International Journal of Innovative
Computing, Information and Control
Volume 8, Number 6, June 2012

ICIC International ©2012 ISSN 1349-4198
pp. 4189–4203

# TRANSFER AN EPC BASED PROCESS MODEL INTO A HIERARCHICAL FUNCTIONAL MODEL TO SUPPORT VALUE CHAIN DEVELOPMENT

Chao Ou-Yang and Shuo Hong Wang

Department of Industrial Management
National Taiwan University of Science and Technology
No. 43, Sec. 4, Keelung Rd., Taipei 106, Taiwan
ouyang@mail.ntust.edu.tw

Received February 2011; revised June 2011

Abstract. *With the increasing complexity of business process model incorporated by ERP, to offer an effective and efficient process modeling tool is an important issue. Currently, a process model was usually included in a lengthy process containing many process components, which makes them difficult for managers and other users to learn. This study proposes an approach for transforming an Event Process Chain (EPC)-based process model into a hierarchical functional model. The proposed approach identifies structural errors in the process model and then converts it to a structured process model. A blocking approach is then used to form a hierarchical model by grouping sets of process elements into several sub-function units and to construct a hierarchical model.*
**Keywords:** Hierarchical model, Process model, Event process chain (EPC), ERP, Value chain

1. **Introduction.** The growing use of business application technologies such as ERP requires approaches to business modeling [1,2]. One example of a business model that incorporates ERP software is the "best practice" model for SAP. In most cases, business models are specified in terms of the process. In practice, however, a process model usually includes a lengthy process involving many process components and may therefore be difficult for users and managers to learn. A popular alternative to the process model is the functional approach. The upper part of Figure 1 shows the many tools such as IDEF and ARIS that apply the hierarchical concept to functional modeling. In this approach, a macro model is developed in the top level, and detailed models are then built in the lower level, which generates a tree structure. In other words, in the hierarchical approach, models in the lower layer tend to be detailed, and models in the higher level tend to be abstract. Therefore, the business model can be studied from the macro level to the micro level and *vice versa*. One advantage of this approach is that managers need only study the properly abstract model and can let subordinates perform the detailed process [3]. Another advantage is that process management and monitoring of the overall process status is generally easier in hierarchical structures [4].

Another benefit of this approach is that the "tree" structure concept is consistent with the idea of a value chain. That is, the core value functions in a business are usually expressed in the upper levels of the tree. Therefore, core business processes can be identified by investigating upper level functions. For instance, the simplified SAP R/3 reference model in Figure 2(a) specifies a portion of the "process stock material" business process [5]. Figure 2(b) shows how it can be re-modeled as a value chain model. The four core value chain functions, which are "assign purchase request", "process purchase request", "assign purchase order for stock material", and "delivery confirmation", are located in the first level of the hierarchical model and are thus easily identified by readers.

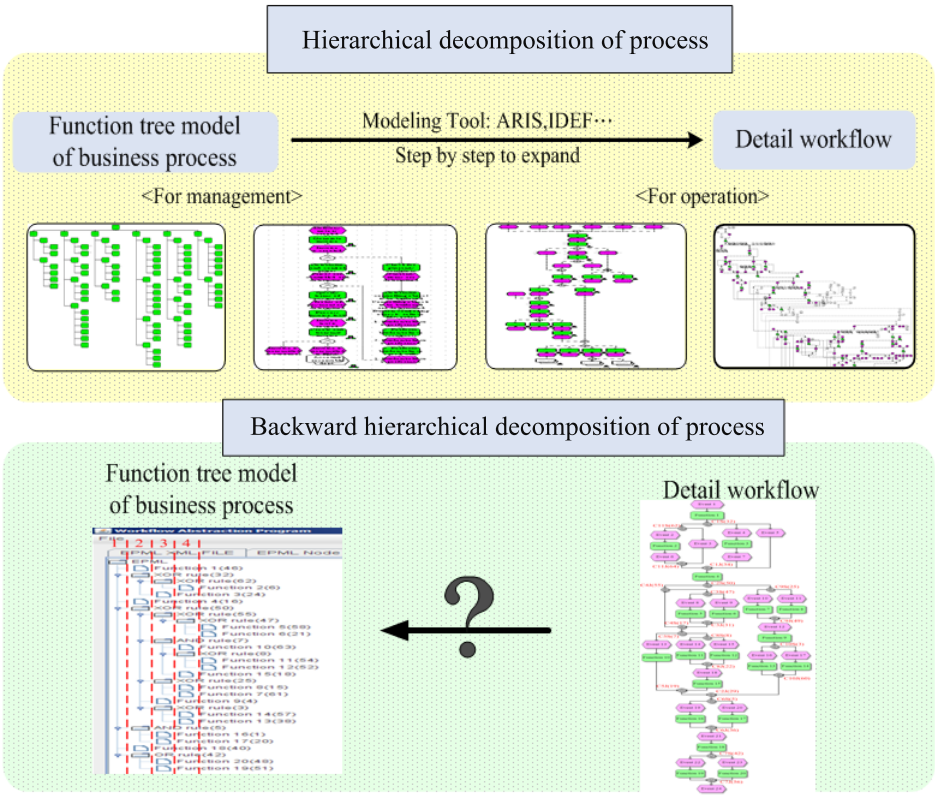



FIGURE 1. Relations between hierarchical model and process model

Business process models are often too detailed for study by management level personnel. A hierarchical function model can represent the core value chain in the upper levels and is thus suitable for study by managers. However, the process model must still be linked to its respective hierarchical model (lower part of Figure 1). Therefore, the problem addressed in this study is how to analyze the sequential and logical relationships among the functional constructs in a process model and how to properly transform it into a hierarchical model.

This study therefore proposes an approach for transforming an EPC-based process model into a hierarchical functional model. Structural errors identified in the process model are then corrected, which converts the model to a structured process model. A blocking approach is then used to group sets of process elements into several sub-functional units and to construct a hierarchical model. The converted hierarchical model can be used to support business process management and re-engineering by examining the structural complexity of the model.

2. **Background.** Application of information technology in the workflow and ERP domains has generated interest in applying business modeling technologies to support the building of ERP reference models. Until the late 1980s, most modeling methods focused on developing a comprehensive approach in a specific domain. Very few techniques have attempted to bridge the gaps among the domains. This situation changed in the mid 90s, when new approaches were proposed for integrating the modeling concepts from various perspectives. Beginning in year 2000, widespread adoption of IT in business resulted in modeling approaches that considered the environment of automated workflow systems. For instance, a three-layer method of discovering business processes was developed to account for inaccurate and incomplete business information [6]. The modeling approach

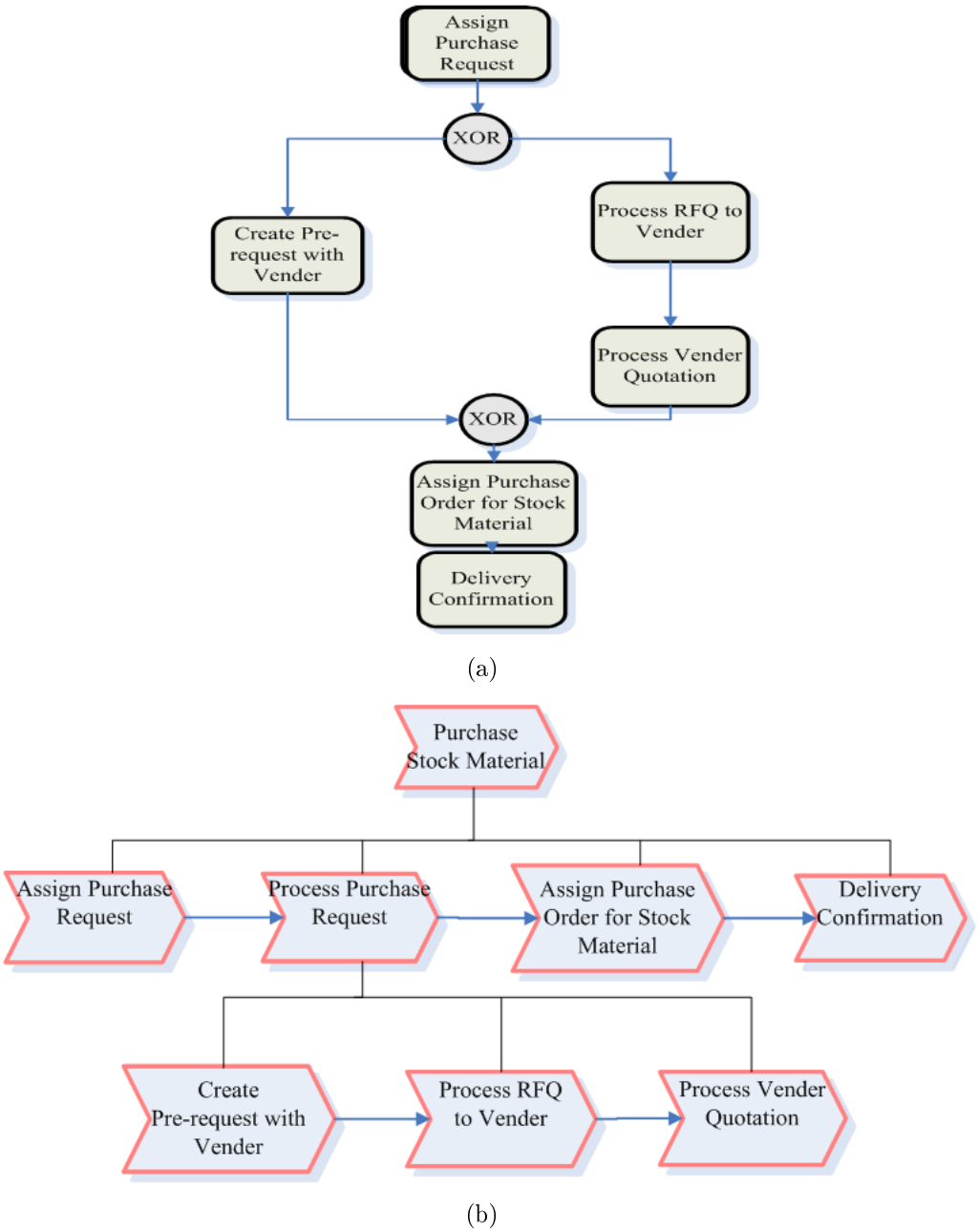



FIGURE 2. (a) A simplified business process model, (b) the hierarchical value chain model

used in another study addressed the development of dynamic business processes in the virtual enterprise environment [7]. Another study developed an approach for extracting business-relevant semantics from business logic. The approach in that study transformed extracted elements into the elements required in software architecture [8].

As mentioned above, most modeling methods focus on a specific domain. For example, IDEF0, IDEF1X and IDEF3 address modeling function, information, and process areas, respectively [9]. Although the various IDEF approaches have not been formally integrated, several works have developed techniques for integrating IDEF methods in a business framework such as a CIM system [10].

Researchers have applied IDEF approach in various domains. For example, comprehensive IDEF modeling approach was developed for and applied in manufacturing [11].

Another IDEF0 model was developed to integrate a maintenance system with an MRPII [12]. Additionally, several works have integrated IDEF with other modeling approaches such as UML [13,14].

Since the early 1990s, several works have proposed various integrated modeling techniques as comprehensive approaches to business modeling. Among them, CIMOSA and ARIS applied similar business modeling approaches [15,16]. Both approaches model a business from four perspectives: organization, information and function, and process. Additionally, the business modeling approach in both methods is performed in three stages: requirements definition, design specification and implementation.

The difference between these methodologies is that ARIS integrates three business views, organization, information and function, into the process view. In CIMOSA, however, business models are categorized into three layers: generic, partial and particular. The CIMOSA also provides modeling templates in various views for regulating modeling tasks.

Performance evaluation of workflow systems has been studied by many researchers. Petri net (PN) is among the most popular systems modeling methods. Basically, PN properties can be categorized as behavior properties and structure properties. The former are properties related to the flow of the tokens in the net such as reachabilities, liveness, and boundness. The latter are properties related to the entire structure of the net such as consistency and conservativeness [17]. Maeno et al. introduced an approximation method that uses PN to model the semiconductor manufacturing system and achieved efficient improvement in solving the optimal firing sequence problem via decomposing the PN [18]. Kim et al. introduced a process version management method according to PN structural patterns and version stamps to handle dynamic situations occurring in the process [19].

Recently, event process chain (EPC), a process modeling approach, has been studied intensively. For instance, a well-structured EPC model has been defined to prevent structural errors such as deadlock and multiply instance. Researchers have also studied the properties of structured models in terms of corresponding relationships of control elements in an EPC model [17]. Another approach, graph reduction, applies rules for progressively identifying the correct structure of an EPC model. The left portion of the model represents improper structures [20]. An approach transforming an EPC model into its corresponding PN model was developed to enable use of certain PN behavior properties such as liveness or boundness to analyze an EPC model [21]. Finally, one proposed approach simplifies the EPC-oriented SAP reference model based on proposed reduction rules. The reduced EPC can then be analyzed using a mixture of analytical methods of finding certain modeling errors [22].

Although modeling approaches such as IDEF0 and ARIS embed hierarchical modeling concepts so that readers can view the model from the top level abstract view to the lower level detailed view, other process modeling tools such as EPC lack this capability. Models that are lengthy and complex cause difficulty for users when learning the model.

## 3. The Proposed Approach.

Figure 3 shows the three-level framework of the proposed approach. In the conceptual level, the process model structure is analyzed such that certain improper structure behaviors are modified. The structured model is then divided into several blocks to form a tree structure and to build a hierarchical business model.

In the design level, the business model is constructed according to an EPC (event process chain) diagram. Improper structures such as improper nesting, deadlock, and multiple instances of an EPC model can then be analyzed and corrected. Finally, the structured EPC model is divided into several main blocks and sub-blocks. The grouped EPC model can then be transformed into a hierarchy model.

In the final design-specification level, an ARIS modeling tool is used to construct the EPC model. The XML file of the model is generated, and a Java-based XSLT (XML Translation Template) template is developed to translate the XML into EPML (EPC XML) file. The translated file provides the input to the software EPC Tools, which identify the unstructured model. The category of the improper structure is also identified so that the structure can be converted to a structured model. This model is grouped as several blocks in terms of the developed algorithm. The hierarchical structure of the EPC model is then constructed using the Java-based tools developed in this research.

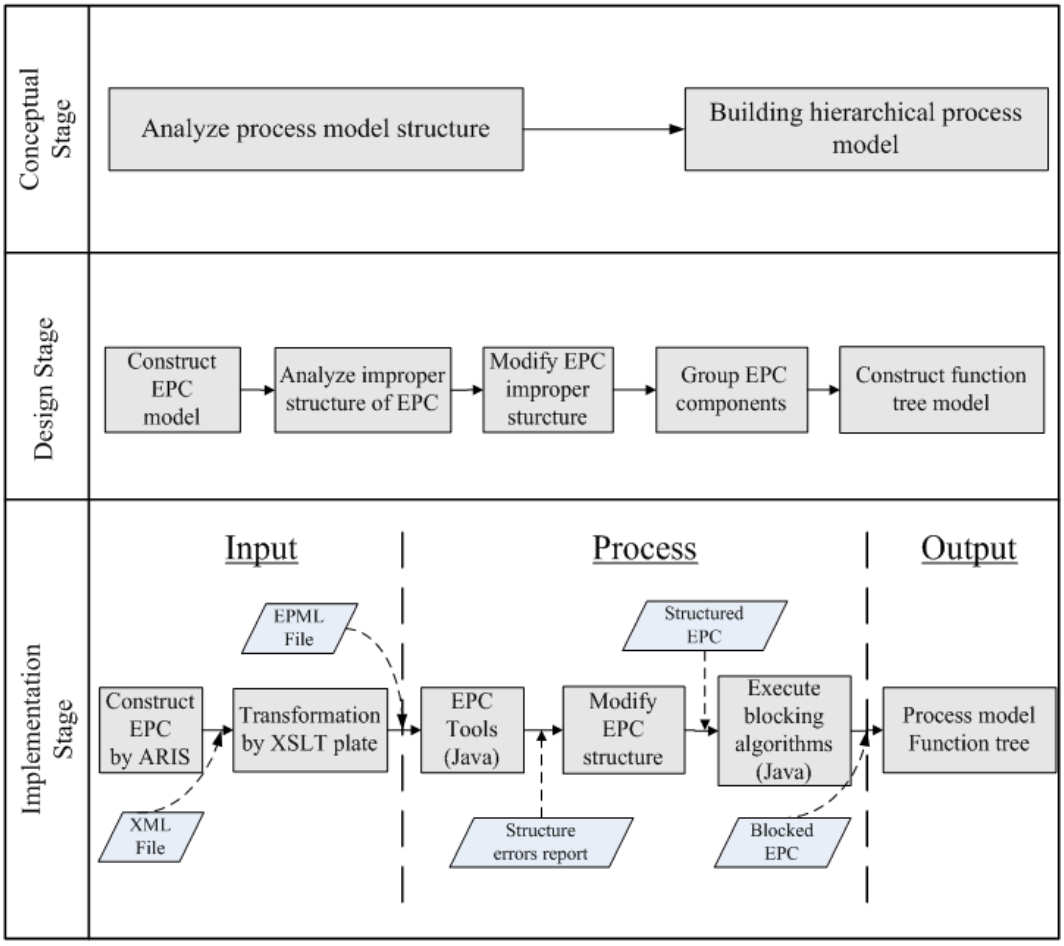



FIGURE 3. Proposed framework

3.1. **The conceptual level.** As noted above, the objective of the study was to develop an approach for viewing a process model hierarchically. However, building a tree-structure model requires modification of an improper model into a hierarchical structure. Therefore, the improper process must first be defined and identified. The methods of modifying unstructured models into their respective structured models must also be specified in advance. The next step is developing an approach for grouping a structured model into several main blocks and sub-blocks and for building a hierarchical model based on the blocked model.

3.2. **The design level.** This level specifies an approach for converting an unstructured business model to a structured model. The improper structure is identified first. The constructs that caused the improper structure are then identified. Approaches for modifying

the constructs and for converting an improper structure to a proper structure are proposed. The next stage then converts a structured model to its hierarchical form. Finally, a structure model is grouped into several main blocks containing respective sub-blocks, and the grouped blocks are then converted into a tree structure.

3.2.1. *Improper structure modification.* Figure 4 shows the approach for transforming an improper structure to a proper structure. The first stage identifies the improper structure of the process model, and the second stage analyzes the semantic errors in the model.

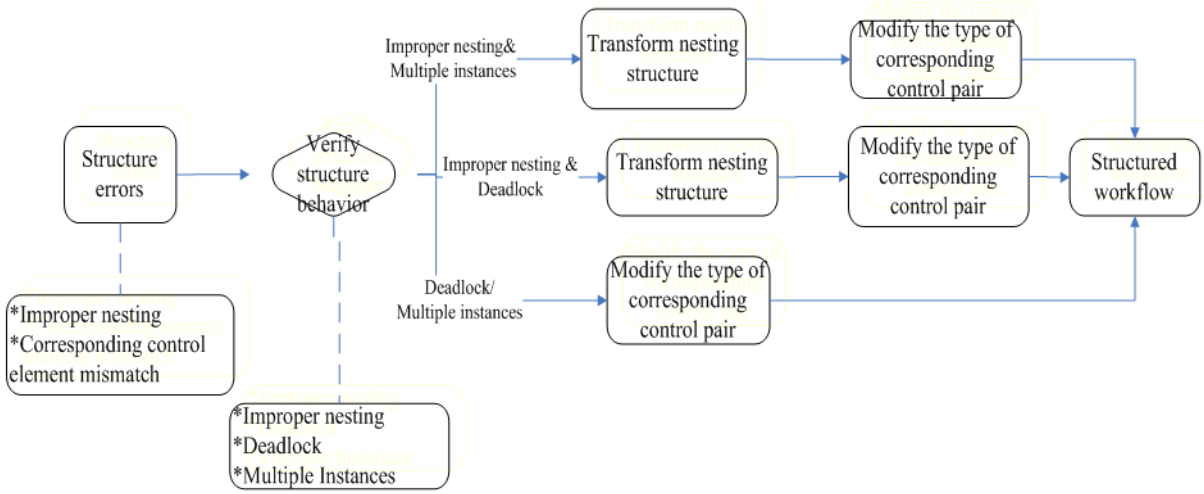



FIGURE 4. Improper structure modification

An improper structure in a process model has two possible causes: mismatched split-joint pairs or improper nesting [17]. Basically, a pair of corresponding control elements in a process model is matched split-joint pairs if either AND Split *versus* AND Join, OR Split *versus* OR Joint, or XOR Split *versus* XOR Joint. Otherwise, a mismatched split-joint pair occurs. Additionally, if two sets of matched split-joint pairs are not in a completely common path, improper nesting occurs.

In an improper structure model, deadlock and multiple instances are the two possible semantic errors [23]. Basically, most situations result from unmatched pairs of control elements.

After examining situations involving improper structure and semantic errors, the following outcomes are possible: (1) improper structure and multiple instances; (2) co-existing improper structure and deadlock; and (3) deadlock and multiple instances.

In the first situation, the improper nesting structure must be converted into a proper hierarchical structure so that the multiple instance problem can be solved by analyzing corresponding control elements.

In the second situation, improper structure and deadlock are co-existent. In addition to solving the improper structure problem, improper corresponding control elements must be identified, and the deadlock problem must be solved.

The final situations are deadlock and multiple instances. Since both result from improper corresponding control elements, corresponding elements must be analyzed and modified.

3.2.2. *Hierarchical model development.* Once a process model has been transferred to a structure model, the next task is constructing its hierarchical model. The two main tasks in this stage are grouping the model into several blocks based on the corresponding control

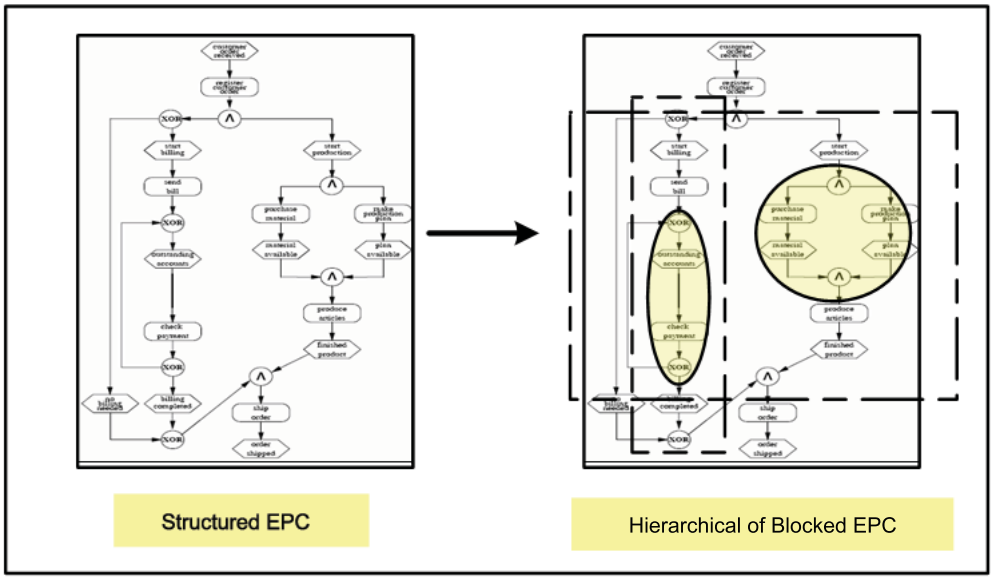



FIGURE 5. The concept of blocking a structured EPC model

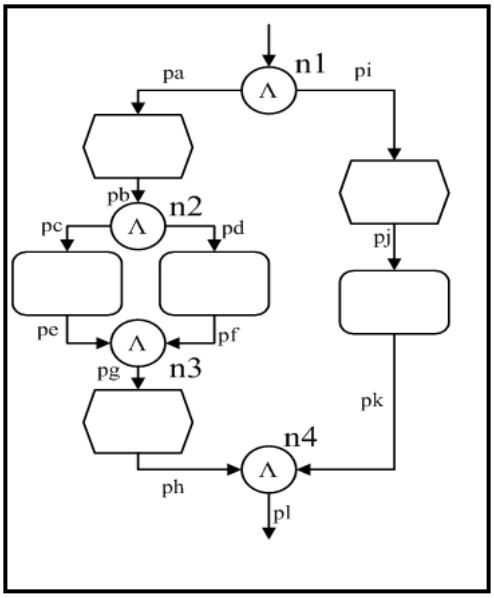



FIGURE 6. An example ECP model

elements and then developing the tree structure for this model based on the grouped blocks.

The first stage is grouping the model constructs into several blocks (Figure 5). The proposed approach first finds the corresponding control elements in the process model and then forms a group of blocks. The model in Figure 6 is used as an example.

The following symbols would be defined first:

$SP = \{sp_i, i = 1, \ldots, n\}$: a set of split control nodes in a process model.

$JP = \{jp_j, j = 1, \ldots, n\}$: a set of joint controls in a process model.

$L_{sp_i} = \{l^1_{sp_i}, l^2_{sp_i}, \ldots, l^n_{sp_i}\}$: a set of outgoing links connected split node $sp_i$.

$L_{jp_j} = \{l^1_{jp_j}, l^2_{jp_j}, \ldots, l^n_{jp_j}\}$: a set of incoming links connected joint node $jp_j$.

$FPATH^{i,j} = \{p^{i,j}_1, p^{i,j}_2, \ldots, p^{i,j}_n\}$: a set of forward paths from a split node $sp_i$ to a joint node $jp_j$.

$p_k^{i,j} = [l_{k,1}^{i,j}, l_{k,2}^{i,j}, \ldots, l_{k,n}^{i,j}]$: a path in a process model which is composed by a list of links $l_{k,n}^{i,j}$ in a sequence order.
$[p_k^{i,j}]^T = [l_{k,n}^{i,j}, l_{k,n-1}^{i,j}, \ldots, l_{k,1}^{i,j}]$: a reversed path for path $p_k^{i,j}$.
$RPATH^{j,i} = \{p_1^{j,i}, p_2^{j,i}, \ldots, p_n^{j,i}\}$: a set of reversed paths from a joint node $jp_j$ to a split node $sp_i$.

In Figure 6, the split node set and joint node set can be represented as $SP = \{n1, n2\}$, $JP = \{n3, n4\}$. Two links are connected to $n1$ and $n4$. Therefore, $L_{n1} = \{pa, pi\}$ and $L_{n4} = \{ph, pk\}$. Additionally, three paths from split node $n1$ to joint node $n4$ can be formulated as: $FPATH^{n1,n4} = \{p_1^{n1,n4}, p_2^{n1,n4}, p_3^{n1,n4}\}$. Each of the three paths can be represented as:

$$p_1^{n1,n4} = [pa, pb, pc, pe, pg, ph], \quad p_2^{n1,n4} = [pa, pb, pd, pf, pg, ph], \quad p_3^{n1,n4} = [pi, pj, pk, pl]$$

In addition, there are three reversed paths from joint node $n4$ to the split node $n1$. Therefore, it can be written as $RPATH^{n4,n1} = \{p_1^{n4,n1}, p_2^{n4,n1}, p_3^{n4,n1}\}$. Each of the three paths can be represented as:

$$p_1^{n4,n1} = [ph, pg, pe, pc, pb, pa], \quad p_2^{n4,n1} = [ph, pg, pf, pd, pb, pa], \quad p_3^{n4,n1} = [pl, pk, pj, pi].$$

It can be found that $[p_k^{i,j}]^T = p_k^{i,j}$ in this case.

Based on the above definitions, a pair of corresponding control elements can be defined as

**Definition 3.1.** *A split control element $sp_i$ and a joint element $jp_j$ form a pair of corresponding control elements if*

1. *For each outgoing link connected to $sp_i$ ($l_{sp_i}^k \in L_{sp_i}$), there exists at least one forward path $p_k^{i,j} \in FPATH^{i,j}$ such that $l_{sp_i}^k \in p_k^{i,j}$.*
2. *For each incoming link connected to $jp_j$ ($l_{jp_j}^k \in L_{jp_j}$), there exists at least one reversed path $p_k^{j,i} \in RPATH^{j,i}$ such that $l_{jp_j}^k \in p_k^{j,i}$.*
3. *For each identified forward path $p_k^{i,j}$ in $FPATH^{i,j}$, there exists a recognized reversed path $p_k^{j,i}$ in $RPATH^{j,i}$ such that $[p_k^{i,j}]^T = p_k^{j,i}$.*
4. *The $sp_i$ and $jp_j$ have the same Boolean type (AND, OR, XOR).*

A pair of corresponding control elements consists of the typical split and joint nodes. The first and second conditions indicate that each link connected to those two nodes must be in one of the paths from the split node to the joint node or *vice versa*. The third condition specifies that, for each path from the split node to the joint node, there exists a corresponding return path from the joint node to the split node.

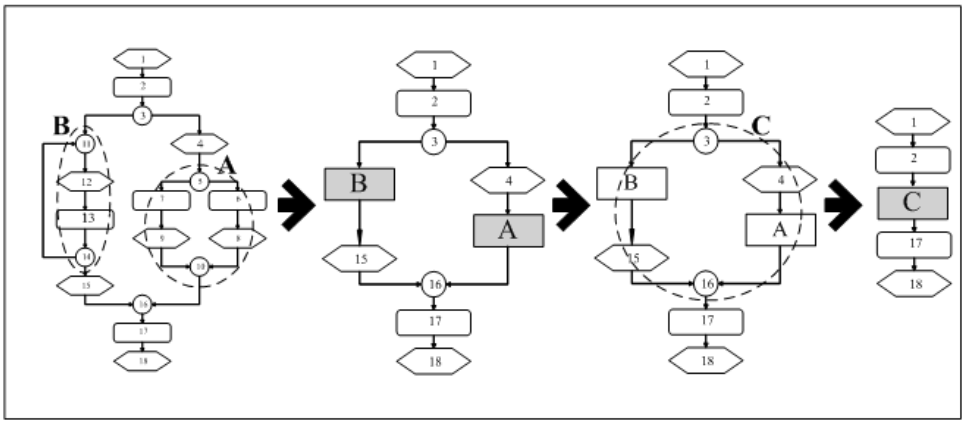
For instance, in Figure 6, split node $n1$ (which has two outgoing paths $pa$, $pi$) and joint node $n4$ (which has two incoming paths $ph$, $pk$) would form a pair of corresponding control points because

1. $pa \in L_{n1}$ and $p_1^{n1,n4} \in FPATH^{n1,n4}$, $p_2^{n1,n4} \in FPATH^{n1,n4}$ such that $pa \in p_1^{n1,n4}$ and $pa \in p_2^{n1,n4}$. In addition, $pi \in L_{n1}$ and $p_3^{n1,n4} \in FPATH^{n1,n4}$ such that $pi \in p_3^{n1,n4}$.
2. $ph \in L_{n4}$ and $p_1^{n4,n1} \in RPATH^{n4,n1}$, $p_2^{n4,n1} \in RPATH^{n4,n1}$ such that $ph \in p_1^{n4,n1}$ and $ph \in p_2^{n4,n1}$. In addition, $pk \in L_{n4}$ and $p_3^{n4,n1} \in RPATH^{n4,n1}$ such that $pk \in p_3^{n4,n1}$.
3. This condition has been specified in the previous paragraph. It can be found that for each $p_k^{n1,n4}$ *in* $FPATH^{n1,n4}$, and for each $p_k^{n4,n1}$ *in* $RPATH^{n4,n1}$, such that $[p_k^{n1,n4}]^T = p_k^{n4,n1}$ for $k = 1, 2, 3$.
4. Both of the node $n1$ and $n4$ are AND nodes,

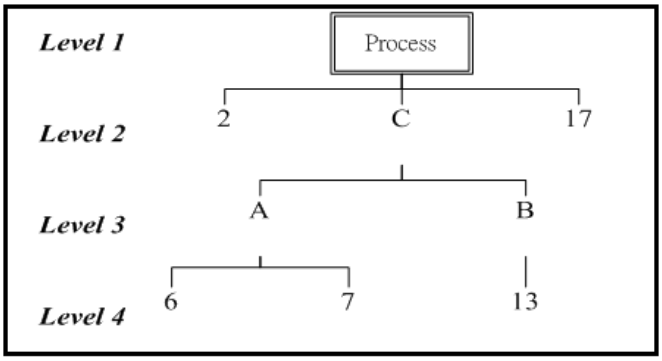
However, split node $n1$ and joint node $n3$ cannot form a pair of corresponding control elements since $pa \in L_{n1} pi \notin FPATH^{n1,n3}$.

Based on the above definition, an algorithm can be developed to form split/joint node pairs. Thus, for each pair, all constructs in the path, including the control nodes, event nodes, and function nodes, are grouped as a block (Figure 5).

The final step in this stage is building a hierarchical function model according to the information about the split/joint nodes and the grouped blocks. For instance, in Figure 7(a), the constructs between control nodes 4 and 10 can be grouped as a block (block A). The constructs between control nodes 11 and 14 can also be grouped as another block (block B). The constructs between control nodes 3 and 16 can then be grouped as a higher level block (block C). Finally, the remaining constructs can be grouped. Figure 7(b) shows the resulting hierarchical model.




(a)




(b)

FIGURE 7. (a) The sequence of building a hierarchical model, (b) the hierarchical model

The algorithm developed for constructing the tree model assumes that each event construct in an ordinary EPC model should follow a function construct. However, in this work, the event construct is not considered when constructing the hierarchical model since each functional construct in the model has a corresponding event construct.

The additional symbols were required to specify the algorithm.

$f_i$: *a function construct, where* $i = 1, \ldots, n$.

$F_i^k$: *the kth function block in the level* $i$ *of the tree model.*

A hierarchical model can be defined as:

**Definition 3.2.** *A hierarchical business model can be represented as* $F_i^k = \{F_{i+1}^{k+1}, F_{i+1}^{k+2}, \ldots, f_j, f_{j+1}, \ldots\}$ *where* $i$ *indicates the level of the tree.* $F_i^k$ *indicates the root node of the level*

*i of the tree. Its leaf nodes can be composed of a set of function blocks and/or a set of functional constructs.*

*According to this definition, the hierarchical model for the process shown in Figure 7(a) can be specified as*

$$F_1^P = \{F_2^C, f_2, f_{17}\}, \quad F_2^C = \{F_3^A, F_3^B\}, \quad F_3^A = \{f_6, f_7\} \text{ and } F_3^B = \{f_{13}\}.$$

*The steps for constructing the hierarchical model algorithm are as follows:*

1. *Start from the root node of the tree, which is* $F_i^P$ *and* $i = 1$.
2. *Find the first function node* $f_j$ *in the process model, and add the current level function block* $F_i^P = F_i^P + f_j$.
3. *Find the next node or stop search in the end of the process path.*
   - 3.1 *If that node is another function node* $f_{j+1}$*, then let* $F_i^P = F_i^P + f_{j+1}$*, and go to Step 3.*
   - 3.2 *If that node is a split control node sp, then*
     - 3.2.1 *Find its corresponding control element jp based on Definition 3.1, and go to Step 3.*
     - 3.2.2 *Generate a next level leaf node* $F_{i+1}^{\alpha}$*, and add* $F_i^P$*. That is,* $F_i^P = F_i^P + F_{i+1}^{\alpha}$.
     - 3.2.3 *Let the leaf node be the next level root node* $F_{i+1}^P = F_{i+1}^{\alpha}$*, and increase the level counter* $i = i + 1$*. For each split path, go to Step 3.*

For example, in the EPC model in Figure 7(a), the event nodes in the model would not be considered. The hierarchical model generation sequence would be

i Generate root node $F_1^P$.
ii Find the first function node (node-2). Let it be $f_2$, and add to $F_1^P$.
iii Find the next node (node-3), which is a split control node. Use Step 3.2 in the algorithm.
iv For Step 3.2.1, find the corresponding control element for node-3, which is node-16.
  i Return to Step 3 to find the next node, which is function node $f_{17}$. According to Step 3.1, add that node to $F_1^P$.
  ii Return to Step 3 to find the end of the path, and stop the search of this path.
v For Step 3.2.2, generate a next level functional block $F_2^C$, and add $F_1^P$.
vi For Step 3.2.3, let $F_2^C$ be the next level root node. For each split path, go to Step 3.

The rest of the process adds sub-function blocks $(F_3^A, F_3^B)$ into root $F_2^C$ since the path has two split nodes (node-11 and node-5). For each function block, certain function nodes $(f_6, f_7, f_{13})$ are then added separately. Figure 7(b) shows the final hierarchical model.

$$F_1^P = \{F_2^C, f_2, f_{17}\}, \quad F_2^C = \{F_3^A, F_3^B\}, \quad F_3^A = \{f_6, f_7\} \text{ and } F_3^B = \{f_{13}\}$$

Basically, the construction algorithm first identifies the main path and puts it in the first level of the tree. In that level, a node is also added for each pair of corresponding control elements found in the main path. The functions located between a pair of corresponding control elements are put in the next tree level and are linked to the corresponding node. This forward algorithm iteratively identifies the functions and control elements in the sub-paths and puts them in the respective sub-levels until each construct in the EPC model has been identified. The computational complexity of the algorithm is $O(n!)$ because of the iteratively identification procedure, where $n$ is the number of sub-levels of the hierarchical structure. In other words, the computation time will increase exponentially as the level of the process model increased.

The required condition in order to properly implement the proposed algorithm is that the workflow model should be in a proper structure form. The first stage of the proposed approach, to identify the improper structure of the model, would be a difficult task if the

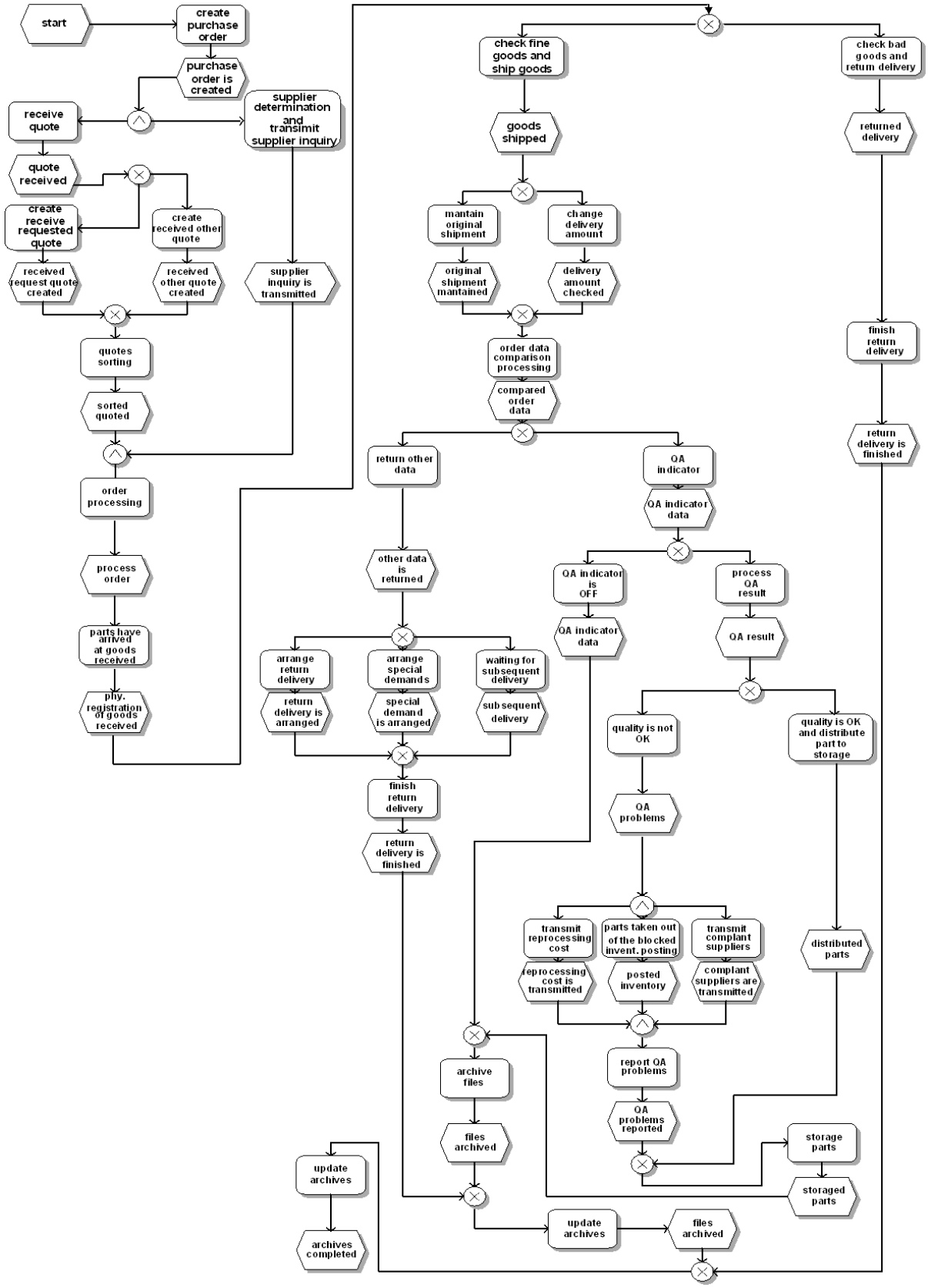



FIGURE 8. ARIS acquisition handling model

workflow model has a complicated structure. Therefore, it should carry out by experienced modelers.

3.3. **The implementation stage.** In this level, the EPC business model is constructed in ARIS. The XML file of the model is exported and translated to an EPML (EPC Markup Language) file through the Java-based XSLT translation templates developed by Mendling et al. [24]. This EPML file provides input to "EPC Tools", which are used to test the semantics of the model [23]. This tool simulates the EPC process model and identifies improper structures in the model. The model should then be modified by the analyzer according to the principles defined in this study. The structured model is then analyzed by a Java-based algorithm, which translates it into a model represented by a set of blocks. This model is then transformed into a hierarchical tree model by another Java-based OR converted to a hierarchical tree model by another Java-based module.

4. **Implementing the Proposed Approach.** An "acquisition handling" event process chain (EPC) model from ARIS "Mechanical Engineering Reference Model" library was selected as an empirical model. This model was developed according to ARIS industrial

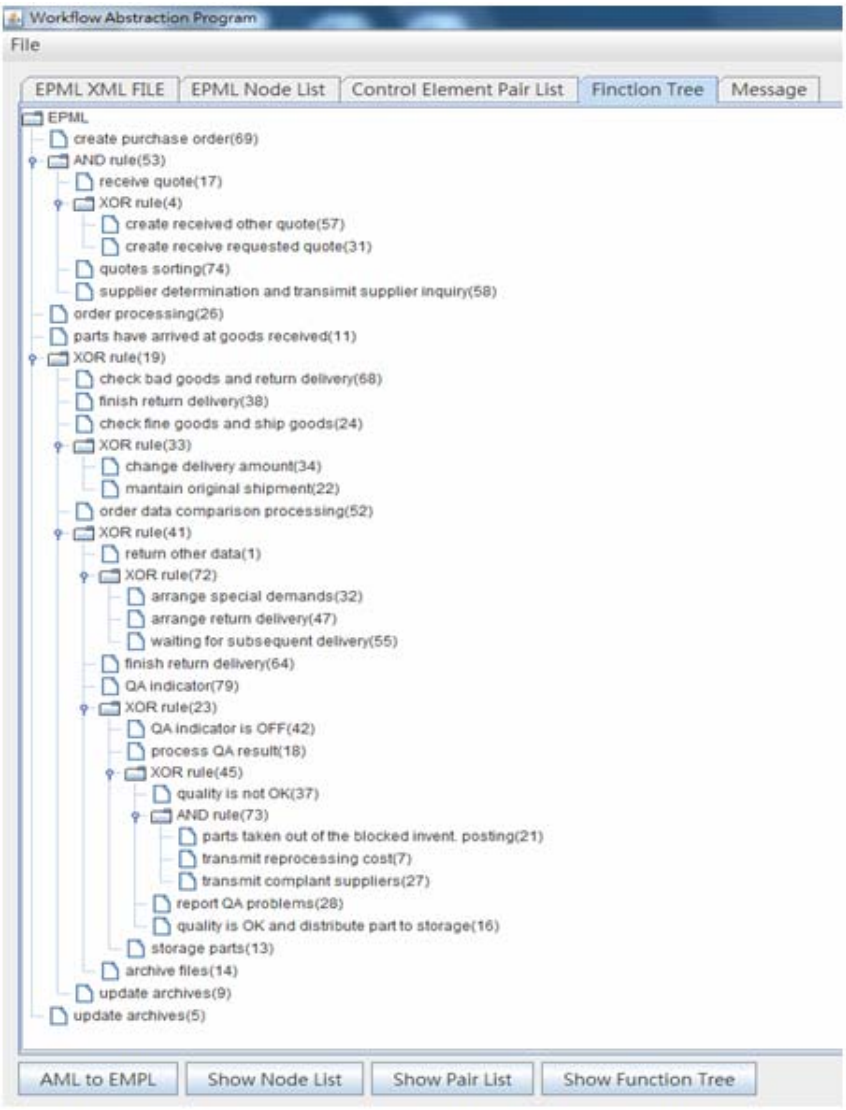



FIGURE 9. Hierarchical tree structure of the acquisition model

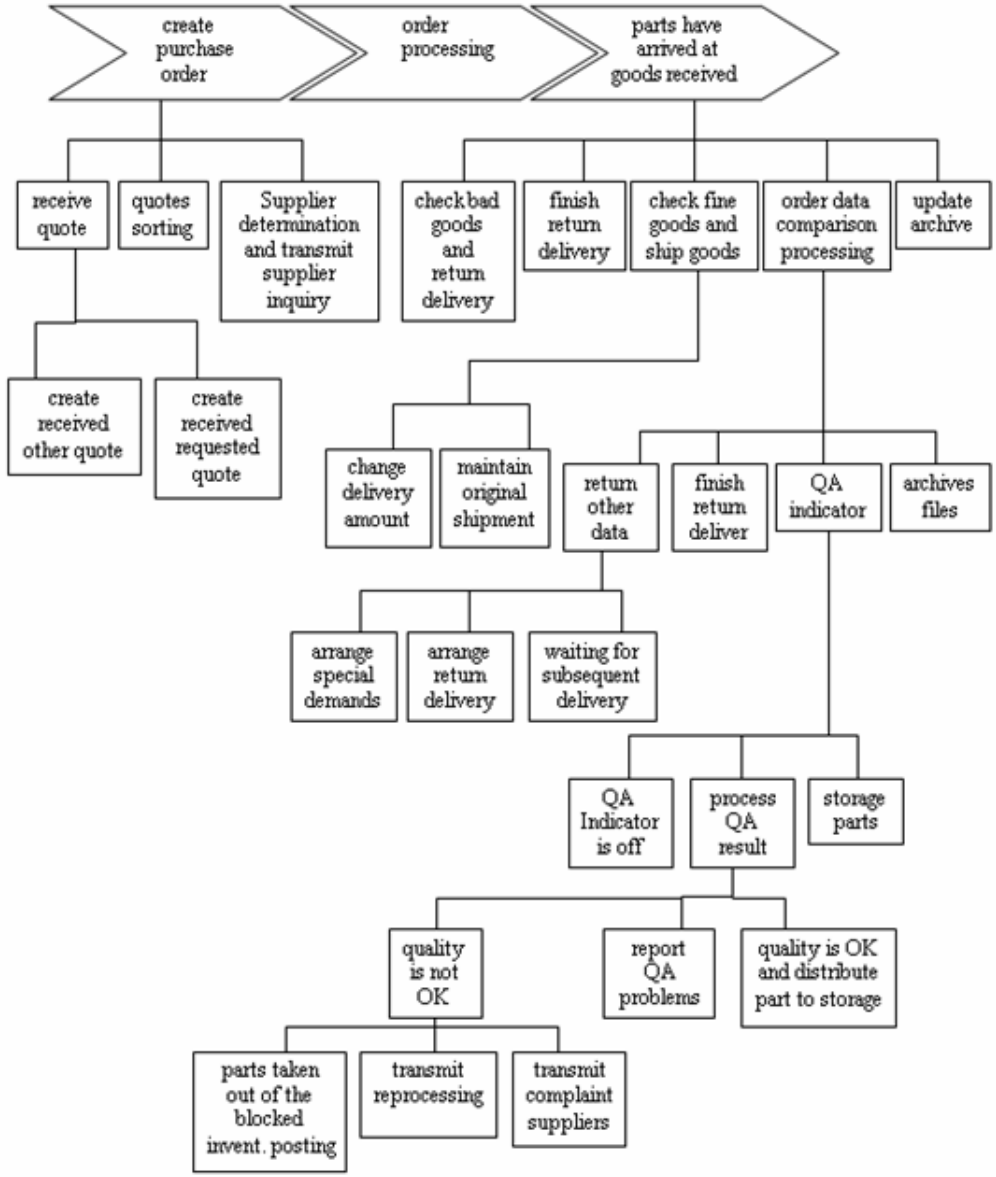



FIGURE 10. The value-chain model

experience. Further details for the ARIS business reference modeling can be found in Scheer [16]. Figure 8 shows the model. This lengthy process involves about thirty functions. It can be found that this model contains multi-level structure. That is, one split path is embedded in another split route. As mentioned, matrix decomposition approach such as DSC requires extra effort to group the activities contained between a split-joint pair of nodes. However, the proposed approach and the developed Java-based tool could identify the hierarchical structure of the "acquisition handling" process model (Figure 9).

The identified hierarchical structure model can aid construction of the value chain model. Figure 10 shows the main value chain model of the acquisition handling process. It can be found that, there are three core functions in this process: "create purchase order", "order processing" and "part arrived and good receiving". Among those, the third function contains more sub-level functions than the others.

The constructed hierarchical model and related value-chain model can be used to support business process management and re-engineering. For instance, from the process

management points of view, core functions "part arrived and good receiving" might support too many sub-level activities and hence require more resources than the others. In addition, the bottleneck might occur if the responsible organization unit is understaffed. From re-engineering point of view, the related process activities contained by the third function might need to be totally re-designed by using IT technology to reduce the consumed resources and to avoid inefficiency.

5. **Conclusion and Discussion.** This work developed an approach for identifying the functional hierarchy of an EPC model and then constructing a tree model. This work can support the construction of a complex process model from its abstract level to the detailed stage.

The major contributions of this paper are the following:

1. A structural approach for modifying an unstructured EPC model and then converting it into a multi-level hierarchical tree model.
2. An approach for finding the pairs of corresponding control elements from an EPC model.
3. A hierarchical modeling approach for supporting value chain development in the business process.

The authors recommend the following further research works:

1. Analysis of process flow diversity after logic change in the control elements.

In this work, a properly structured EPC model required a pair of corresponding control elements with similar control type. However, a change in control type may affect the decision flow of the process model. Therefore, an approach is needed for analyzing how modifying a control element in a model affects process flow.

2. Extract the semantic meaning of a functional block.

In this work, once a functional block was identified, the label of the leading function was used as a tag of the function block. However, in the real world, this approach might be unsuitable for a block with many constituted functional elements. An approach is needed for analyzing the semantic meanings of labels and relationships of the functions in order to find a semantic label that properly represents the function block.

**Acknowledgment.** The authors would like to thank the National Science Council of Taiwan for financially supporting this research under Contract No. NSC-96-2221-E-011-024.

## REFERENCES

[1] J. Mendling, H. M. W. Verbeek, B. F. van Dongen, W. M. P. van der Aalst and G. Neumann, Detection and prediction of errors in EPCs of the SAP reference model, *Data & Knowledge Engineering*, vol.64, no.1, pp.312-329, 2008.

[2] M. L. Rosa, F. Gottschalk, M. Dumas and W. M. P. van der Aalst, Linking domain models and process models for reference model configuration, *Business Process Management Workshops Lecture Notes in Computer Science*, vol.4928, pp.417-430, 2008.

[3] D. J. Elzinga, T. Horak, C. Y. Lee and C. Bruner, Business process management: Survey and methodology, *IEEE Transactions on Engineering Management*, vol.42, no.3, pp.119-128, 1995.

[4] F. Leymann, D. Roller and M. T. Schmidt, Web services and business process management, *IBM Systems Journal*, vol.41, no.2, pp.198-211, 2002.

[5] T. Curran, G. Keller and A. Ladd, *SAP R/3 Business Blueprint, Understanding the Business Process Reference*, Prentice Hall, 1998.

[6] K. Xu, L. Liu and C. Wu, A three-layered method for business processes discovery and its application in manufacturing industry, *Computers in Industry*, vol.58, no.3, pp.265-278, 2007.

[7] P. Grefen, N. Mehandjiev, G. Kouvas, G. Weichhart and R. Eshuis, Dynamic business network process management in instant virtual enterprises, *Computers in Industry*, vol.60, no.2, pp.86-103, 2009.

[8] Q. Yao, J. Zhang and H. Wang, Business process-oriented software architecture for supporting business process change, *Proc. of the International Symposium on Electronic Commerce and Security*, pp.690-694, 2008.

[9] C. Menzel and R. J. Mayer, The IDEF family of languages, in *Handbook on Architectures of Information Systems*, P. Bernus, K. Mertins and G. Schmidt (eds.), Springer, 2006.

[10] A. Doniavi, A. R. Mileham and L. B. Newnes, A systems approach to photolithography process optimization in an electronics manufacturing environment, *International Journal of Production Research*, pp.2515-2528, vol.38, no.11, 2000.

[11] Q. Liu, X. Sun, S. M. Mahdavian and S. Ding, Establishment of the model for flexible manufacturing system based on CORBA and IDEF0, *The International Journal of Advanced Manufacturing Technology*, vol.42, no.3-4, pp.301-311, 2009.

[12] W. H. Ip, C. K. Kwong and R. Fung, Design of maintenance system in MRPII, *Journal of Quality in Maintenance Engineering*, vol.6, no.3, pp.177-191, 2000.

[13] C. H. Kim, R. H. Weston, A. Hodgson and K. H. Lee, The complementary use of IDEF and UML modeling approaches, *Computers in Industry*, vol.50, no.1, pp.35-56, 2003.

[14] J. M. Dorador and R. I. M. Young, Application of IDEF0, IDEF3 and UML methodologies in the creation of information models, *International Journal of Computer Integrated Manufacturing*, vol.13, no.5, pp.430-445, 2000.

[15] F. B. Vernadat, The CIMOSA languages, in *Handbook on Architectures of Information Systems*, P. Bernus, K. Mertins and G. Schmidt (eds.), Springer, 2006.

[16] A. W. Sheer and K. Shneider, ARIS – Architecture of integrated information systems, in *Handbook on Architectures of Information Systems*, P. Bernus, K. Mertins and G. Schmidt (eds.), Springer, 2006.

[17] R. Liu and A. Kumar, An analysis and taxonomy of unstructured workflows, *Proc. of the 3rd International Conference on Business Process Management*, pp.268-284, 2005.

[18] R. Maeno, M. Konishi and J. Imai, Petri net solver for semiconductor manufacturing plan, *International Journal of Innovative Computing, Information and Control*, vol.4, no.12, pp.3193-3206, 2008.

[19] D. Kim, N. Lee, S. Kang, M. Cho and M. Kim, Business process version management based on process change patterns, *International Journal of Innovative Computing, Information and Control*, vol.6, no.2, pp.567-575, 2010.

[20] H. Lin, Z. Zhao, H. Li and Z. Chen, A novel graph reduction algorithm to identify structural conflicts, *IEEE Proc. of the 34th Annual Hawaii International Conference on System Science*, 2002.

[21] H. M. W. Verbeek, T. Basten and W. M. P. van der Aalst, Diagnosing wokflow pocesses using Woflan, *The Computer Journal*, vol.44, pp.244-279, 2001.

[22] B. F. van Dongen, M. H. Jansen-Vullers, H. M. W. Verbeek and W. M. P. van der Aalst, Verification of the SAP reference models using EPC reduction, state-space analysis, and invariants, *Computers in Industry*, vol.58, no.6, pp.578-601, 2007.

[23] N. Cuntz and E. Kindler, On the semantic of EPCs: Efficient calculation and simulation, *Proc. of the 3rd International Conference on Business Process Management*, pp.398-403, 2005.

[24] J. Mendling and M. Nüttgens, Transformation of ARIS markup language to EPML, *Proc. of the 3rd GI Workshop on Event-Driven Process Chains (EPK 2004)*, pp.27-83, 2004.