Successful ERP Operations: Process Integration Perspectives and an Agent-Based Support System

Kwangho Park

College of Business Administration, Hanyang University
1271 Sa-1-Dong, Ansan 425-791, South Korea
Tel: +82-31-400-5644, Fax: +82-31-400-5591, E-mail:oobepark@hanyang.ac.kr

bstract

ny ERP system pushes a company toward full process ntegration and solves the fragmentation of information. owever, the tight process integration can propagate and agnify mistakes made in one department into the other epartments in real time. Thus, it can be posited that a entral support system for the coordination can help ERP sers and administrators dig out problems, take care of edious validation and verification, and maintain process ntegration of ERP with great consistency. This paper roposes an agent-based ERP operations support system EOSS) that aims at achieving and maintaining process ntegration of ERP at the highest level possible. With EOSS, he process integrity is monitored, with anomalies revented as early as possible and repaired as precisely as ossible.

eywords: ERP Operations; Agent System; Decision upport System; Process Integration

ntroduction

RP (Enterprise Resource Planning) is recognized as the ost imperative system based on the information echnology (IT) infrastructure [27]. Recently, a range of echnologies, many of which emerged after ERP began to e implemented, can enhance the capability of the original ystem, for example, sell-side e-commerce, electronic rocurement, customer relationship management (CRM), upply chain management (SCM) and so on [14].

n general, in order for a company to fully gain the benefits f ERP, it must follow a four-phase framework defined by Markus and Tanis [19]: project adoption, project onfiguration (implementation), shakedown (testing), and ystem operation. The earlier phases of the ERP life cycle ave gained research attention, especially the adoption [5, 5 19] and implementation [3, 11, 12, 20]. Also, agreeing with an argument that the previous research on information ystem still hold for ERP-like new information technologies [16] one could see that most of the previous empirical and ield studies on IS success [4, 6, 7, 17, 21, 23, 24, 26, 28] re applied to the earlier phases of the ERP life cycle.

However, assuming that an ERP system is successfully adopted and implemented to the expected level for a company does not automatically guarantee the ultimate success of the ERP system because it must be operated as planned and thus can provide the integrated information in real-time with integrity. A recent empirical study on the relationship between completeness of each of the ERP development phases and its impact on system performance reports that the final phase, system operation, has the most positive impact on the system performance [22].

So far, relatively little research has been conducted on successful ERP operations, the final phase of the ERP life cycle. Recently, the MDBM (Model-Driven Business Management) environment [10] was proposed to effectively support system operations, maintenance, and development. The linguistic component in the MDBM gives users to alternative to traversing the business model with search facilities. Although the MDBM concept provides useful framework for supporting dynamic and adaptable system operation and maintenance, it does not focus on process integration aspect of ERP operations and has limited functionalities to support and train users, only concentrating on semantic lexicons and simple query interpretations.

Any ERP system pushes a company toward full process integration and solves the fragmentation of information [5]. As a result, it streamlines a company's data flow and provides management with direct access to a wealth of real-time operating information. For many companies, these benefits have translated into dramatic gains in productivity and speed. However, Bingi *et al.* [3] stated that the tight process integration can propagate mistakes made in one department into the other departments in real time. Also, the original mistake gets easily magnified as they flow through the associated process chain. Thus, companies must be aware of the potential risks of the errors and take proper steps, such as monitoring the transactions and taking immediate steps to correct the problems.

Process integration of ERP requires all departments of a company to cooperate and process data in harmony. The previous findings [1, 4] imply that an appropriate level of support for users great influences the success of ERP operations. However, the coordination tasks are increasingly complex and indeterminate in nature and place increasingly heavy burdens on communication and decision making of MIS department and end users [29].

Thus, it can be posited that a central support system for the coordination can help ERP users and administrators dig out problems, take care of tedious validation and verification, and maintain process integration of ERP with great consistency. This paper proposes the architecture of an ERP operations support system (EOSS) that aims at achieving and maintaining process integration of ERP at the highest level possible. With EOSS, the process integrity is monitored, with anomalies prevented as early as possible and repaired as precisely as possible.

Process Integration Perspectives of ERP Operations

The Process Integration Model

From the process integration point of view, ERP can be modeled with PROCESS CHAIN, PCLINK, FUNCTION, FLINK, TRANSACTION DATA classes. This process integration model represents the fundamental definitions to abstract the process integration: integration objects, relationships among them, and cardinalities of the relationships. The classes of the model are formally defined next.

Class Definition: ERP

ERP is an ordered triple (P, PL, g) where where P = a nonempty set of PROCESS CHAINs; PL = a set of PCLINKs; g = a function associating with each PCLINK *pl* an ordered pair (p_1, p_2) of PCLINKs where p_1 is the predecessor PROCESS CHAIN and p_2 is the successor PROCESS CHAIN of the *pl.*

Class Definition: PCLINK

A PCLINK is a binary relation on P and connects two PROCESS CHAINs. It is a specialized form of FLINK that is defined below. Two types of PCLINKs are defined.

(a) Unconstrained: The predecessor PROCESS CHAIN can proceed regardless of the status of the successor PROCESS CHAIN.

(b) Constrained: The predecessor PROCESS CHAIN can proceed only after the successor PROCESS CHAIN finishes completely.

Class Definition: PROCESS CHAIN

PROCESS CHAIN is a totally ordered triple (F, FL, h) where F = a nonempty set of FUNCTIONs; FL = a set of FLINKs; h = a function associating with each FLINK *fl* an ordered pair (f_1, f_2) of FUNCTIONs where f_1 is the predecessor FUNCTION and f_2 is the successor FUNCTION of the *fl*. Since PROCESS CHAIN is totally ordered, it has a start and a finish FUNCTION and, therfore, must be executed in a strictly sequential order from the start FUNCTION to the finish FUNCTION.

Class Definition: FUNCTION

A FUNCTION is symbolized f: $V \rightarrow T$ where V is the domain of input values and T is the codomain of TRANSACTION DATAs. It is a fundamental unit of ERP operations that creates a TRANSACTION DATA as a result of its execution and references its predecessor FUNCTION except start FUNCTIONs. Thus, given a set of FUNCTIONs, F, a binary relation on F is defined as FLINK that is defined below.

Class Definition: TRANSACTION DATA

A TRANSACTION DATA is a data that is created as a result of a FUNCTION execution. Given a set of TRANSACTION DATAs, T, a binary relation on T is defined as FLINK. In that sense, the relations on T and F, respectively, are isomorphic. In most cases, TRANSACTION DATA is defined as a "header-item" structure: a header record with multiple associated item records. Thus, a TRANSACTION DATA is created in a "stereoptype" sturcture: a header-items. However, for the simplicity, the structure is implictly embedded in the model.

Class Definition: FLINK

A FLINK defines a binary relation on F & T. That is, it connects two adjacent FUNCTIONs or TRANSACTION DATAs. Four basic types of relations are defined in terms of the cardinalities of relationships: sparate (one-to-one); consolidated (many-to-one); split (one-to-many); and split & consolidated (many-to-many).

The Process Instantiation Model

Having defined the process integration model, the paper explores the operational aspect of process integration of ERP and presents a process instantiation model. While the process integration model shows the fundamental definitions of the process integration, the process instantiation model abstracts actual instances of ERP operations by PROCESS CHAIN INSTANCE, TRANSACTION DATA HEADER INSTANCE, and TRANSACTION DATA ITEM INSTANCE classes. Next, the internal structure of TRANSACTION DATA is explicitly represented in the process instantiation model. It should be noted that FUNCTION INSTANCEs are ommitted to simplify the model. And also, we will use TRANSACTION DATA INSTANCE as the single term to comprehend TRANSACTION DATA HEADER INSTANCE and TRANSACTION DATA ITEM INSTANCE. The classes of the process instantiation model are defined next.

Class Definition: PROCESS CHAIN INSTANCE

A PROCESS CHAIN INSTANCE is an actual execution of the corresponding PROCESS CHAIN and comprises of a series of TRANSACTION DATA INSTANCEs.

Class Definition: TRANSACTION DATA HEADER INSTANCE

A TRANSACTION DATA HEADER INSTANCE is the general information about the transaction itself such as transaction number (primary key), transaction date, user, etc. Since a PROCESS CHAIN INSTANCE involves many FUNCTIONs to execute, it is assoicated with many TRANSACTION DATA (HEADER) INSTANCEs. Also, since a TRANSACTION DATA (HEADER) INSTANCE includes many TRANSACTION DATA ITEM INSTANCEs each of which was created by other PROCESS CHAIN INSTANCEs, there exists a many-to-many relationship between PROCESS CHAIN INSTANCE and TRANSACTION DATA (HEADER) INSTANCE classes.

Class Definition: TRANSACTION DATA ITEM INSTANCE

A TRANSACTION DATA ITEM INSTANCE belongs to a TRANSACTION DATA HEADER INSTANCE and, therefore, is created only when the TRANSACTION DATA HEADER INSTANCE is created. Normally, many item instances are created for a particular header instance.

A PROCESS CHAIN is instantiated when a user executes its start FUNCTION and wants to save the transaction. Actually, from the same start FUNCTION, many different PROCESS CHAINs can be defined. For example, various sales PROCESS CHAINs can be started from OrderEntry FUNCTION: Domestic Sales, Export Sales, Local L/C Sales, Refund, etc. Thus, the target PROCESS CHAIN for the instantiation is identified as a user enters data into a "hub" control (i.e., text box) of the start FUNCTION. The value of a hub control decides which PROCESS CHAIN INSTANCE should be instantiated from the program execution.

After a PROCESS CHAIN INSTANCE is created, a series of TRANSACTION DATA HEADER INSTANCEs and their dependent TRANSACTION DATA ITEM INSTANCEs are created by executing the associated FUNCTIONs. Those TRANSACTION DATA INSTANCEs are interrelated with each other sequentially as defined in the corresponding PROCESS CHAIN. The interdependency of TRANSACTION DATA INSTANCEs is transitive such that if B refers to A, C refers to B, then C refers to A through B. Thus, it is possible for a PROCESS CHAIN INSTANCE to track all transitive TRANSACTION DATA INSTANCEs associated with. The longer span of a PROCESS CHAIN results in the longer chain of TRANSACTION DATA INSTANCEs. Admitting that most TRANSACTION DATA INSTANCEs are unforgiving data that have little margin for error [31], one can see that effective support systems must be devised to maintain the integrity of this complex reference network of TRANSACTION DATA INSTANCEs because it is inheritantly prone to integrity loss by operational mistakes and errors.

For management purposes, it is defined that a PROCESS CHAIN INSTANCE becomes "complete" when its TRANSACTION DATA INSTANCE created by its first FUNCTION is "completely referenced" by the subsequent TRANSACTION DATA INSTANCEs. Here, "completely referenced" implies that the TRANSACTION DATA INSTANCE is referenced completely by any successor TRANSACTION DATA INSTANCESs so that no more direct and transitive references are possible. Thus, a PROCESS CHAIN INSTANCE is in one of two states: {incomplete, complete}.

With the FLINK types defined, four types of interdependencies of TRANSACTION DATA INSTANCEs can be defined: sparate (one-to-one); consolidated (many-to-one); split (one-to-many); and split & consolidated (many-to-many).

Use Cases

In practice, PROCESS CHAINs do not always proceed as defined. A PROCESS CHAIN specifies, in fact, a normal execution flow of its FUNCTIONs included. However, there are abnormal use cases as defined in Jacobson *et al.* [13]. For the executions of any PROCESS CHAIN, it must be anticipated that cancellations or corrections take place. Abnormal use cases represent such modifications of the whole or a part of previous executions. However, the cancellations or corrections cannot be performed in an isolated fashion due to the nature of process integration. The effect of cancellations or corrections must be completely propagated through the associated PROCESS CHAIN INSTANCEs. Improper executions for the abnormal use cases may result in violations of referential integrity constraints.

For example, price errors on purchase orders could mislead financial analysts by giving a distorted view of how much the company is spending on materials. Companies must be aware of the potential risks of the errors and take proper steps, such as monitoring the transactions and taking immediate steps to rectify the problems should they occur. They must also have a formal plan of action describing the steps to be taken if an error is detected. A proper means to communicate to all the parties who are victims of the errors as soon as the errors are detected is extremely important. Consider another example of a manufacturing company that implemented an ERP package. Suddenly experiencing a shortage of manufacturing materials, production managers noticed that it was due to incorrect bills of materials and made necessary adjustments. However, the company did not have any procedures to notify others the error. The domino effect of the errors had already affected other functional areas. It took almost eight weeks to clean up the incorrect bills of materials in the database.

Furthermore, many different users or user groups from different departments may execute the same PROCESS CHAIN. It implies that potential communication problems may occur during the execution. As a PROCESS CHAIN INSTANCE proceeds, there exist potential delays and errors due to hand-offs. Apparently, abnormal use cases make these problems worse.

Abnormal use cases can be classified into two types: deletion and update. A deletion must take place in a part of TRANSACTION DATA INSTANCEs of the corresponding normal PROCESS CHAIN INSTANCE and proceed to the opposite direction. Actually, the range of the whole deletion is defined dynamically, depending on the state of the corresponding PROCESS CHAIN INSTANCE and the

FUNCTION where the original deletion takes place. Sometimes, a different PROCESS CHAIN is defined for an abnormal use case rather than undoing the previous executions. For example, Refund PROCESS CHAIN is for canceling (deleting) a previous Sales PROCESS CHAIN INSTANCE, when the Sales PROCESS CHAIN INSTANCE is "complete".

Updates occur in many different results. A simple update can take place in a FUNCTION for a particular TRANSACTION DATA INSTANCE without involving other FUNCTIONs. A propagation update is more complicated such that all the TRANSACTION DATA INSTANCEs that are referencing the target TRANSACTION DATA INSTANCE must be updated, too. A deletion update is the case which the update can be accomplished only by deletions. There are two ways to carry out the deletion update. One way is to use a diversion FUNCTION that is an additional FUNCTION for a special update. The diversion FUNCTION is employed when the PROCESS CHAIN INSTANCE is "complete" so that deletions for the update must take place in the entire PROCESS CHAIN INSTANCE. Another way is to delete the first affected TRANSACTION DATA INSTANCEs from the propagated FUNCTION back to original update FUNCTION and then re-create TRANSACTION DATA INSTANCE from the original update FUNCTION forward to the propagated FUNCTION.

The definition of abnormal use cases are embedded in the process integration model. The FUNCTION class where an abnormal use case take place must know how to proceed and thus provide the information of the appropriate sequence (i.e., opposite direction to the normal use cases) of FUNCTION execution for either deletion or update. Also, with such definitions, the range of deletions and updates is identifed by the process instantiation model. For example, the findPropagationTrDataInstances() operation that can be defined in TRANSACTION DATA ITEM INSTANCE class generates a list of TRANSACTION DATA INSTANCEs that must be carefully edited in order to maintain the integrity of process integration.

Metrics and Anomalies of the Process Integration

Process integration of ERP involves a great deal of complexity. Without careful administration, various types of anomalies can occur in the ERP operations. In order to define the anomalies, define metrics to evaluate the status of the ERP operations.

The first category of the metrics is concerned with the fundamental complexity of the process integration (FCPI). In order to compute the FCPI, the following metrics are defined: the number of PROCESS CHAINs (NPC); the number of FUNCTIONs (NF); the number of TRANSACTION DATAs (NTD); the number of FLINKs (NFL) and the number of PCLINKs (NPCL) for each link type, respectively. These metrics are used firstly to compute the fundamental complexity of individual process chain (FCPC) and then all the FCPCs are added up to FCPI. A simple computation model of FCPI and FCPC can be formulated as equations (1) and (2), respectively:

$$\mathrm{FCPI} = \sum_{i=1}^{N} \mathrm{FCPC}_i \tag{1}$$

where N is the total number of NPCs

$$\mathrm{FCPC}_i = \sum_{j=1}^{4} (w_{ij} * (\mathrm{NFL}_{ij} + 2 * \mathrm{NPCL}_{ij})) \tag{2}$$

where if link type = separate, j = 1; if link type = consolidated, j = 2; if link type = split, j =3; if link type = consolidated & split, then j =4; $w_{i1} = 1$, $w_{i2} = w_{i3} = 2$; $w_{i4} = 4$

The level of FCPI of a company can be computed against the reference FCPI. For an ERP system, the maximum level of the FCPI can be computed assuming that all of the process chains are associated with the most complex link types. Thus, the formula for the FCPI level for a company P (FCPIL_P) can be designed as equation (3):

$$\mathrm{FCPIL}_p = \frac{\mathrm{FCPI}_p}{\mathrm{FCPI}_R} \tag{3}$$

where FCPI_p = FCPI of a company P, FCPI_R = the Maximum FCPI of the ERP system;

The second category of the metrics is concerned with the operational complexity of the process integration (OCPI) for a given period. In order to compute OCPI, the following metrics per each PROCESS CHAIN are defined: the number of complete PROCESS CHAIN INSTANCEs (NCPCI); the number of incomplete PROCESS CHAIN INSTANCEs (NIPCI); the average duration of complete PROCESS CHAIN INSTANCEs (ADPCI); the maximum duration of complete PROCESS CHAIN INSTANCEs (MDPCI); the number of abnormal PROCESS CHAIN INSTANCEs (NAPCI); the number of TRANSACTION DATA INSTANCEs (NTDI); the average number of TRANSACTION DATA INSTANCEs (ANTDI) per PROCESS CHAIN INSTANCE; the maximum number of TRANSACTION DATA INSTANCEs (MNTDI). The abnormal PROCESS CHAIN INSTANCE is defined for each PROCESS CHAIN as an instance which has not been completed within the normal cycle time. Such abnormality damages two different quality factors of ERP operations: cost and service. Abnormally delayed PROCESS CHAIN INSTANCEs can increase business operation costs such as inventory holding cost, labor cost, equipment utilization cost, etc. On the other hand, the quality of customer services also can be degraded when just-in-time actions are not taken appropriately.

The third category of the metrics is concerned with the work complexity of the process integration (WCPI) for a given period. The monthly WCPI represents a work load of a particular user (or group) who is in charge of the execution of FUNCTIONs assigned for each month. The WCPI is computed by the following metrics: the number of TRANSACTION DATA INSTANCEs of the predecessor FUNCTION that have been referenced completely (NTDIR) by each FUNCTION; the normal TRANSACTION DATA INSTANCE man day (NTDIMD);

the total TRANSACTION DATA INSTANCE man day (TTDIMD); the number of TRANSACTION DATA INSTANCEs of the predecessor FUNCTION that have not been referenced completely (NTDINR) by each FUNCTION. The NTDINR represents the backlog for the user (or group). Then, the monthly work load (MWL) of each user can be computed by equation (4):

$$MWL = \frac{AVG(TTDIMD)}{SMD} \qquad (4)$$

where SMD = Standard Man Days ina month (i.e., 22 days in June)

With metrices defined above, the paper defines the anomalies embedded in process integration of ERP operations. The first type of anomalies is the fundamental complexity anomaly (FCA) and can be measured by FCPI or FCPC. FCA represents the overall complexity of the underlying ERP system operation, and, thus, occurs simply because the process integration is complex itself. The higher FCPI implies the higher FCA. This is because ERP systems inherently produce mass amounts of TRANSACTION DATA INSTANCEs that are inter-related. The higher FCA cannot help producing more errors that break the integrity of process integration.

The second type of anomalies is the operational complexity anomaly (OCA). OCA represents abnormal states of ERP operations due to either negligent or intended executions of FUNCTIONs by users. Each anomaly of OCA can be computed by OCPI metrics defined above. For example, a PROCESS CHAIN INSTANCE may have been in partially complete state for a long time. Such symptom implies the longer process chain intervals that cause operators to forget completing partially executed process chains. If an order was released a long time ago while picking required inventory of items and the delivery has not yet taken place, then, the items have been put on hold for nothing, causing substantial inventory holding cost. Such unprocessed data may result in a reduction in productivity of the organization [2]. Thus, it should be pursued to reduce TNPCIN and shorten ADPCI and ANTDIPPC.

The third type of anomalies is the work complexity anomaly (WCA). WCA is a kind of bottleneck symptom that the work load of a particular user (or goup) is too high. If MWL is too high, then the quality of corresponding customer services must be low. There are three causes for this anomaly:

- Work overload: The work load of the user (or group) is too high.
- Under-skilled user: The user (or group) is not well trained to operate FUNCTIONs.
- Over-complex process chain: The process chain is too complex for the user (or group) to operate adequately.

Therefore, the work load must be adjusted according to the causes:

- Work force support: For work overload, work force must be added or adjusted.
- Training: For under-skilled user, well training sessions can cure the problem.
- Process chain adjustment: For over-complex process chain, if possible, there is significant potential for simplifying the process integration, reducing the complexity to a manageable level for the user (or group).

Architecture of EOSS

Support Tasks

Although ERP promised huge improvements in efficiency, for example, shorter intervals between orders and payments, lower back-office staff requirements, reduced inventory, and improved customer service [8, 9], installing ERP was traumatic. Following long and expensive implementations, some companies had difficulty identifying any measurable benefits due to the embedded anomalies. Thus, in order for companies to fully accomplish the benefits of ERP, effective support for ERP operations is required. Before discussing the architecture of EOSS, let's define the support tasks.

Firstly, there is a significant potential for simplifying the fundamental complexity of the process integration. The continuous splitting and combining of the process is the major cause of its complexity. If the causes can be simplified by some organizational efforts, then the complexity may be greatly reduced to a manageable level. The task to this simplification, however, is not easy. Partners involved in the target process must cooperate and endure some level of inconvenience. Nonetheless, simplifying the complexity of process integration is worthwhile to make efforts. Thus, the first support task is to reveal the FCPI of ERP operations by providing various FCPI metrics.

The next support task for ERP operations is to support users in various ways. For normal use cases, the system can guide executions of PROCESS CHAINs, informing what to do with a particular program(FUNCTION), what to do next after the current program, and what have been done for the PROCESS CHAIN associated with the program. Consultants and users tend to pay less attentions to abnormal use cases. However, the abnormal use cases are indeed major causes to break the integrity of process integration. It is similar for a novice driver to drive backward. Cancelling a previous execution for a FUNCTION is not easy in a highly integrated information system. Especially, when the PROCESS CHAIN associated with the FUNCTION has been executed deep down to the last FUNCTION and an execution of the start FUNCTION has to be cancelled, the cancellation should start at the end of the PROCESS CHAIN and continuously move backward to the start FUNCTION. Also, individual cancellation must be carried out carefully not to affect the other instances of the PROCESS CHAIN. In this regard, it is effective to help users execute without breaking the integrity, informing that where to start, what to do next, how the cancellation has proceeded so far, and what to delete or update exactly. Also, periodically, it is required to generate an process integration report that shows various OCPI metrics such as NPCIC, NPCIN, ADPCI, MDPCI, ANTDI, MNTDI, and NTDI. Especially, NIPCI and NAPCI metric reveals potential abnomality of ERP operations. With such supports, users can continually perform an interactive check to see if the ERP operations goes as intended. User inquiries can be answered on the basis of up-to-date information regarding the ERP operations.

The last support task for ERP operations is to handle WCA. In order to maintain a desirable level of service, bottleneck operations and users must be identified and their work load must be adjusted. If shipping is a bottleck FUNCTION, the delivery service may be provided poorly. If payment is a bottleleck FUNCTION, the account receivable may not be cashed in within a tolerable time. By providing WCA information, ERP administrators can take appropriate adjustments to solve the anomalies according to the causes.

System Architecture

In order to carry out the support tasks defined, agents are defined and employed. As defined in Russell and Norvig [25], our agents accept transaction data as percepts from the underlying database and generate appropriate advices either interactively or automatically in order to maintain the integrity of process integration. Since the taks are too burdensome, agents are practical alternative for ERP operators as well as users. With agents dispatched, the data record accuracy and integrity are maintained and users can operate the ERP system exactly as expected. As depicted in Figure 1, EOSS is equipped with 4 types of agents: Instance Agent (IA); Support Agent (SA); Monitoring Agent (MA); Report Agent (RA). Users may execute ERP programs without noticing our agents. The agents are running at the backgroud and support ERP operations by communicating with business logic components of ERP. Relevant messages and guides are provided through user interface of ERP. However, sometimes, users may communicate with agents interactively, requesting supports they want. The definitions and instances information of ERP operations are stored in operation database.

ERP Operation Database

The process integration model represents system requirements and business processes constructed as a part of the implementation project. The definitions in the model are the framework for ERP operations afterwards. Users are to obey the business rules and flows defined in the model. If there is a case that a user cannot digitally perform a particular business practice with the ERP system implemented, the model must be updated dynamically.

On the other hand, the process instantiation model defines how to create and use ERP operations instances. It represents how underlying transaction data are created and modified rather than the actual transaction data. When the instances are created, the process integration model is accessed because it knows about them: which instance should be created, how it should be related with others, etc.

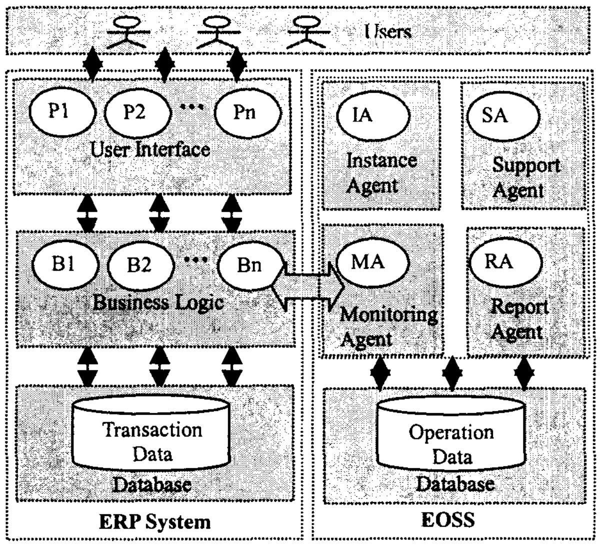

Figure 1- System architecture

The ERP operations database stores data about both models. It can be viewed as a projection of the transaction database reflecting the integrational aspect. Thus, as shown in Figure 2, the operation database contains definition data and instance data. Internally, the definition data designate how to create and relate the instance data.

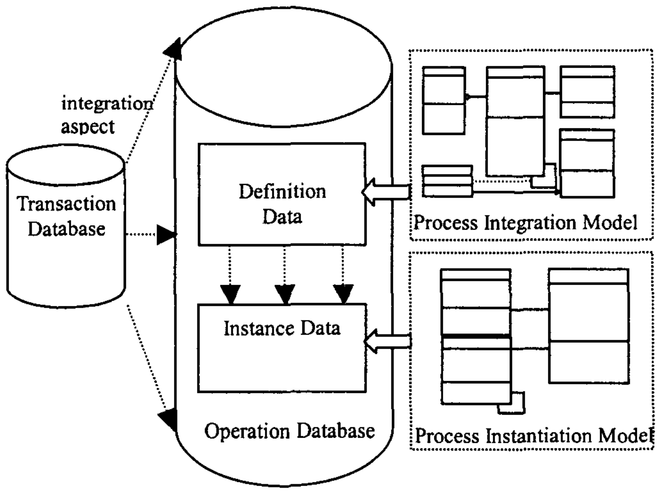

Figure 2 - Operation database architecture

Instance Agent

Basically, instance agents are responsible for creating, retrieving, updating, deleting the instance data in the operation database such as process chain instances and transaction data instances. Thus, two types of instance agents are designed: process chain instance agent (PCIA) and transaction data instance agent (TDIA) as shown in Figure 3. One of major roles of instance agents is to provide a snapshot of a particular process chain instance and its dependent transaction data instances.

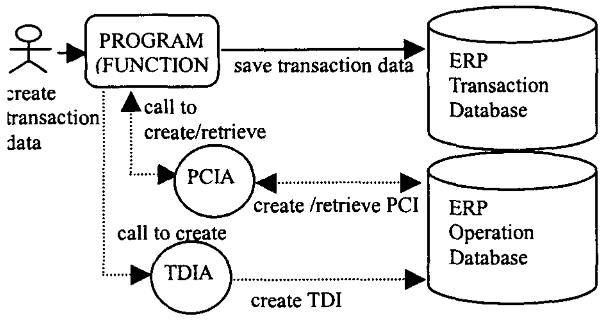

Figure 3- Architecture of instance agents

Support Agent

Support agents assist users to execute either normal use cases or abnormal use cases. They may be implemented as wizards that pop up during users' execution. Since an execution of a FUNCTION may be assocated with many PROCESS CHAIN INSTANCEs and TRANSACTION DATA INSTANCEs, support agents for the FUNCTION must get access to definition data and the instance data in the operaiton database.

Two types of support agents are defined: normal use case support Agent (NUSA) and abnormal use case support Agent (AUSA). As shown in Figure 4, when a user tries to create a new TRANSACTION DATA INSTANCE, a corresponding NUSA wizard is instantiated and assists the user executing the FUNCTION adequately. The NUSA wizard can play in an interactive mode. That is, the NUSA wizard displays information about the associated PROCES CHAIN such as how the particular PROCESS CHAIN has proceeded at the point, what FUNCTION follows the current FUNCTION, how the previous TRANSACTION DATA INSTANCEs are referenced, i.e., separate, split, consolidated, and split & consolidated. Doble-clicking on a FUCTION displayed in the PROCESS CHAIN screen, the user is taken directly to the ERP program for execution. Furthermore, it can send messages to the next user to notice the current transaction after the current FUNCTION execution finishes.

Likewise, an AUSA is instantiated and launched when users try to cancel or update existing transaction data instances. Since transaction data instances may be referenced by several other transaction data instances, such modifications are not easy. With the primary key of the transaction data instance to be deleted, the AUSA firstly analyzes the current reference status of the transaction data instance and find all transaction data instances that the deletion must propagate through. Then, it advises the sequence of deletions from the last transaction data instances backward to the intial transaction data instance to be deleted. Even it may show a simulation of such sequential deletions. Also, double-clicking a program in the wizard scrren, the user can be taken to the program where the deletion should start. When a process chain instance is "complete" so that the deletion must take place in the entire process chain instance, then the AUSA may advise to use a diversion program. The internal operations of AUSA are more complicated because it may have to show related transaction data instances as well as process chain instances.

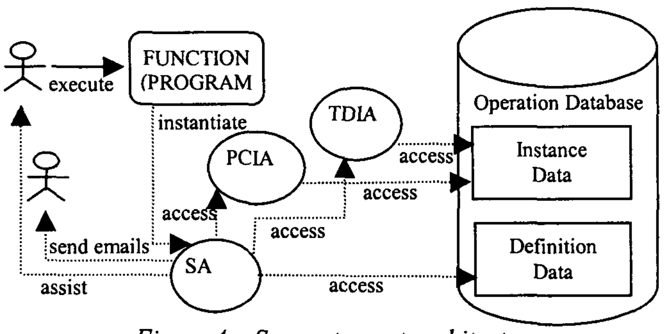

Figure 4 – Support agent architecture

Monitoring Agent

As shown in Figure 5, monintoring agents are continuously monitoring various process integration metrics and signaling anomalies when they are found. Especially, anomalies from abnormal PROCESS CHAIN INSTANCEs must get quick attentions from responsible users. For example, consignment goods that have not been invoiced for some time may damage the overall inventory turnover rate. This anomaly is found in most Sales PROCESS CHAINs. Stock shortage in production lines may decrease the productivity and machine utilization. This anomaly is detected when the WithdrawPart FUNCTION is delayed and thus the cycle time of work orders is getting longer in the Production PROCESS CHAIN.

Sometimes, anomalies may involve more than a single PROCESS CHAIN. Long receivable turnover period is found between a Sales PROCESS CHAIN and a Payment PROCESS CHAIN. If an invoice has not been cashed in for a long time, it may be a bad receivable. Disused article anomaly may involve several PROCESS CHAINs such as Sales, SalesPlanning, ProductionPlanning, and Procurement because the causes of this anomaly may be spread over these PROCESS CHAINs as follows:

- when an order for a make-to-order (MTO) product turns out to be placed with wrong specifications
- when a sales plan for a make-to-stock (MTS) product turns out to be over estimated
- when a production plan ordered to produce a product too many quantity
- when a purchase order request for a part is placed too many quantity

Since monitoring agents involve company-specific business rules as well as industry common business rules, they may have to be customized for each company. However, the following four typical monitoring agents may be required: consignment goods monitoring agent (CGMA); stock shortage monitoring agent (SSMA); bad receivable monitoring agent (BRMA); disused article monitoring agent (DAMA). For example, a DAMA provides a list of disused article and the causes of this anomaly.

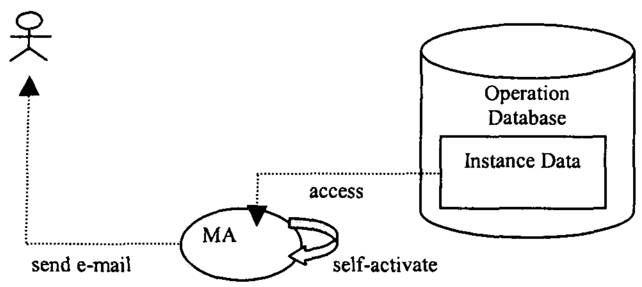

Figure 5 – MA architecture

Report Agent

As shown in Figure 6, periodic or real-time reports for ERP operations may help operators or users to make appropriate decisions in order to maintain the integrity of the process integration of ERP. There are three types of report agents: fundamental complexity report agent (FCRA), Operation compexity report agent (OCRA), and work complexity report agent (WCRA). FCRA shows the overall complexity of the ERP system under operation. By FCPIL, the relative complexity of the ERP system can be judged. OCRA shows the current operational complexity by showing various OCPI metrices. Abnormalities are identified such that long-delayed PROCESS CHAIN INSTANCEs and overly referenced TRANSACTION DATA INSTANCEs are listed on the OCPI panel. Finally, WCRA shows the work load of a particular user group and relevent balancing efforts may be made for overloaded user groups.

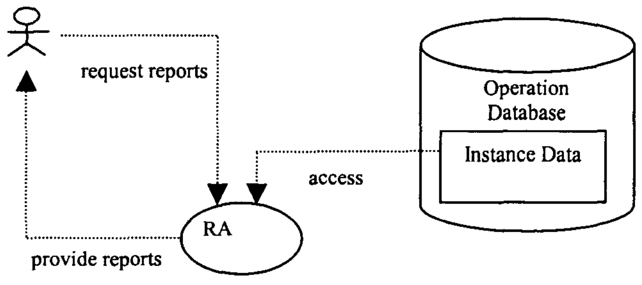

Figure 6 - *RA architecture*

Conclusions and Further Research

As companies aim at higher level of information system integration, successful ERP operations increasingly depends on the integrity of process integration. At the lowest level of process integration, a large amount of transaction data are intermingled with great complexity. Users are vulnerable to make mistakes, leaving many incomplete transaction data that have been intact for a while and may end up with financial losses. Complex cardinality between functions (programs) or transaction data threatens ERP operations out of control. Individual responses to such information overload include ommission of data and errors [30]. In this regard, the paper presents a support system for successful ERP operations. The process integration models presented helps ERP operators understand the fundamental nature of process integration. Based on these models, the architecture of EOSS is presented. Employing various agents, the system will help companies better operate ERP and continuously improve the status quo to the best business excellence.

Our agent-based support system can serve for multiple purposes. Especially, support agents may guide existing users to execute ERP programs without messing up transaction data. Also, new users can be trained so that they can take charge of ERP operations as quickly as possible. Operators and administrators may receive the most beneficial assistance because these agents prevent users' mistakes at an early stage and, therefore, minimizes damages to the integrity of process integration. Also, monitoring agents as well as report agents help users find anomalies as early as possible by generating alarms and various analysis reports.

The research limits the scope of process integration to only within a company, so called, internal process integration. It does not include external aspects of processes such as customers, distributors, suppliers, etc. However, admitting that up-to-date ERP systems must be expanded into the external information integration [18], EOSS must include the external aspects of process integration in further reseach. Also, the implementational issues must be attacked in order to make EOSS more practical in terms of performance, considering that it involves additional operation data and computations. Finally, the methods to compute complexties of the process integration and underlying metrics must be refined more accuately and objectively.

References

[1] Aiken, M., and Hage, J. (1968). "Organizational Interdependence and Intra-Organization Structure," *American Sociological Review*, Vol. 33, pp. 912-930.

[2] Ben-Arieh, D., and Pollatscheck, M. A. (2002). "Analysis of information flow in hierarchical organizations," *International Journal of Production Research*, Vol. 40, No. 15, pp. 3561-3573.

[3] Bingi, P., Sharma, M. K., and Golda, J. K. (1999). "Critical issues affecting an ERP implementation," *Information Systems Management*, Vol. 16, No. 3, pp. 7-14.

[4] Cheney, P., and Dickson, G. W. (1982). "Organizational characteristics and information systems: an exploratory investigation," *Academy of Managerial Journal*, Vol. 25, No. 1, pp. 170-184.

[5] Davenport, T. H. (1998). "Putting the enterprise into the enterprise system," *Harvard Business Review*, Vol. 76, Issue 4 (July-August), pp. 121-131.

[6] Delone, W. H. (1981). "Firm Size and the characteristics of computer use," *MIS Quarterly*, Vol. 5, No. 4, pp. 65-77.

[7] Ein-Dor, P., and Segev, E. (1978). "Organizational

context and the success of management information systems," *Management Science*, Vol. 24, No. 10, pp. 1067-1077.

[8] Foley, J. (1999). "ERP and e-business: perfect together?" *www.informationweek.com*, Sept. 13, p. 169.

[9] Gilbert, A., and Swear, J. (1999). "Reinventing ERP," *www.informationweek.com*, Sept. 13, pp. 15-20.

[10]Gulla, J. A., and Brasethvik, T. (2002). "A model-driven ERP environment with search facilities," *Data & Knowledge Engineering*, Vol. 42, pp. 327-341.

[11]Holland, C. P., and Light, B. (1999). "A critical success factors model for ERP implementation," *IEEE Software*, Vol. 16, Issue. 3(May/June), pp. 30-36.

[12]Hong, K., and Kim, Y. (2002). "The critical success factors for ERP implementation: an organizational fit perspective," *Information & Management*, Vol. 40, pp. 25-40.

[13]Jacobson, I., Christerson, M., and Josson, P., and Overgaard, G. (1992). *Object Oriented Software Engineering*. Reading, MA: Addison-Wesley.

[14]James, D., and Wolf, M. L. (2000). "A second wind for ERP," *The McKinsey Quarterly*, No. 2, pp. 100-107.

[15]Kumar, V., Maheshwari, B., and Kumar, U. (2002). "Enterprise resource planning systems adoption process: a survey of Canadian organizations," *International Journal of Production Research*, Vol. 40, pp. 509-523.

[16]Lee, A. (2000). "Researchable directions for ERP and other new information technologies," *MIS Quarterly*, Vol. 24, No. 1, pp. 3-8.

[17]Lehman, J. A. (1985). "Organizational size and information system sophistication," Working Paper 85-18, MIS Research Center, University of Minnesota.

[18]Li, C. (1999). "ERP packages: What's next?" *Information Systems Management*, Vol. 16, No. 3, pp. 31-35.

[19]Markus, M. L., and Tanis, C. (2000). The enterprise system experience: from adoption to success. In R. W. Zmud (ed.), *Framing the Domains of IT Management: Projecting the Future Through the Past.* Pinnaflex Educational Resources, Inc.

[20]Markus, M. L., Tanis, C., and Van Fenema, P. C. (2000). "Multisite ERP implementations," *Communications of the ACM*, Vol. 43, No. 4, pp. 42-46.

[21]Olson, M. H., and Chervany, N. L. (1980). "The relationship between organizational characteristics and the structure of the information services function," *MIS Quarterly*, Vol. 4, No. 2, pp. 57-68.

[22]Park, M., Lee, J., and Jeoung, S. (2002). "An explorative study of ERP system implementation: relationships between completeness of each phase and its impact on system performance," *Information Systems Review*, Vol. 4, No. 2, pp. 237-256 (In Korean).

[23]Raymond, L., (1985). "Organizational characteristics and MIS success in the context of small business," *MIS Quarterly*, Vol. 9, No. 1, pp. 37-52.

[24]Raymond, L., (1990). "Organizational context and information systems success: a contingency approach," *Journal of Management Information Systems*, Vol. 6, No. 4, pp. 5-20.

[25]Russell, S. J., and Norvig, P. (2002). *Artificial Intelligence: A Modern Approach*, (2nd ed.). Englewood Cliffs, NJ: Prentice Hall.

[26]Sanders, G. L., and Courtney, J. F. (1985). "A field study of organizational factors influencing DSS success," *MIS Quarterly*, Vol. 9, No. 1, pp. 77-93.

[27]Shar, R., Goldstein, S. M., and Ward P. T. (2002). "Aligning supply chain management characteristics and inter-organizational information system types: an exploratory study," *IEEE Transactions on Engineering Management*, Vol. 49, pp. 282-292.

[28]Srinvasan, A., and Kaiser, K. M. (1987). "Relationships between selected organizational factors and systems development," *Communications of ACM*, Vol. 30, No. 6, pp. 556-562.

[29]Thompson, J. (1967). *Organizations in Action.* New York, NY: McGraw-Hill.

[30]Vickery, B. C., and Vickery, A. (1987). *Information Science in Theory and Practice.* London: Butterworths.

[31]Wallace, T. F., and Kremzar, M. H. (2001). *ERP: Making It Happen.* New York, NY: John Wiley & Sons.