A Three-tier Software Architecture for Manufacturing Activity Control in ERP Concept

Stefan Dumbrava, Doru Panescu and Mihaela Costin

Abstract: *Markets globalisation makes industrial companies to change radically, becoming more flexible, customer oriented and agile in manufacturing. Consequently, the enterprise software applications are becoming more and more complex, exploiting distributed object technology in multi-tier architectures. The aim of the paper is to present the design specification for an enterprise application that integrates the ERP concept in three-tier architecture, containing modularized, distributed subsystems in configurable and maintainable software. Starting from the customer order, to planned order dispatch, the ERP main components and the information flow within the system is presented. Then, the software architecture is discussed, where the ERP components are implemented using Enterprise Java Beans (EJB), deployed in a J2EE application server, for performing transactions with business environment and the database. Constructing the visual model in Unified Modeling Language (UML), the application is designed and analyzed. The use case diagram of the system is presented.*

Key words: *Enterprise Resource Planning, Material requirements Planning, Distributed object technology, Unified Modeling Language.*

INTRODUCTION

> *"Whereas at one time the decisive factor of production was the land, and later capital…..today the decisive factor is increasingly man himself, that is his knowledge".*
> Pope John Paul II - 1991 Centesimus Annus

Due to increasing inter-organizational production and control, a new management dimension occurs that focuses on both manufacturing activities and the supply chain. Its demands are the cost control, needs to analyze costs/revenues on a product or customer basis, flexibility to respond to changing business requirements, more informed management decision making and changes in the way of doing business, [5], [10]. The enterprise information systems and the business software applications have continuously followed the new managerial concepts, evolving from systems without decision support towards complex applications, including assistance in the decision taken process. ERP system is an overall company concept that has capabilities for quality management, field service, maintenance management, distribution, marketing, and supplier management. Software solutions implementing this concept may address the enterprise needs, making the process view of an enterprise to meet the organizational goals by integrating all functions of an enterprise. In the same time, it may focus on the external supply chain enabling the firm to deal directly with suppliers and to assess the availability of their resources. Consequently, with the support of software technology, ERP becomes a distributed application in an Internet environment, developed and able to communicate directly with suppliers and customers, sharing e-markets in an e-business environment, [6]. For these purposes enterprise applications typically require the ability to handle a large number of users and large amount of data, flexibility to expand quickly, security services in order to prevent unauthorized access, complex transaction processing and database access, [7]. The advances in computer communication and databases together with the emergence of new software technologies, based on scalable, interoperable and reconfigurable information architectures enabled business applications to be designed, build and implemented more rapidly, cheaper and with fewer resources. The multi-tiered application model for enterprise application is a client-server architecture in which the user interfaces, functional process logic, data storage, meets these requirements, [1]. The three-tier architecture is distributed in three locations: client machine, server machine and the database machine, the architecture extending the two-tier client/server applications to three-tier applications.

THE ERP SYSTEM ARCHITECTURE

The basic blocks and the information flow within the system are represented in Fig. 1.

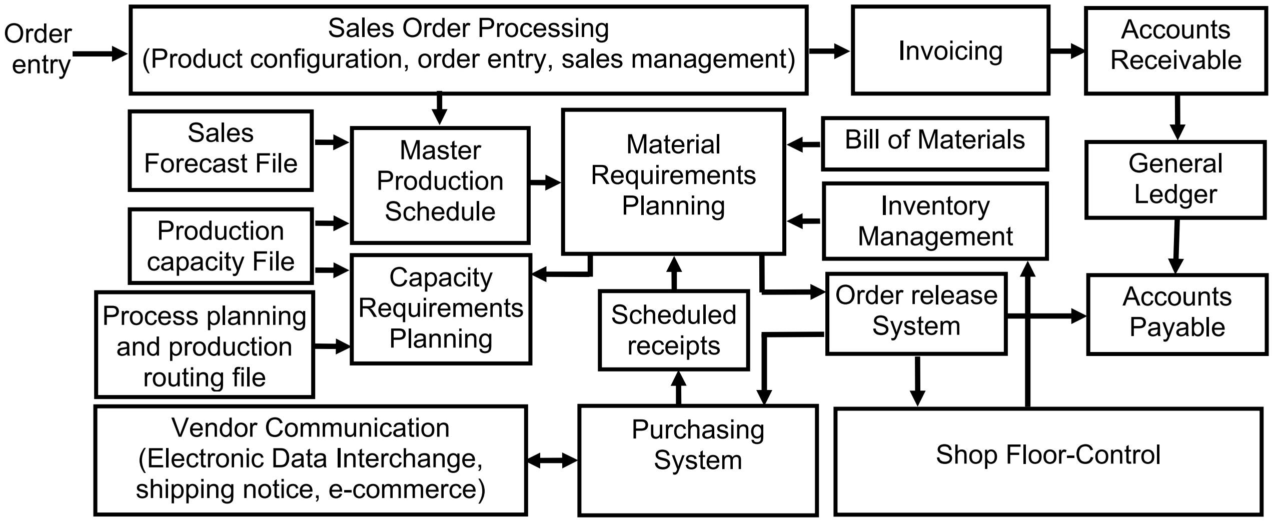
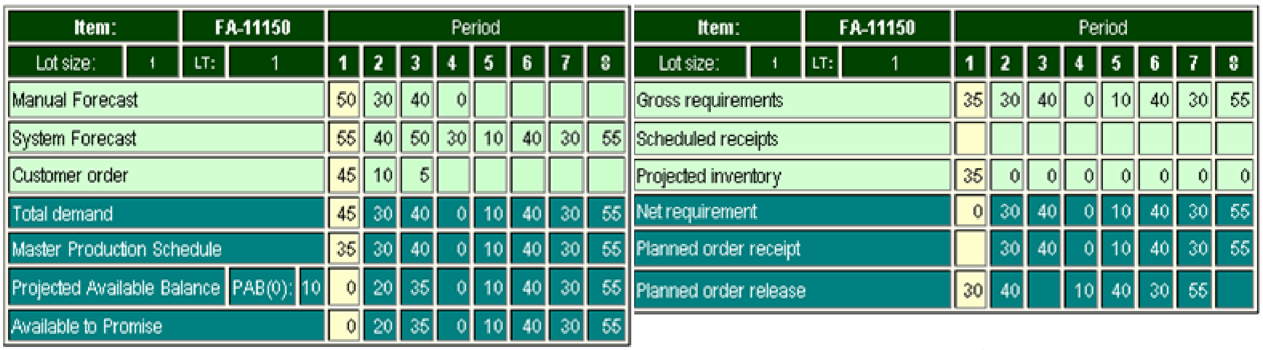


Fig. 1 Information flow in an ERP architecture

The main goal of the architecture in Fig. 1 is to disaggregate the production plane, to schedule and issue production orders for the final products together all their components, while ensuring the feasibility of production plans and the general ledger, [6]. The input in the system is the customer order that after it is processed in the sales and marketing department enters the Master Production schedule (MPS). The MPS is a modelling technique that allows for demand-driven production plans, making statements on what, when and how much is to be produced. Based on data input (Manual forecast, system forecast and actual customer order) the system generates the total demand for each time periods, using the following rule: for the lead-time (emphasized in yellow) the actual customer order is considered; outside, the manual forecast has priority to system forecast, [5]. The initial inventory stock is also correlated to generate the MPS records: the projected available balance and the available to promise quantities. The MRP modelling technique uses time-phased gross requirements, scheduled receipts, inventory information and other system parameters, for elaborating the net requirements, planned order receipts and planned order release schedule, Fig. 2.



Item:		FA-11150		Period							
Lot size:	1	LT:	1	1	2	3	4	5	6	7	8
Manual Forecast				50	30	40	0				
System Forecast				55	40	50	30	10	40	30	55
Customer order				45	10	5					
Total demand				45	30	40	0	10	40	30	55
Master Production Schedule				35	30	40	0	10	40	30	55
Projected Available Balance		PAB(0):	10	0	20	35	0	10	40	30	55
Available to Promise				0	20	35	0	10	40	30	55

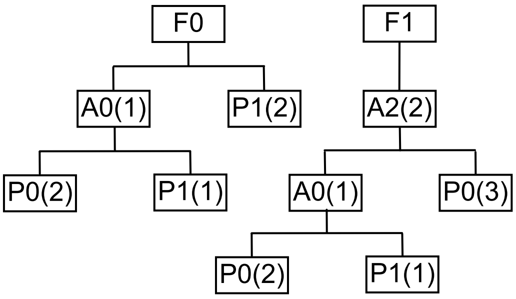
a)

Item:		FA-11150		Period							
Lot size:	1	LT:	1	1	2	3	4	5	6	7	8
Gross requirements				35	30	40	0	10	40	30	55
Scheduled receipts											
Projected inventory				35	0	0	0	0	0	0	0
Net requirement				0	30	40	0	10	40	30	55
Planned order receipt					30	40	0	10	40	30	55
Planned order release				30	40		10	40	30	55	

b)

Fig. 2 MPS (a) and MRP (b) matrices for a generic item with Lot for Lot Sizing Technique

Gross requirement is total actual demand for that item at that time. Scheduled receipt indicates the expected units that will be completed at that time. Planned order release indicates the suggested amount of units to order from the Shop-floor system. With the knowledge of time-phased gross requirements, scheduled receipts and other MPS parameters, MRP can be filled out. In order to adapt to suppliers conditions, several techniques for lot sizing are used. Explosion of requirements in MRP programs flows down through component levels following the linkage specified in the Bill of Materials (BOM) structure. The planned order release of a parent item multiplied by the quantity of the child item that goes into the making of one parent item, become the gross requirement of the component required at the next level. The information is organised in tree type structures, each tree being used to describe the structure of one final product. The BOM structure is exploded many times during MRP calculation, requiring fast data retrieval. The format used to structure its content and to represent the hierarchical structures is the Extensible Markup Language (XML). The attributes considered for each node are name, quantity, and lead-time. As a sample, the Fig. 3 shows hierarchy structure of BOM for a mixed type production of final products F0 and F1. Each of them consists of assemblies and parts.



```
<BOM>
       <product name="F0">
             <subassembly name="A0" quantity="2" lead_time="2">
                   <part name="P0" quantity="2" lead_time="1"></part>
                   <part name="P1" quantity="1" lead_time="1"></part>
             </subassembly>
             <part name="A1" quantity="2" lead_time="3"></part>
       </product>
       <product name="F1">
            <subassembly name="A2" quantity="2" lead_time="2">
                        <subassembly name="A0" quantity="1" lead_time="1">
                              <part name="P1" quantity="2" lead_time="1"></part>
                              <part name="P2" quantity="1" lead_time="1"></part>
                        </subassembly>
            <part name="P0" quantity="3" lead_time="1"></part>
            </subassembly>
       </product>
       </BOM>
```

Fig. 3 BOM Hierarchy Structure and the corresponding XML file

The file is plain text format and Java language is used to access the BOM file, parsed with Document Object Model (DOM) specification, no matter what platform it resides on.

THE THREE-TIER ARCHITECTURE OF THE ERP APPLICATION

The ERP application is designed as a web-based and server-side application that is partitioned in terms of application logic into three-tiers, Fig. 4. Each layer has a different responsibility in the overall deployment, and within each layer, there can be one or more components. The layer partitioning is as follows. Presentation Tier contains components dealing with user interfaces and user interaction. The presentation layer of the web-based deployment could use HTML pages, Java Server Pages, and/or Java Applets. Middle Tier is composed of the Web Tier – JSP and Business Tier. The Web-Tier is the web server part that includes the web container and other protocols by the J2EE specifications. In the web container, servlets, JSP pages, filters, and web event listeners execute and may respond to HTTP requests from web clients. They may also be used to generate XML or other format data that is consumed by other application components.

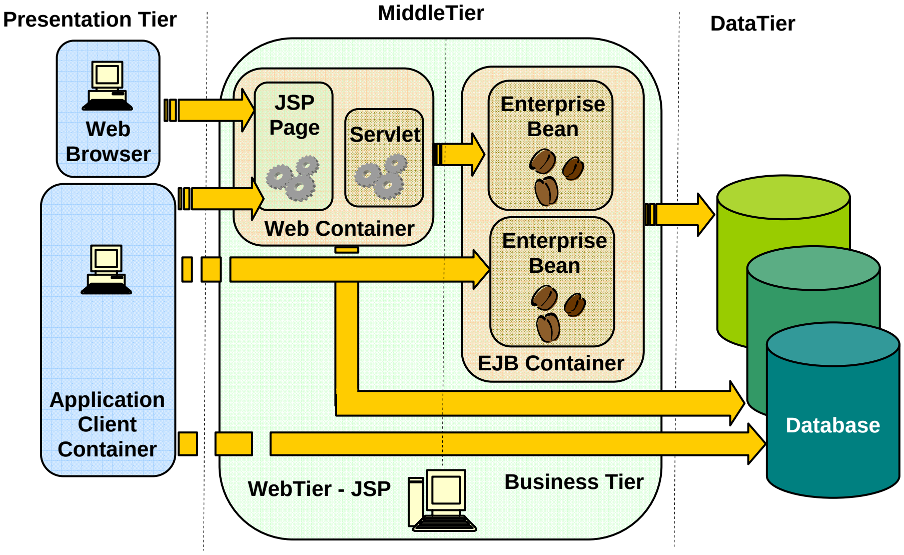


Fig. 4 The three-tier architecture of an enterprise application

The model-view-controller (MVC) paradigm was developed to map the input, processing, and output tasks to the graphical user interaction model. The presentation model, view, and controller exist in the Presentation services tier. The Model 2 architecture introduces a controller servlet between the browser and the JSP pages or servlet content being delivered. The controller centralizes the logic for dispatching requests to the next view based on the request URL, input parameters, and application state. The controller also handles view selection, which decouples JSP pages and servlets from one another. The Model 2 controller servlet provides a single point of control for security and logging, and often encapsulates incoming data into a form usable by the back-end MVC model. The Business Tier contains Enterprise Java Beans (EJB) components that encapsulate business logic into server side and work together to solve the business problems, [4], [8]. The ERP application is implemented as a set of business-logic-controlling EJB components that have been configured in application-specific ways inside an EJB container such as an application server. Clients are written to communicate with the EJB components and handle the results. EJB covers two fundamental models for synchronous communication in building enterprise applications: Session Beans– objects that represent a transient conversation with a client and Entity Beans - data in a database, along with the methods to act on that data. *Session Bean* can be Stateless (each call to the bean's methods is independent of the following call) or Stateful (a client has exclusive use of the bean's methods and maintains state between method calls). Another way to communicate with EJB components is using M*essage-Driven Beans* for sending receiving asynchronous messages The *EJB Container provider* is part of the infrastructure as the runtime system for one or multiple enterprise beans. It links different beans and EJB server. Data Tier is represented by the database system that storages the business data and state. It is accessible from web components, enterprise beans, and application client components through the JDBC API. The advantage to partition the application into these logical layers is to isolate each tier. Thus, it should be possible to change the Presentation Tier while minimizing impacts on the business logic in the Application Tier or Data Tier.

VISUAL MODELLING IN UNIFIED MODELING LANGUAGE (UML)

Visual modelling is a communication tool that uses standard graphical notations for specifying, visualizing, constructing and documenting the software and/or business models, independently of the implementation language, [2]. UML is such a platform-independent graphical standard used to define the enterprise architecture, the strategic reuse, system capabilities and enterprise application integration. The model of the system comprises several views that characterise it; *use case* and *class diagrams* (which define the functionality and the logical view of the system), *state chart* and *activity diagrams* (which define the behaviour of the system), *sequence* and *collaboration diagrams* (which define the interaction within the system), *component* and *deployment diagrams* (which define the implementation issues). It also has the following main capabilities: identify and design business objects and then map them to software components, partition services across a multi-tier model and/or architecture, design a distributed object architecture, code generation directly from the model, use reverse engineering to create models from existing components and applications, use round-trip engineering facilities in order to keep the designs synchronized with code, [3]. The conceptual design of the application is done using this visual modelling tool. The use case diagram for the ERP application is represented in Fig.5.

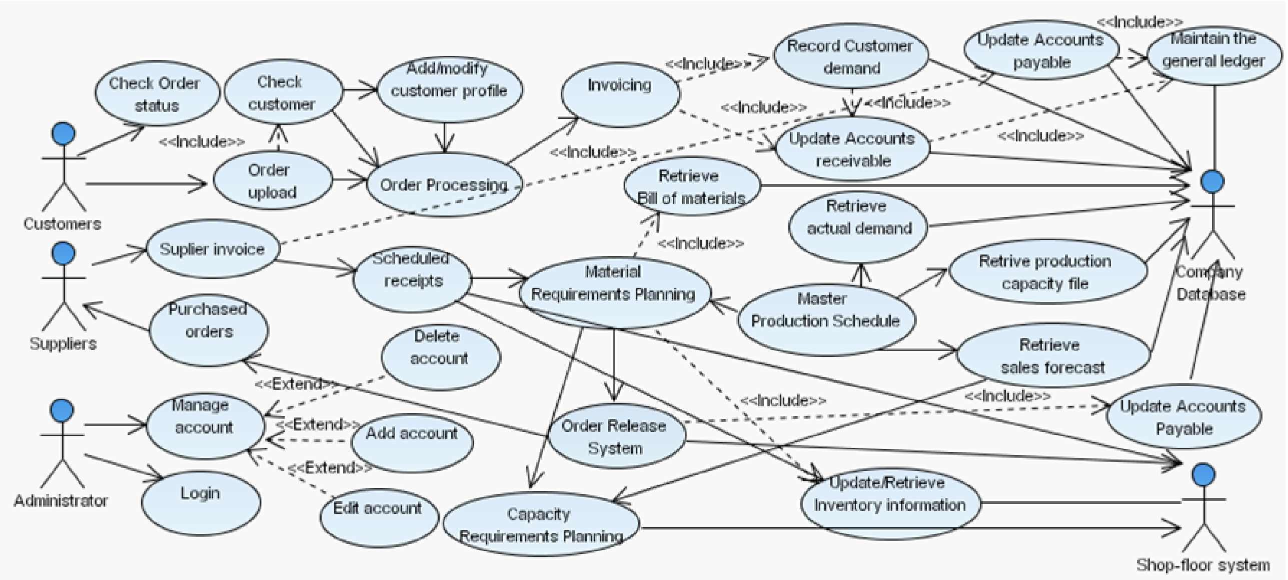


Fig. 5 The Use case diagram of the ERP application

According to the architecture, five kinds of group users will use the system: Customer, Administrator, Supplier, Shop-floor system, and the database. They are the actors in the model, categorized by means of generalization relationships. Use cases can be related by association and generalization relationships and by two stereotyped relationships called <<include>> and <<extend>>. Actors and use cases are linked by a special kind of association called <<communicate>> relationship. Use case for Administrator is to manage the accounts and to supervise the system. Checked and updated by the system, Customer group may upload orders that are processed before issuing the invoice. It includes the firm record of the demand and the update of the receivable account. The customer group may check the order status or modify the order. Based on actual demand and the sales forecast the MPS is generated, taking into account the available production capacity. The schedule for final products it is correlated with inventory information, scheduled receipts and the exploded BOM to generate the MRP. Using the make/buy decision associated to the part, the planned orders are directed to the Shop-floor system or to purchasing system by Order Release System. Supplier group may receive the purchase orders from MRP system by using web browser. Once the customers add or modify orders, MRP system may update data automatically to rapidly respond to change of customer requirement. MPS, MRP and Capacity Requirements Planning are designed as Session Beans that invokes other Entity Beans for logic performing. BOM, Customer Order, Process Plan and Inventory Data are designed as Entity Bean, representing persistent data. All these Enterprise Java Beans reside in Application Server and supply services like security, transaction management, and scalability. The complete implementation of the application using EJB comprises: writing remote and home interfaces, which define the methods of creating, finding and interacting with an EJB, writing the main EJB class, writing the deployment descriptor component and compiling the EJB classes and interfaces. The UML is extensively used through the entire procedure.

CONCLUSIONS AND FUTURE WORK

In this paper, the specifications for an enterprise application designed to integrate an ERP system using distributed object technology are presented. The research has been conducted within the framework of the project entitled "*The development of an Executive Information System Portal for Manufacturing Activity Control in the ERP Concept*", developed with the National Counsel for Scientific Research in the Higher Education. This is part of the first year activity, in which the objectives have been to design the architecture of the ERP application, the information flow and activity simulation. The architecture is designed in Enterprise Java Beans specifications, as a modern solution for developing multitier enterprise services. J2EE applications can be rapidly deployed and easily enhanced, ensuring a flexible reaction to the business environment feedback.

REFERENCES

[1] Blaha M., W.Premerlani. Object-Oriented Modeling and Design for Database Applications, Prentice Hall, 1997.

[2] Cheesman, J., J. Daniels. UML Components: A Simple Process for Specifying Component-Based Software, Addison-Wesley, 2000.

[3] Quatrani, T. Visual Modeling with Rational Rose and UML - Addison-Wesley, 1998.

[4] Monson-Haefel, R. Enterprise Java Beans, Sebastopol, CA: O'Reilly & Associates, Inc., 2000.

[5] Proud, J. P. Master Scheduling: A Practical Guide to Competitive Manufacturing, John Wiley & Sons Inc., 1999.

[6] Rehg, J. A., H. W. Kraeber. Computer-Integrated Manufacturing, 2nd edition, Prentice Hall, 2001.

[7] Senn, J.A. Analysis and Design of Information Systems, McGraw-Hill, 1989.

[8] Singh, I., B. Stearns, M. Johnson, and the Enterprise Team. Designing Enterprise Applications with the J2EETM Platform, 2nd edition, Addison-Wesley, 2002.

[9] Sipper, D., B. Bulfin. Production: planning, control, and integration - New York: The McGraw-Hill Companies, Inc., 1997.

[10] Vollman, T. E, W. L. Berry, D.C. Whybark. Manufacturing Planning and Control Systems, 4th edition, McGraw-Hill, 1999.

ABOUT THE AUTHORS

Assoc. Prof. Stefan Dumbrava, PhD and Prof. Doru Panescu, PhD, Department of Automatic Control and Industrial Informatics, Faculty of Automatic Control and Computer Engineering, "Gh. Asachi" Technical University of Iasi, Blvd. Mangeron 53 A, Iasi, 700500, Romania, Phone: +40 232 278683, E-mail: sdumbrav@ac.tuiasi.ro, dorup@ac.tuiasi.ro.

Scientific researcher Mihaela Costin, PhD, Computer Science Institute, Romanian Academy, Iasi Branch, Blvd. Carol I 22A, Iasi, 700505, Romania, Phone: +40 232 241708, E-mail: mihaelacostin@yahoo.com.