Responsible staffID: Staff; Determined issuer through issuerSystem: Issuer; and related external issuer either by an oprID: Operation or QRID: QualityRemedy instance.

Issuer is enumerated with inventory, production, and quality systems.

Conclusion:

This paper focused on the modular-basis development of data modelling for production logistics inventory system at its highest domain level. The research was carried out to identify all business process and system requirements for the inventory system and carefully addressed them at the data modelling development with UML. All inventory system interfaces which are required for the integration purposes with the other back office systems in production logisticsare identified and addressed in the models. As such, the structure of the messaging and notification in the modular-baseddesign are determined. The entire data modelling procedure achieving the UML domain and entity class diagrams to demonstrate the system structure, behaviour and database blueprint are explained.The models are able tomanipulatethe production logistics inventory data dynamically to keep the information up-to-date and provide report generating system for applicationssuch as inventory allocation, receptions, and order-assignments. The model is able to manipulate all warehousing operation data including receiving, storing, retrieval, allocations and traceability, and load balancing for actual inventory Stock Keeping Units (SKU) in real-time stored either at the Warehouse or in the lots at the Shopfloor including item types, conditions, and rates.Through enhanced visibility and dynamic store/retrieval of data, all inventory data applications for inbound logistics are well responded to support quick decision making with minimum efforts and or errors.

The designed model is limited to inventory management system for a single company environment with basically batch production system for make-to-order. To keep the referentiality purpose of the model,the replenishment policies requirements such as defining safety stock has seen out-scope and are excluded in the modelling process. Since the solution is a research note in manufacturing control and logistics systems, the data model is initially applicable to environment with identical operational boundaries and complexity. It is expected that such a datamodel can be further developed in a way that can be integrated in the middle-level (e.g. knowledge management) or executable level to advocate inventory control system effectively.

REFERENCES

Barker, R., 1990. Case Method: Entity Relationship Modeling, Addison-Wesley.

Booch, G., I. Jacobson, J. Rumbaugh, 2000. 10 September. OMG Unified Modeling Language Specification. Available: http://www.omg.org/docs/formal/00-03-01.pdf.

Boute, R.N., S.M. Disney, M.R. Lambrecht, B.V. Houdt, 2007. An integrated production and inventory model to dampen upstream demand variability in the supply chain. European Journal of Operational Research, 178: 121-142.

Chen, P.P.S., 1976. The entity-relationship model - toward a unified view of data. ACM Transactions on Database Systems (TODS), 1: 9-36.

Dean, P.R., Y.L. Tu, D. Xue, 2007. An information system for one-of-a-kind production. International Journal of Production Research, 47: 1071-1087.

Halsall, D.N., D.H.R. Price, 1999. A DSS approach to developing systems to support production planning and control in smaller companies. International journal of Production Research, 37: 1645-1660.

Harding, J.A., B. Yu, 1999. Information-centred enterprise design supported by a factory data model and data warehousing. Computers in Industry, 40: 23-36.

He, Q.M., E.M. Jewkes, J. Buzacott, 2002. The value of information used in inventory control of a make-to-order inventory-production system. IIE Transactions, 34: 999-1013.

Heizer, J., B. Render, 2004. Principles of operations management, Pearson: Prentice-Hall, Inc.

Hung, W.Y., N.J. Samsatli, N. Shah, 2006. Object-oriented dynamic supply-chain modelling incorporated with production scheduling. European Journal of Operational Research, 169: 1064-1076.

Idef1x, 1993. July Announcing the standard for Integration Definition For Information Modeling (IDEF1x). Available: http://www.idef.com/IDEF1x.html.

Jansen-Vullers, M.H., C.a.V. Dorp, A.J.M. Beulens, 2003. Managing traceability information in manufacture. International journal of Information Management, 23: 395-413.

Kang, Y., S.B. Gershwin, 2005. Information inaccuracy in inventory systems: stock loss and stockout. IIE Transactions, 37: 843-859.

Khabbazi, M.R., M.K. Hasan, R. Sulaiman, S.A. Mousavi, 2010a. Extending quality data for lot-based traceability system in SME. Information Technology (ITSim), 2010 International Symposium in. Kuala Lumpur, Malaysia, IEEE.

Khabbazi, M.R., M.D.Y. Ismail, N. Ismail, S.A. Mousavi, 2010b. Modeling of Traceability Information System for Material Flow Control Data. Australian Journal of Basic and Applied Sciences, 4: 208-216.

Khabbazi, M.R., M.Y. Ismail, N. Ismail, S.A. Mousavi, H.S. Mirsanei, 2011. Lot-base traceability requirements and functionality evaluation for small- to medium-sized enterprises. International Journal of Production Research, 49: 731-746.

Lee, H.T., J.C. Wu, 2006. A study on inventory replenishment policies in a two-echelon supply chain system. Computers & Industrial Engineering, 51: 257-263.

Rabinovich, E., M.E. Dresner, P.T. Evers, 2003. Assessing the effects of operational processes and information systems on inventory performance. Journal of Operations Management, 21: 63-80.

Razi, M.A., J.M. Tarn, 2003. An applied model for improving inventory management in ERP system. Logistics information management, 16: 114-124.

Rezaei, J., S. Dowlatshahi, 2010. A rule-based multi-criteria approach to inventory classification. International Journal of Production Research, 48: 7107-7126.

Rönkkö, M., 2006. A model for item centric material control in manufacturing. Industrial Engineering and Management. Finland, Helsinki University of Technology.

Ruiz-Torres, A.J., F. Mahmoodi, 2010. Safety stock determination based on parametric lead time and demand information. International Journal of Production Research, 48: 2841-2857.

Ruiz, N., A. Giret, V. Botti, V. Feria, 2011. Agent-supported simulation environment for intelligent manufacturing and warehouse management systems. International Journal of Production Research, 49: 1469-1482.

Sana, S.S., 2012. A collaborating inventory model in a supply chain. Economic Modelling, 29: 2016-2023.

Shapiro, J.F., S.N. Wagner, 2009. Strategic Inventory Optimization. Journal of Business Logistics, 30: 161-173.

Stair, R., G. Reynolds, 2006. Fundamentals of Information Systems, Boston, Thomson.

Tsai, T., R. Sato, 2004. A UML model of agile production planning and control system. Computers in Industry, 53: 133-152.

Vries, J.D., 2007. Diagnosing inventory management systems: An empirical evaluation of a conceptual approach. International Journal of Production Economics, 108: 63-73.

Xiaoqing, T., H. Yun, 2008. Data model for quality in product lifecycle. Computers in Industry, 59: 167-179.

Yang, D., H. Wu, L. Tong, 2009. A UML-based approach for the development of shop floor control systems. International Journal of Production Research, 47: 1601-1633.

Yao, Y., P.T. Evers, M.E. Dresner, 2007. Supply chain integration in vendor-managed inventory. Decision Support Systems, 43: 663- 674.

Yeh, C.H., 1994. Production data modelling: an integrated approach. International Journal of Operations & Production Management, 15: 52-62.