Journal of Theoretical and Applied Information Technology
31st August 2013. Vol. 54 No.3
© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

INVENTORY SYSTEM AND FUNCTIONALITY EVALUATION FOR PRODUCTION LOGISTICS

[1*]KHABBAZI M.R., [1]HASAN M.K., [1] SHAPI'I A., [2]SULAIMAN R., [1]TAEI-ZADEH A.

[1]Department of Industrial Computing, FTSM, University Kebangsaan Malaysia, Bangi, Malaysia

[2] Institute of Visual Informatics, FTSM, University Kebangsaan Malaysia, Bangi, Malaysia

E-mail: *mrkabbazi@gmail.com

ABSTRACT

This paper addressed a comprehensive modelling and functionality requirement evaluation of perpetual production logistics inventory system for small-to-medium enterprises. The main idea is to analyse the system behaviour and data transactions invoked by a module-based inventory system using business process modelling and object-oriented data modelling techniques. The paper explains through the methodology and modelling procedure which eventually leads to a computerized inventory information system as a module able to be integrated with other back office systems in production logistics. The system is able to manipulate all inbound warehousing operations including docking, storing, and retrieval data as well as actual inventory SKU balance dynamically to keep the inventory data instantly up-to-date and in real-time. The functionality of the solution is evaluated based on the identified inventory system requirements responding to all inventory controlling of receiving, allocating, load balancing, and traceability linking through different types of lookups and data queries.

Keywords: *Computer-aided Systems, Production Logistics, Inventory Management System, BPMN, UML.*

1. INTRODUCTION

Managing inventory is an indispensable part of the larger practice of supply chain management [1,2]. Nearly every business worldwide, from small to large, needs to do some kind or another inventory management to keep track of supplies they provide for internal use or to sell.

The role of IT and responsive information system has always been considered as the final solution and dramatically beneficial while integrated with other parts of enterprise automated systems for real-time controllability and lookups [3]. Inventory management and control cannot be as an isolated system [4]. A modular-basis inventory management system that can be integrated with other back-office systems provides an enterprise advantages in abilities to plan effectively, execute predictably with customers, and minimize labour costs and errors associated with manual reconciliations. Besides, the SMEs as the engine of economy growth in developing countries requires more attention on system automation and integration and therefore SME-IT alignment is seen dramatically required [5]. An efficient inventory management system developed from an accurate data modelling based on the comprehensive business process analysis effort provides the valuable fundamental input data for numerous inventory control applications in SMEs. Among others can be mentioned to the determining safety level (SL) to calculate safety stock (SS) [6], demand forecast, or re-order point (ROP) which is vital in enterprise design and performance evaluation [7]. In addition, it provides considerable amount of required data for applications such as stock out, stock loss [8] or new product development, lean environment, and etc. to respond quickly at changes or redesign in short time [9].

Maintaining the optimum level of inventory is of the main objectives of any inventory system as abundantly concerned in a large number of researches [10]. Both periodic or continues alternative reviews of inventory control policies [11] 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 database design methodologies in integrated information systems, still more precise and coherent models of all diverse supply chain functions are ambitiously desired [12,13].

The aim of this paper is to apply business process modelling and data modelling and functionality requirement evaluation of the production logistics inventory system applicable under an SME environment. Different techniques and tools were employed for an extensive business process and object-oriented data modelling for a perpetual modular-basis inventory system at its highest-level. This paper also addresses how the modelling of a computerized inventory system is conducted and implemented in a user-friendly and easy-to-understand way. Production logistics inventory controlling requirements are highlighted and functionalities of the solution based on the identified requirements are evaluated through data queries providing real-time controllability. The modular-based design is another advantage of the proposed system proving the ability for integration with other back-office system in SMEs.

2. INVENTORY SYSTEM AND REQUIREMENTS

Effective inventory management is all about knowing what is on hand, where it is in use, and how much finished product results. Vries [3] proposes a generic overview model of the management inventory systems by categorizing it into four different domains of which the management information architecture is one of the main significant. Clearly, management information architecture plays a key role for other domains such as the decision-making at planning & control applications and redesigns and improvements for physical infrastructure which are closely dependent to data provided by information domain being used. Here, the concept of inventory is within the information domain through modelling capturing business processes in production logistics and structure of storing the data based on the inventory elements and requirements.

Inventory model proposed by Heizer and Render [14] 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 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 [15,16]. The item types are controversially categorized into four groups as raw material, work-in-progress, rework/consumable supply, and finished goods [17]. Stock Keeping Units (SKUs) are unique codes or numbers to identify inventory items describing item type, properties and conditions.

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 [18]. **Usage Policies:** The inventory management is based on rules of material usage like FIFO (first input–first output), LIFO (last input–first output), criteria approaches to control inventory usages (e.g. food expiration date), etc. Usage policy concepts are significantly concerned in a several researches such as recent studies by [19,20].

Replenishment Policies: There are generally two main model types: Continuous, and Periodic [11]. 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 [6]. Periodic models are fixed order-period models where orders are placed at fixed intervals periodically such as order-up-to policy. Reviews and models on replenishment and control policies are abundant reclining on [2,21,22].

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. The importance of traceability gets bolded in applications like the need for order amount adjustment before reaching to a dangerous or unfavourable level. Numerous studies and models been carried out at traceability such as [23,24] 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 modelling is carried out.

3. MODLING DEVELOPMENT

3.1. Procedure Adopted for Inventory System Modeling

Based on the study on the production logistics inventory system requirements and the case observation and data-gathering on the car component industry focusing on the production logistics activities, the modelling requirement are identified. Capturing all the "as-is" activities and classification into domain highest and lowest level is carried out. Next, the business process modelling is accomplished using Business Process Modelling Notification (BPMN) standard achieving a referential "to-be" model for the highest domain level of production logistics inventory system. Next, based on the identified structure and behaviour of the system through business process modelling, the development of data modelling is carried out to conceptualize the structure of the system data transactions. As an object-oriented methodology, Unified Modelling Language (UML) is employed. The initial step was to identify the components of the system and interconnection with other back office systems. As such, UML component diagram is fabricated. Next, data modelling is continued with identification of all classes. The generated models and diagrams are the fundamental inputs which are carefully analysed. As the result, classifier and instance classes are identified and conceptualized in UML domain and entity class diagrams. This step is elaborately discussed within another article written by the authors. Next, the main entity classes are 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 a blueprint database design. This step includes the identification of foreign and primary keys as well as required tables, fields, relationships, and database properties to reach to the fabrication of prototype database. After finalizing the designed prototype database, the functionality of scenario based data loaded database is evaluated through examining data queries as lookups.

3.2. Business Process Modeling for Inventory System

Inventory BPM represents essential view of how the controlling, storing, and retrieval of physical objects including the raw material/outsourcings, work-in-progress, and finished products are accomplished. It provides key information of how the system works and how it is possible to develop a data system through which the available inventories and tracking and tracing of item/lot relations can be achieved. Supplier, Bidder, and Customer are identified as three "Black box" pools; Manufacturer as "Abstract" pool; Warehouse as lane; and Sales Dept., Production Dept. and Quality Dept. are considered as "Black box" lanes. At least four triggering message has been identified for the system to initiate.

Next, the reception activity at the highest-level of production logistics inventory system is explained. The *Purchased items received* "Parallel Multiple" start event upon arrival of purchased outsourcing items either initiated by *Supplier* or *Bidder* triggers the system. The purchased items are put away based on *QA instructions* for some time to get controlled and confirmed through *Temporary docking* "User" task. Later, *Purchased arrival notification* "Send" task notifies *Sales* and *Quality departments*. Upon receiving the *Quality results* as "Receive" intermediate event after a while a decision is made based on the QC results. *QC results OK* "Data-based exclusive" gateway determines two possible flows of *Yes* condition executes *Storing process* "Re-usable" sub-process and is followed by *Updating inventory system* task to register new outsourcing materials and traceability coding as new records into *Database*. Upon completion, the flow is finished with an *end* event. The other diverged flow of the *QC result OK* gateway with *No* condition reaches to the *Wait for return order* intermediate "Time" event "Attached" to *Receive return purchase order* "Receive" task. As such, *Return purchased items* sub-process is executed and the flow is terminated with an end event. Figure 1 illustrates an integrated framework of the domain BPM for production logistics inventory system.

Next, the order assigning and inventory allocation activities at the highest-level of production logistics inventory system are explained. *Production plan and required resources* "Catching Message" start event from *Production Department* triggers the system to issue the raw material for production purposes. *Retrieval/ picking required resources* "Embedded" sub-process is executed and followed by *Issue to production* "User" task. *Updating inventory system* task and *Deploy resources to Shopfloor* "Manual" task are performed respectively up to *Wait for finished goods* intermediate "Time" event until the *Finished goods received* from *Shopfloor* intermediate event. Next, *Storing process* "Reusable" sub-process is called through call activity element and is followed by *Updating inventory system* task to register the new records of available inventory and associate lot/item relations into the *Database* resulting the flow reaches to *end* event.

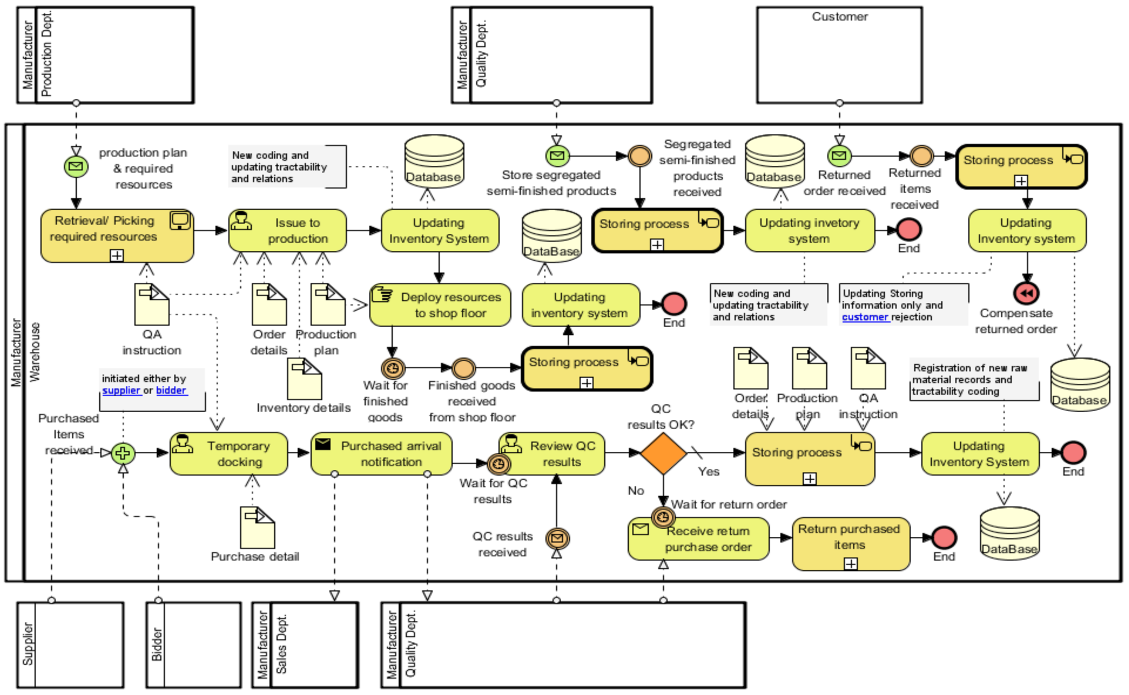
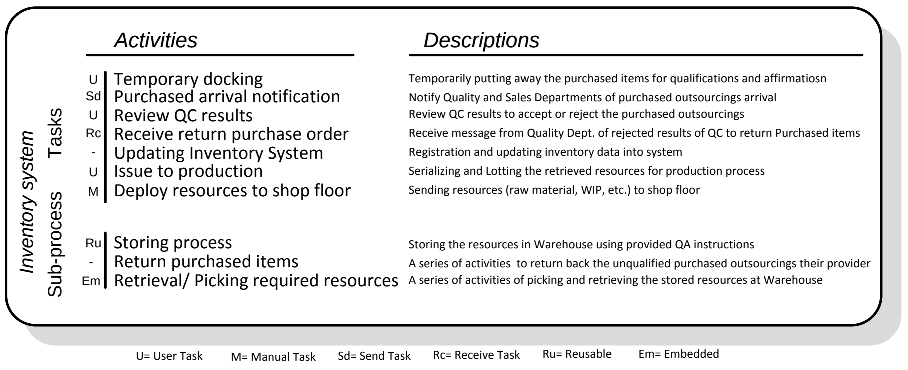

Manufacturer Production Dept.
Manufacturer Quality Dept.
Customer
Manufacturer Warehouse
production plan & required resources
New coding and updating tractability and relations
Database
Store segregated semi-finished products
Segregated semi-finished products received
Returned order received
Returned items received
Storing process
Retrieval/ Picking required resources
Issue to production
Updating Inventory System
DataBase
Updating invetory system
End
Updating Inventory system
QA instruction
Order details
Production plan
Deploy resources to shop floor
Updating inventory system
Updating Storing information only and customer rejection
Compensate returned order
initiated either by supplier or bidder
Wait for finished goods
Finished goods received from shop floor
Inventory details
Registration of new raw material records and tractability coding
Purchased Items received
Temporary docking
Purchased arrival notification
Review QC results
QC results OK?
Yes
No
Wait for QC results
QC results received
Wait for return order
Receive return purchase order
Return purchased items
Purchase detail
Supplier
Bidder
Manufacturer Sales Dept.

Figure 1: Domain Business Process Model for Production Logistics Inventory System

The other system initiation is on *storing the segregated semi-finished products* after remedy operations by Quality department. Through a "Catching" Store segregated semi-finished products start "Message" event inventory system is triggered and upon *Segregated semi-finished products received* intermediate event *Storing process* "Reusable" sub-process element is called once again. Next, the *Database* is updated with *Updating inventory system* task adding new inventory records and relations and the flow ends with an *end* event.

The last system initiation is through receiving a *Return order received* "Message" start event from the *Customer* due to the rejection of the sold items. As such, *Storing process* "Reusable" sub-process element is called and the *Database* is updated resulting with *Compensate returned order* "Compensation" end event which triggers the CRM system at the lower levels. Figure 2 provides additional details on every activity described in the modelling process.



Inventory system			Activities	Descriptions
	Tasks	U	Temporary docking	Temporarily putting away the purchased items for qualifications and affirmatiosn
		Sd	Purchased arrival notification	Notify Quality and Sales Departments of purchased outsourcings arrival
		U	Review QC results	Review QC results to accept or reject the purchased outsourcings
		Rc	Receive return purchase order	Receive message from Quality Dept. of rejected results of QC to return Purchased items
		-	Updating Inventory System	Registration and updating inventory data into system
		U	Issue to production	Serializing and Lotting the retrieved resources for production process
		M	Deploy resources to shop floor	Sending resources (raw material, WIP, etc.) to shop floor
	Sub-process	Ru	Storing process	Storing the resources in Warehouse using provided QA instructions
		-	Return purchased items	A series of activities to return back the unqualified purchased outsourcings their provider
		Em	Retrieval/ Picking required resources	A series of activities of picking and retrieving the stored resources at Warehouse

U= User Task M= Manual Task Sd= Send Task Rc= Receive Task Ru= Reusable Em= Embedded

Figure 2: Activities Descriptions of Production Logistics Inventory System

Journal of Theoretical and Applied Information Technology
31st August 2013. Vol. 54 No.3
© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

3.3. Data Modeling for Inventory System

Inventory system module 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. This module 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 system modules.

Inventory business process models and component diagrams are initially employed to identify the system behaviour. The component diagram illustrated at Figure 3 represents the internal and external components of the system and required interfaces of one component to another. Internal components are retrieve/issue, receive, and store while the external identified components are comprised of the quality system, order system including the sales and purchasing, and production system. The external interconnections (i.e. components) provide detail information and notifications about the production planning, purchase, order, and quality results of which the inventory system is depended into to work efficiently.

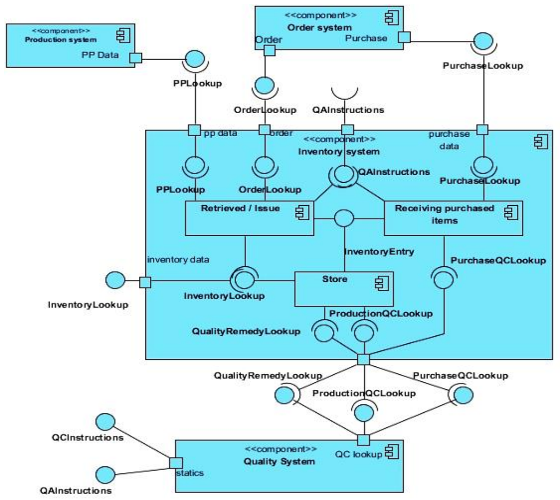


Figure 3: Inventory System Component Diagram

In addition, Table 1 displays a classification of all involving Classes at Inventory component 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 is dependant to the remaining listed Interface classes assumably provided by other components.

Table 1: Identified Involving Inventory System Classes

Actors	***Departments***	***Common***	**Entity**
WarehouseStaff	**Warehouse**	Lot	**Docked**
Bidder	SalesDept	Relation	**PAN**
Supplier	ProductionDept	Retrieved	**ReturnPurchase**
	Shopfloor	Item	**Stored**
	QualityDept		**Retrieved**
			Issued

Notifications Received	***Primitive & Enummeration***	***Interfaces***
PurQCR	Condition	**InventoryLookup**
RetPurN	ActionBound	PPLookup
PPD	Issuer	OrderLookup
SSIN		PurchaseQCLookup
SFGN		PurchaseLookup
		ProductionQCLookup

Legend	
PPLookup	**Production Plan Lookup**
PurQCR	**Purchase Quality Control Result**
PPD	**Production Plan Document**
RetPurN	**Return Purchase Notification**
PAN	**Purchase Arrival Notification**
SFGN	**Store Finished Goods Notification**
SSIN	**Store Segregated Item Notification**

Figure 4 illustrates the final view of Domain class diagram for Inventory system data model including all identified classes.

Docked, Stored, Retrieved, Issued, and ReturnPurchase are identified as the main Entity Classes in which the fundamental specific dynamic Inventory System data are meant to be stored. However, Retrieved class is considered as common in other modules responding to pick and pack requirement and notifications for shipping system. The description of each entity class is explained at Table 2.

Identified primary and foreign keys for each Entity classes are listed at Table 3. Next the data model is translated to logical model for a detailed finer view of the attributes (i.e. fields), keys, and relationships for development of prototype application as is shown in Figure 5.

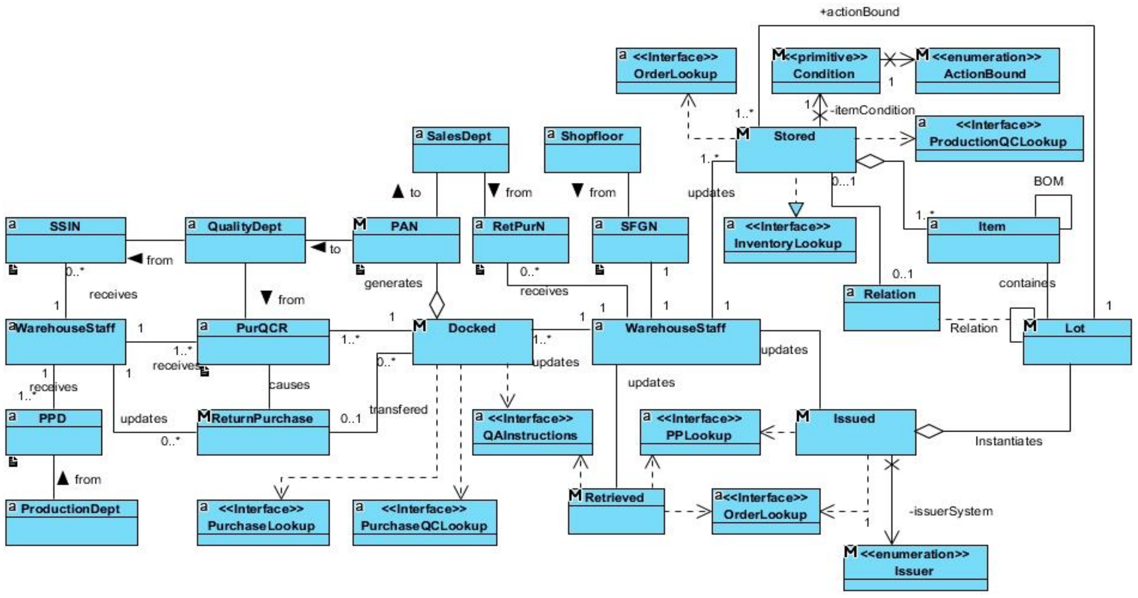


Figure 4: Domain Class Diagram for Modular Inventory System

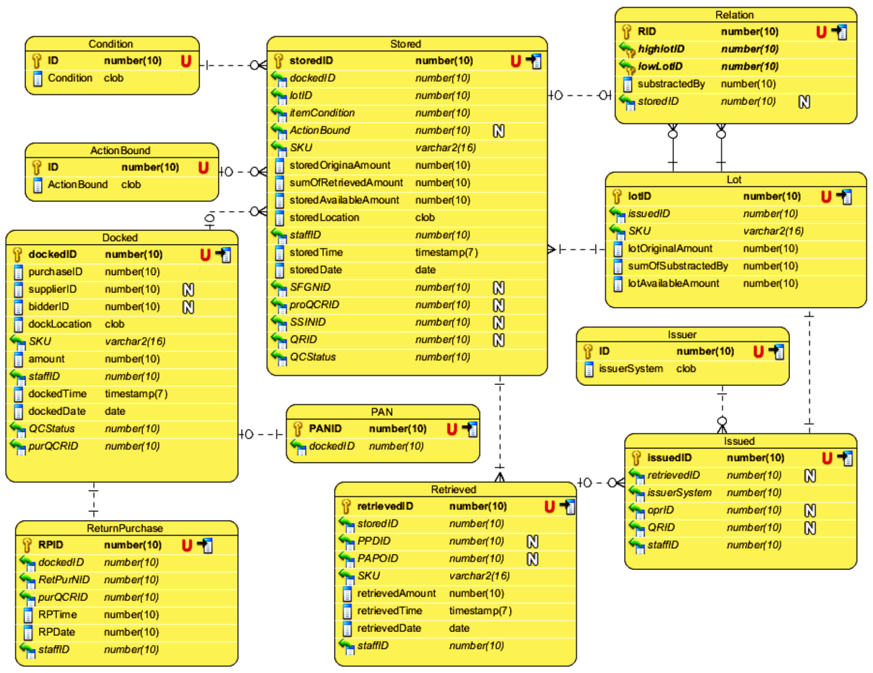


Figure 5: Attributes and relationships of inventory system logical data model

Table 2: Data Dictionary for Identified Entity Classes

Entity class	*Description*
Docked	Hold informaiton of receptions and docking operations. records update on events such as QC results.
Stored	Hold informaiton of storing operations at warehouse and associated linked data. Records and actual amounts update on events.
Retrieved	Hold informaiton of Retrieval operations for production allocation or shipment. A common class.
Issued	Hold informaiton of new lot instantiations.
ReturnPurchase	Hold informaiton of returning operation of purchased items. Records update on events.
PAN	Notoficaiton class of purchasing arrival
Condition	Primitive data type class for stored items set as "+availableStock:int=1" and "+actionBound:int=2"
ActionBound	Enummeration class for Condition datatype class set as "QcremedyRequired"int=1", "temporaryDocked:int=2", and "reworkRequired:int=3"
Issuer	Enummeration class for Issued class set as "inventory", "production", and "quality".
Lot	Informaiton identifies every lot in the system with unique number and amounts
Relation	Holds information of "where used", "where from", "how used", and "how exterminated" for lots

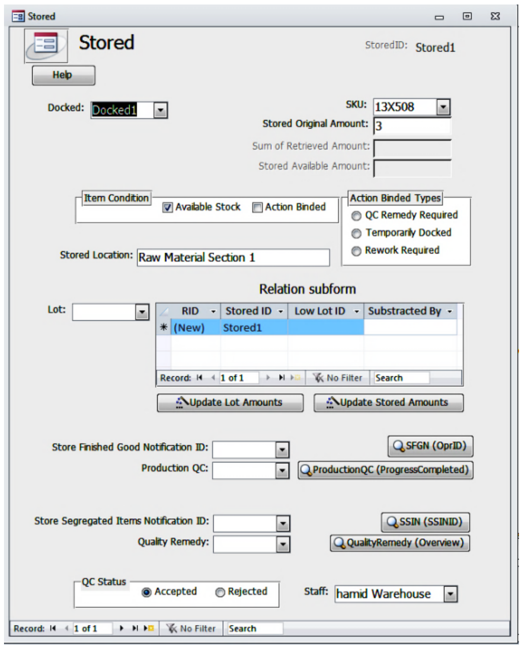
Table 3: Primary and Foreign keys at Inventory System Entity Classes

Entity Class (Name)	*Unique identifi (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**

4. FUNCTIONALITY EVALUATION

Using the resulting data models and conceptualizing designed database blueprint of logical model, the database prototype is fabricated. Tables including fields with unique primary key as well as relationships and foreign keys are created. Designed prototype is finalized with creating notifications templates and other setups. Next, the functionality of the inventory system requirements is evaluated. First the model will be loaded with data according to a certain scenario of reception of purchased order.

The first scenario is for inventory reception process. Purchased items are docked and registered with incrementally counting pattern (e.g. docked1) and are associated with expected purchase order (e.g. pur2). Purchase Arrival Notification is sent respectively via email detailing supplierID, SKU, amount, etc., to QualityDept. and is registered automatically (e.g. PAN1). Later, docked1 is updated upon receiving purchase QC results (e.g. PurQCR1) which triggers storing process if QCStatus is "Accepted". Docked items are then stored at Warehouse into the assigned location and the operation is registered (e.g. stored1) and associated to docked1. In the case of QCStatus is "Rejected" and upon receiving Return Purchase Notification (e.g. RetPurN1) from SaleDept., purchased items are return back to supplier and the operation is registered (e.g. RP1). In this scenario, the purchased item stored at warehouse increases the inventory SKU and is considered as main important requirement to evaluate.



Stored

StoredID: Stored1

Docked: Docked1

SKU: 13X508
Stored Original Amount: 3
Sum of Retrieved Amount:
Stored Available Amount:

Item Condition: [✓ Available Stock] Action Binded

Action Binded Types: QC Remedy Required / Temporarily Docked / Rework Required

Stored Location: Raw Material Section 1

Relation subform

Lot:

RID	Stored ID	Low Lot ID	Substracted By
(New)	Stored1		

Record: 1 of 1 No Filter Search

Store Finished Good Notification ID:
Production QC:

Store Segregated Items Notification ID:
Quality Remedy:

QC Status: [✓ Accepted] Rejected

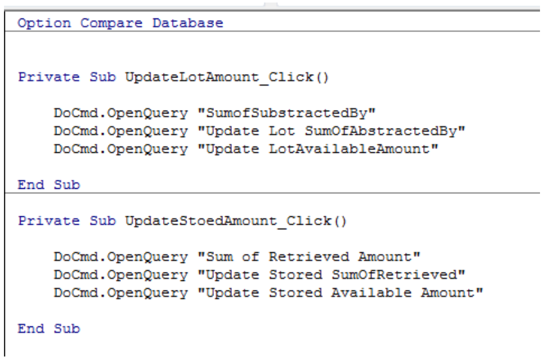
Staff: hamid Warehouse

Record: 1 of 1 No Filter Search

Figure 6: Stored entry form screenshot.

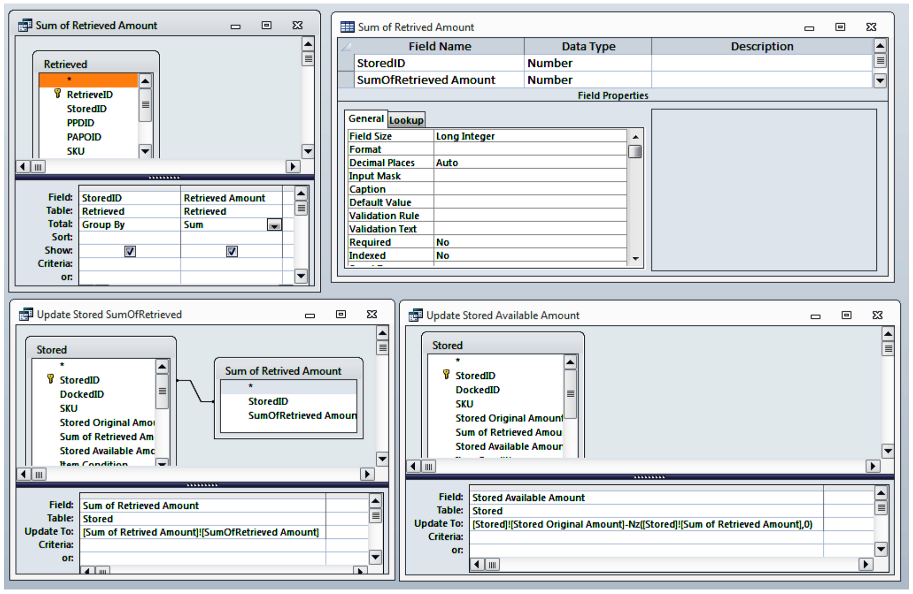
Figure 6 displays a screenshot of Stored Form. Stored Table is filled up in three different situations of storing docked items, storing finished goods instantiated by Production Dept., and at storing items after quality remedy operation at Quality Dept. for example, in such scenario of receiving finished goods along with Store Finished Goods Notification (e.g. SFGN1) via email from ProductionDept., items are stored at warehouse and same as before database is updated due to incrementing inventory SKU.

Another scenario is for inventory allocation and order assignment. Stored items are retrieved either for shipping to customer (i.e. inventory allocation) or to send to Shopfloor for production operations (i.e. order assignment). Hence, Retrieved Table is one shared Table which can be accessed at inventory system module for order assignment and at shipping system module for pick and pack, consolidation and labelling. Based on detailed production plan and BOM received from ProductionDept (e.g. PPD1), retrieved items are assigned to order and registered (e.g. Retrieve1) with detail and associated StoredID. Load balancing is made through automatic updating of stored amount. Figure 7 displays the stored updates procedure code at inventory SKU change in database. Figure 8 displays a screenshot of combination of three update queries to ad hoc update of inventory SKUs.



```
Option Compare Database

Private Sub UpdateLotAmount_Click()

    DoCmd.OpenQuery "SumofSubstractedBy"
    DoCmd.OpenQuery "Update Lot SumOfAbstractedBy"
    DoCmd.OpenQuery "Update LotAvailableAmount"

End Sub

Private Sub UpdateStoedAmount_Click()

    DoCmd.OpenQuery "Sum of Retrieved Amount"
    DoCmd.OpenQuery "Update Stored SumOfRetrieved"
    DoCmd.OpenQuery "Update Stored Available Amount"

End Sub
```

Figure 7: Screenshot of Stored and Lot update codes.



Sum of Retrieved Amount

Retrieved: *, RetrieveID, StoredID, PPDID, PAPOID, SKU

Field:	StoredID	Retrieved Amount
Table:	Retrieved	Retrieved
Total:	Group By	Sum
Sort:		
Show:	✓	✓
Criteria:		
or:		

Sum of Retrived Amount

Field Name	Data Type	Description
StoredID	Number	
SumOfRetrieved Amount	Number	

Field Properties

General Lookup

Field Size	Long Integer
Format	
Decimal Places	Auto
Input Mask	
Caption	
Default Value	
Validation Rule	
Validation Text	
Required	No
Indexed	No

Update Stored SumOfRetrieved

Stored: *, StoredID, DockedID, SKU, Stored Original Amou, Sum of Retrieved Am, Stored Available Amc, Item Condition

Sum of Retrived Amount: *, StoredID, SumOfRetrieved Amoun

Field:	Sum of Retrieved Amount
Table:	Stored
Update To:	[Sum of Retrived Amount]![SumOfRetrieved Amount]
Criteria:	
or:	

Update Stored Available Amount

Stored: *, StoredID, DockedID, SKU, Stored Original Amount, Sum of Retrieved Amou, Stored Available Amour

Field:	Stored Available Amount
Table:	Stored
Update To:	[Stored]![Stored Original Amount]-Nz([Stored]![Sum of Retrieved Amount],0)
Criteria:	
or:	

Figure 8: Screenshot of Designed Data Queries and Coding at Ad Hoc Stored Update.

To enable traceability of lots back to initial lot as well as lot consumptions (i.e. splitting or pooling), unique identification (e.g. R1) is required. This registration for relation between lots is either activated by production, quality or authorized initially by inventory. The existence termination of a given lot by storing process is referred at Relation Table. Inventory system is responsible for registration of initial instantiation and termination of any lot. Therefore, inventory amounts in lots considered as another requirement for model evaluation.

Next, validity of system functionality of the data-loaded designed database is achieved by implementing data queries. Queries are fundamental means used to obtain data in a predefined format from records in Table(s). An overview of the designed data queries to respond to lookups for the prototype application is presented at Table 4.

Based on implemented queries general requirements of inventory system highlighted at literature findings are addressed or explanatory discussed of how they are supposed to be covered. Perpetual inventory SKU updates is considered throughout the model and addressed in database application automatically or as an ad hoc application to support the real-time inventory lookups.

Table 4: Implemented Lookups and Data Queries Description for Functionally Tests

Lookup Name	***Data Query Display Setting***	***Functionality***

Inventory (All SKU @ Lots)	All inventory SKU within all lots.
	SELECT Lot.SKU, Sum(Lot.[Lot Original Amount]) AS [SumOfLot Original Amount], Sum(Lot.[Sum of Substracted By]) AS [SumOfSum of Substracted By], Sum(Lot.[Lot Available Amount]) AS [SumOfLot Available Amount]
	FROM Lot
	GROUP BY Lot.SKU;
		Traceability and real-time inventory

Inventory (All SKU @ Warehouse)	All inventory SKU at the Warehouse.
	SELECT Stored.SKU, Sum(Stored.[Stored Original Amount]) AS [SumOfStored Original Amount], Sum(Stored.[Sum of Retrieved Amount]) AS [SumOfSum of Retrieved Amount], Sum(Stored.[Stored Available Amount]) AS [SumOfStored Available Amount]
	FROM Stored
	GROUP BY Stored.SKU;
		Real-time inventory

Inventory (SKU @ Warehouse)	Availability of a SKU inventory stored at the Warehouse.
	SELECT Stored.SKU, Stored.StoredID, Stored.[Stored Original Amount], Stored.[Sum of Retrieved Amount], Stored.[Stored Available Amount], Stored.[Item Condition], Stored.[Action Binded Condition], Stored.[Stored Location], Stored.[Stored Time], Stored.[Stored Date], Stored.[QC Status]
	FROM Stored
	GROUP BY Stored.SKU, Stored.StoredID, Stored.[Stored Original Amount], Stored.[Sum of Retrieved Amount], Stored.[Stored Available Amount], Stored.[Item Condition], Stored.[Action Binded Condition], Stored.[Stored Location], Stored.[Stored Time], Stored.[Stored Date], Stored.[QC Status]
	HAVING (((Stored.SKU)=[Which SKU?]));
		Real-time inventory

Inventory (Total SKU @ Warehouse & Lots)	Available inventory everywhere.
	SELECT [InventoryLookup (All SKUs Amounts @ Lots)].SKU, [InventoryLookup (All SKUs Amounts @ Lots)].[SumOfLot Available Amount], [InventoryLookup (All SKUs Amounts@Warehouse)].[SumOfStored Available Amount], [SumOfLot Available Amount]+[SumOfStored Available Amount] AS Total
	FROM [InventoryLookup (All SKUs Amounts @ Lots)] INNER JOIN [InventoryLookup (All SKUs Amounts@Warehouse)] ON [InventoryLookup (All SKUs Amounts @ Lots)].SKU = [InventoryLookup (All SKUs Amounts@Warehouse)].SKU
	GROUP BY [InventoryLookup (All SKUs Amounts @ Lots)].SKU, [InventoryLookup (All SKUs Amounts @ Lots)].[SumOfLot Available Amount], [InventoryLookup (All SKUs Amounts@Warehouse)].[SumOfStored Available Amount], [SumOfLot Available Amount]+[SumOfStored Available Amount];
		Real-time inventory

IssuedByInventory (Overview)	All lots which are originally generated by the Inventory for production.
	SELECT Issued.IssuedID, Issued.Issuer, Issued.RetrieveID, Lot.LotID, Lot.SKU, Lot.[Lot Original Amount], Lot.[Sum of Substracted By], Lot.[Lot Available Amount]
	FROM Issued INNER JOIN Lot ON Issued.IssuedID=Lot.IssuedID
	GROUP BY Issued.IssuedID, Issued.Issuer, Issued.RetrieveID, Lot.LotID, Lot.SKU, Lot.[Lot Original Amount], Lot.[Sum of Substracted By], Lot.[Lot Available Amount]
	HAVING (((Issued.Issuer)=1));
		Traceability and order assignment

Stored (Overview) / (StoredID)	All records at the Stored Table or one particular record with StoredID criteria.
	SELECT Stored.*
	FROM Stored;
	or
	WHERE (((Stored.StoredID)=[Which StoredID?]));
		Traceability. Reception concept in inventory and Item type requisitions

StoredAndRetrieved (Overview)	All retrieved records associated to one stored for all stored records.
	SELECT Stored.*, Retrieved.*
	FROM Stored INNER JOIN Retrieved ON Stored.StoredID = Retrieved.StoredID;
		Order assignment and inventory allocation

Docked (Overview)/ (DockedID)	All records at the Docked Table or one particular record with DockedID criteria.
	SELECT Docked.*
	FROM Docked;
	Or
	WHERE (((Docked.DockedID)=[Which DockedID?]));
		Traceability of the receptions

Traceability, order assignment and inventory allocation requirement are well-addressed through many distinctive queries. Lot linking and traceability as well as reception requirements are emphasized by distinct Entity classes and eventually Tables in system design. Moreover, through registering the actual inventories, the designed system acts as supporting tool for production planning and inventory management for efficient decision makings. Figure 9 depicts the whole relationships between Tables in inventory system and the inventory system switchboard is illustrated as one screenshot at Figure 10.

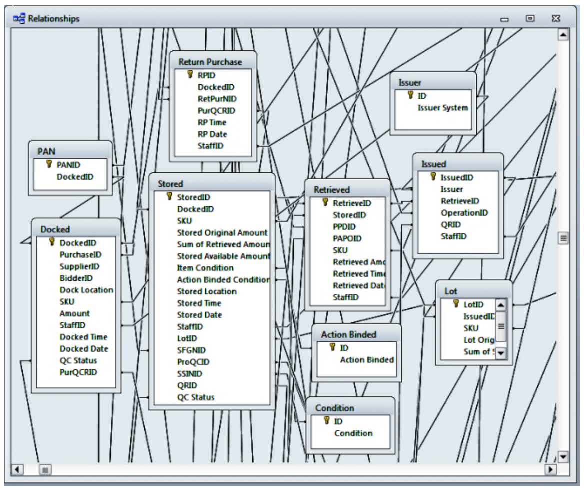


Figure 9: Relationships of inventory module Tables.

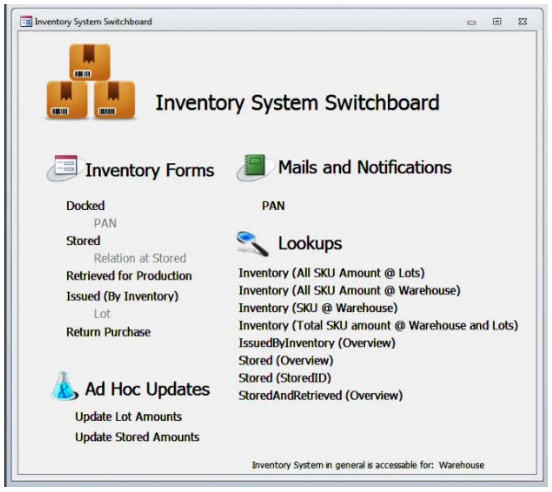

Inventory System Switchboard
Inventory System Switchboard
Inventory Forms
Docked
PAN
Stored
Relation at Stored
Retrieved for Production
Issued (By Inventory)
Lot
Return Purchase
Mails and Notifications
PAN
Lookups
Inventory (All SKU Amount @ Lots)
Inventory (All SKU Amount @ Warehouse)
Inventory (SKU @ Warehouse)
Inventory (Total SKU amount @ Warehouse and Lots)
IssuedByInventory (Overview)
Stored (Overview)
Stored (StoredID)
StoredAndRetrieved (Overview)
Ad Hoc Updates
Update Lot Amounts
Update Stored Amounts
Inventory System in general is accessable for: Warehouse

Figure 10: Screenshot of the inventory system switchboard

5. CONCLUSION

This paper focused on the development of perpetual production logistics inventory system modelling for in a modular-basis at its highest level requirements. Through explanatory modelling technique we address how the business process and data modelling of inventory system can be conducted using BPMN and UML as the state-of-the-art object oriented methodology. Research on literature for inventory system requirements are highlighted and carefully addressed. The resulting models are used for the prototype implementation of computerized inventory information system able to integrate with other back office system to receive and send messages and notifications as well as running queries.

The real-time functionality of integrated information systems improves the flow of information which was evaluated associated with all inventory system requirements. The query system manipulates data dynamically to keep the information up-to-date and guarantees a reliable report generation system for different type of inventory-related requisitions like inventory allocation, receptions, and order-assignments. The model is able to associate all inventory schema management and traceability data generated from purchased orders to finished products at warehouse and to issuance for the shipping. The perpetual designed system is able to reflect the actual physical inventory through ad hoc inventory SKUs updates. It supports variety of real-time control lookups in all incrementing and decrementing inventory SKUs stored at Warehouse and in lots at Shopfloor including item types, conditions, and rates triggered by other integrated modules or by common inventory system console. The provided capability of integration in the proposed inventory models and database design improves the availability of different data analysis, ease of information access, and effective presentation interfaces.

Through enhanced visibility and dynamic store/retrieval of data, all inventory data applications are well responded. The designed model is limited to inventory management system for a single company environment with basically batch production system for make-to-order. Besides, to keep the referentiality and purpose of the model replenishment policies requirements such as defining safety stock has seen out of scope of the paper 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 data model 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. The model takes advantage of web-based environment using email services, it is also expected such modular and component-based model expands its applicability through considering the loosely coupling concept.

REFERENCES:

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

[2] Sana, S. S., 2012. A collaborating inventory model in a supply chain. *Economic Modelling*, 29: 2016–2023.

[3] Vries, J. d., 2007. Diagnosing inventory management systems: An empirical evaluation of a conceptual approach. *International Journal of Production Economics*, 108: 63–73.

[4] Bonney, M. C., 1994. Trends in inventory management. *International Journal of Production Economics*, 25: 107-114.

[5] Taei-zadeh, A., M. Mukhtar, S. Sahran and M. R. Khabbazi, 2012. A Systematic Input Selection for Service Identificaiton in SMEs. *Journal of Applied Sciences*, 12(2): 1232-1244.

[6] Ruiz-Torres, A. J. and F. Mahmoodi, 2010. Safety stock determination based on

parametric lead time and demand information. *International Journal of Production Research*, 48(10): 2841–2857.

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

[8] Kang, Y. and S. B. Gershwin, 2005. Information inaccuracy in inventory systems: stock loss and stockout. *IIE Transactions*, 37(9): 843-859.

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

[10] Shapiro, J. F. and S. N. Wagner, 2009. Strategic Inventory Optimization. *Journal of Business Logistics* 30(2): 161-173.

[11] Boute, R. N., S. M. Disney, M. R. Lambrecht and 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.

[12] Dean, P. R., Y. L. Tu and D. Xue, 2007. An information system for one-of-a-kind production. *International Journal of Production Research*, 47(4): 1071-1087.

[13] Khabbazi, M. R., M. Y. Ismail, N. Ismail, S. A. Mousavi and H. S. Mirsanei, 2011. Lot-base traceability requirements and functionality evaluation for small- to medium-sized enterprises. *International Journal of Production Research*, 49(3): 731-746.

[14] Heizer, J. and B. Render, 2004. Principles of operations management.

[15] Bhattacharya, A., B. Sarkar and S. K. Mukherjee, 2007. Distance-based consensus method for ABC analysis. *International Journal of Production Research*, 45(15): 3405–3420.

[16] Rezaei, J. and S. Dowlatshahi, 2010. A rule-based multi-criteria approach to inventory classification. *International Journal of Production Research*, 48(23): 7107–7126.

[17] Ruiz, N., A. Giret, V. Botti and V. Feria, 2011. Agent-supported simulation environment for intelligent manufacturing and warehouse management systems. *International Journal of Production Research*, 49(5): 1469–1482.

[18] Yeh, C.-H., 1994. Production data modelling: an integrated approach. *International Journal of Operations & Production Management*, 15(8): 52-62.

[19] Bakker, M., J. Riezebos and R. H. Teunter, 2012. Review of inventory systems with deterioration since 2001. *European Journal of Operational Research*, 221: 275–284.

[20] Chung, Y. T. and F. Erhun, 2013. Designing supply contracts for perishable goods with two periods of shelf life. *IIE Transactions*, 45(1): 53-67.

[21] Saad, N. and V. Kadirkamanathan, 2006. A DES approach for the contextual load modelling of supply chain system for instability analysis. *Simulation Modelling Practice and Theory*, 14: 541–563.

[22] Ruifeng, C. and V. Subramaniam, 2011. Performance evaluation for tandem multi-factory supply chains: an approximate solution. *International Journal of Production Research*, 49(11): 3285–3305.

[23] Töyrylä, I., 1999. Realizing the potential of traceability: a case study research on usage and impacts of product traceability, PhD Thesis, Industrial Engineering and Management, Helsinki University of Technology, Finland.

[24] Khabbazi, M. R., M. K. Hasan, R. Sulaiman and S. A. Mousavi, 2010. Extending quality data for lot-based traceability system in SME. *Information Technology (ITSim), 2010 International Symposium in*. Kuala Lumpur, Malaysia, IEEE. pp. 1158-1163