Classification by feedback structure and partitioning into acyclic subgraphs for a cyclic workflow graph

Yongsun Choi

Dept. of Systems Management & Engineering, Inje University, Kimhae, Korea, yschoi@inje.ac.kr

Abstract

This paper introduces a novel method of partitioning a cyclic workflow graph into the subgraphs of acyclic flows. The way of iterative classification of nodes according to feedback structures and deriving subgraphs of acyclic flows is described with illustrative examples. The proposed method allows a cyclic workflow model to be analyzed further, if necessary, with several smaller subflows, which are all acyclic.

Keywords: cyclic workflow graph, graph algorithms, rank, feedback, subflow

1. Introduction

The recent surge of e-business research and development has led to increasing interest in the field of workflow management ([2], [4], [7], [11]). Workflow management systems automate business processes, represented in pertinent workflow models, by coordinating and controlling the flow of work and information among various participants [12] .

Workflow models must be correctly defined before being deployed in a workflow management system to avoid any costly maintenance delays due to runtime errors in the process model [3]. Graph-structured workflow models provide a great flexibility for depicting complex process behavior in a fairly compact form. This free-form nature, on the other hand, yields models that may fall at the discretion of the modeler and create modeling situations that cannot be executed or will behave in a manner not expected by the modeler [5].

Graph reduction [10] or block-wise abstraction [6] has been proposed to identify structural conflicts in workflow graphs, but both approaches are limited to acyclic models [1]. Lin et al. [9] extended the graph reduction technique to handle cyclic models, but with the cost of higher complexity [1].

This paper introduces a novel method of partitioning a cyclic workflow graph into the subgraphs of acyclic flows. An algorithm to compute the rank of a node, defined with elementary paths, in a cyclic workflow graph is introduced. The way of iterative classification of nodes according to feedback structures and deriving subgraphs of acyclic flows is described with illustrative examples.

2. Workflow models in directed graphs and matrix representation

A workflow graph is a directed graph $\boldsymbol{G} = [\boldsymbol{V}, \boldsymbol{T}]$ with a set of nodes $\boldsymbol{V}$ and a set of arcs $(i, j) \in \boldsymbol{T}$, where $i, j \in \boldsymbol{V}$. Each arc, called as a transition, links two nodes and represents the execution order of nodes. A node is classified into two types, task and coordinator. A *task*, represented by a rectangle, stands for the work to be done to achieve some objectives. A *coordinator*, represented by a circle, stands for a point where succeeding path(s) to follow is selected or different paths are merged. Depending on the types of nodes and the number of incoming and outgoing transitions, nodes can be classified into 5 categories, i.e., *sequence*, *AND-split*, *AND-join*, *OR-split*, and *OR-join*, as shown in Figure 1. *Start* and *End* nodes are used to indicate the beginning and the end of the given workflow process, respectively.

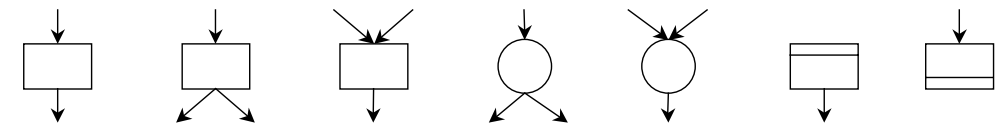


Figure 1. Classification of nodes in workflow graphs

Figure 2(a) illustrates an example workflow with cycles, modified from [10]. In general a workflow process is represented as a sparse graph with $M << N^2$, where $M = |\boldsymbol{T}|$ is the number of transitions and $N = |\boldsymbol{V}|$ is the number of nodes. Figure 2(b) shows the column-wise compacted adjacency matrix, with two tables $\alpha(\bullet)$ of dimension $N+1$ and $\beta(\bullet)$ of dimension M. For each node $i \in \boldsymbol{V}$, table β lists the $\boldsymbol{pred}(i)$, where $\boldsymbol{pred}(i) = \{ j \mid (j, i) \in \boldsymbol{T} \}$, starting from the entry numbered $\alpha(i)$. We make a simplifying assumption on the workflow graph that a node cannot be a join and

split at the same time, which can be converted into a join node and a split node with a transition between them.

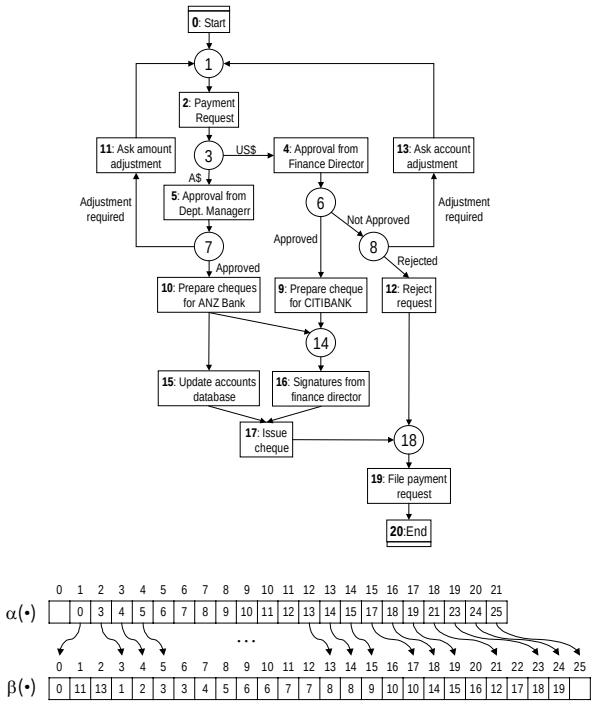


Figure 2. The graph and the column-wise compacted adjacency matrix for an example cyclic workflow

3. Graph normalization with ranks of nodes

By following the transitions from the Start node and visiting the nodes of a cyclic workflow model, transitions that lead to node(s) previously visited can be identified. The procedure **IdentifyFJT( )** identifies all the *feedback join transitions.* This algorithm traverses different transition(s) at each iteration and has complexity of *O*(*M*). Table 1 shows, in part, the results when the above procedure applied to the workflow of Figure 2.

Procedure IdentifyFJT ( ) {
  FJT ← ϕ . ***Ancs***(*i*) ← ϕ for each node *i*.
  P ← {Start}
    *// **P** is the set of nodes of which successor*
    // nodes are to be visited at current iteration
  Repeat Until ***P*** = ϕ {
    P_{next} ← ϕ
    For all *(i, j)* ∈ ***T*** where *i* ∈ ***P*** {
      If *j* ∈ ***Ancs*** *(i)*, ***FJT*** ← ***FJT*** + *(i, j)*
      Else { ***Ancs*** *(j)* ← ***Ancs*** *(j)* + (*i* + ***Ancs*** (*i*));
        P_{next} ← P_{next} + *j* }
    }
    P ← P_{next}
  }
}

TABLE 1. COMPUTATION EXAMPLE OF THE PROCEDURE

iteration	Results of steps
1	***FJT*** = ϕ, . ***Ancs***(*i*) ← ϕ for each node *i*. ***P*** = { 0 }. ***Ancs***(1) = { 0 }.
2	***P*** = { 1 }. ***Ancs***(2) = { 0,1 }.
…	…
7	***P*** = { 8,9,10,11 }. ***Ancs***(12) = ***Ancs***(13) ={ 0,1,2,3,4,6,8 }; ***Ancs***(14) = { 0,1,2,3,4,5,6,7,9,10 }; ***Ancs***(15) = { 0,1,2,3,5,7,10 }; 1 ∈ ***Ancs***(11) → ***FJT*** = {(11, 1)}
8	***P*** = { 12,13,14,15 }. ***Ancs***(16) = { 0,1,2,3,4,5,6,7,9,10,14 }; ***Ancs***(17) = { 0,1,2,3,4,5,6,7,9,10,14,15, 16 }; ***Ancs***(18) = { 0,1,2,3,4,5,6,7,8,9,10,12,14,15, 16 }; 1 ∈ ***Ancs***(13) → ***FJT*** = {(11, 1), (13, 1), }.
…	…

For an acyclic and connected graph, the *rank* of node *i*, denoted by *r(i)* with *r(Start)* = 0, with the complexity of *O(M)*[8]. We define the rank of a node for a cyclic workflow graph by restricting the path should be elementary, i.e. the path should not meet the same node twice. With all *feedback join transitions* removed, the rank of each node can be computed from the resulting acyclic connected workflow graph with the complexity of *O(M)*.

Figure 3 shows the normalized graph of the workflow process in Figure 2, with nodes rearranged by their ranks indicated. Figure 4 shows another example of normalized graph for a workflow with nested cycles.

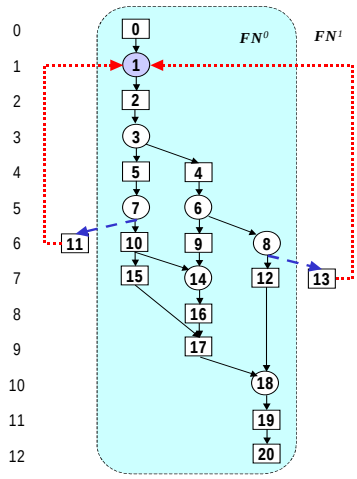


Figure 3. Normalized graph of workflow in Figure 2

4. Identification of feedback structures and partitioning into the subgraphs of acyclic flows

From the normalized graphs of Figure 3 and 4, we could find some interesting things. First, every dotted upstream transition (*i*, *j*), with *r*(*i*) > *r*(*j*), merges a feedback flow. In addition, each dashed transition

initiates a new feedback flow. Transition (13, 7) in Figure 4 (b) has both characteristics. These transitions can be utilized as the cut sets to group the nodes in $\boldsymbol{V}$.

We will call these upstream transitions, from a node to another one of lower or equal rank, as *Feedback Join Transitions*, marked as $\boldsymbol{FJT}$, where

$$\boldsymbol{FJT} = \{ (i, j) \in \boldsymbol{T}, r(i) > r(j) \}. \quad (1)$$

$\boldsymbol{FJT}$ = {(11, 1), (13, 1)} and $\boldsymbol{FJT}$ = {(11, 7), (13, 7), (23, 19), (24, 1)} for the examples shown in Figure 2 and 4, respectively. Each transition of $\boldsymbol{FJT}$ corresponds to each different elementary cycle of Figure 2 and 4, respectively

Definition 1. For any $i \in \boldsymbol{V}$, let the *Order of Feedback of i*, called by *of(i)*, denote the minimum number of transitions of $\boldsymbol{FJT}$ to pass through to reach the *End* node. The subset of nodes with *of(i)* = *n* will be called as n^{th}*-order Feedback Nodes* and denoted by $\boldsymbol{FN}^n$.

Note that $\boldsymbol{V} = \cup_n \boldsymbol{FN}^n$. The subgraph spanned by $\boldsymbol{FN}^0$ will be called as the *main flow* and will be denoted as $\boldsymbol{MF}$*(Start,End).*

Nodes of $\boldsymbol{FN}^0$ will not need any transitions in $\boldsymbol{FJT}$ to reach the *End* node, and $\boldsymbol{FN}^0$ can be identified as follows,

$$\boldsymbol{FN}^0 = \{ i \mid R_{\boldsymbol{FJT}}(i, End) = 1 \}, \quad (2)$$

where $R_{\boldsymbol{FJT}}$ *(i, End)*, which can be computed with complexity of *O(M),* denote the *reachability* of node *i* to the *End* node *without passing through any transitions in* $\boldsymbol{FJT}$. That is, $R_{\boldsymbol{FJT}}$ *(i, End)* = 1 when node *i* can reach the *End* node without passing through any transitions in $\boldsymbol{FJT}$, or $R_{\boldsymbol{FJT}}$ *(i, End)* = 0 otherwise. $R_{\boldsymbol{FJT}}$ *(End, End)* is defined to be 1. Table 2 shows $R_{\boldsymbol{FJT}}$ *(i, End)* for each node *i*, resulting $\boldsymbol{V} - \boldsymbol{FN}^0 = \{ 11, 13 \}$ for the workflow in Figure 2. It can be shown that $\boldsymbol{V} - \boldsymbol{FN}^0 = \{ 11, 16, 17, 18, 19, 20, 21, 22, 23, 24 \}$ for the workflow in Figure 4.

TABLE 2. $R_{\boldsymbol{FJT}}(I, END)$ OF EACH NODE I FOR THE WORKFLOW IN FIGURE 2

i	0	1	2	3	4	5	6	7	8	9	16	11	12	13	14	15	16	17	18	19	20
$\boldsymbol{R}_{\boldsymbol{FJT}}$ *(i,*	1	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	1	1	1	1	1

Definition 2. The n^{th}*-order Feedback Joins*, denoted by $\boldsymbol{FJ}^n$, is the subset of feedback joins that merge the subgraph spanned by $\boldsymbol{FN}^{n+1}$ to the subgraph spanned by $\boldsymbol{FN}^n$. That is $\boldsymbol{FJ}^n = \{ j \mid (i, j) \in \boldsymbol{T}, i \in \boldsymbol{FN}^{n+1} \cup \boldsymbol{FS}^n, j \in \boldsymbol{FN}^n \}$. The n^{th}*-order Feedback Splits*, denoted by $\boldsymbol{FS}^n$, is the subset of feedback splits that split the subgraph spanned by $\boldsymbol{FN}^{n+1}$ from the subgraph spanned by $\boldsymbol{FN}^n$. That is $\boldsymbol{FS}^n = \{ i \mid (i, j) \in \boldsymbol{T}, i \in \boldsymbol{FN}^n, j \in \boldsymbol{FN}^{n+1} \cup \boldsymbol{FJ}^n \}$.

Condition (1) can also be utilized to identify *Feedback Joins of all order*, denoted by $\boldsymbol{FJ}$**,** where

$$\boldsymbol{FJ} = \cup_n \boldsymbol{FJ}^n = \{ j \mid (i, j) \in \boldsymbol{FJT} \}. \quad (3)$$

$\boldsymbol{FJ}$ = { 1 } and $\boldsymbol{FJ}$ = { 1, 7, 19 } for the examples shown in Figure 2 and 4, respectively. Note that

$$\boldsymbol{FJ}^n = \boldsymbol{FJ} \cap \boldsymbol{FN}^n. \quad (4)$$

From $\boldsymbol{FJ}^0 = \boldsymbol{FJ} \cap \boldsymbol{FN}^0$, we now get $\boldsymbol{FJ}^0$ = { 1 } and $\boldsymbol{FJ}^0$ = { 1, 7 } for the examples of Figure 2 and 4, respectively.

Finally, $\boldsymbol{FS}^0$ will be identified. Since $\boldsymbol{FN}^1$ is not available yet and there will be no transition from a node in $\boldsymbol{FN}^0$ to any node in $\cup_{n\geq 2} \boldsymbol{FN}^n$, we identify $\boldsymbol{FS}^0$ as follows,

$$\boldsymbol{FS}^0 = \{ i \mid (i, j) \in \boldsymbol{T}, i \in \boldsymbol{FN}^0, \text{ and } j \in (\boldsymbol{V} - \boldsymbol{FN}^0) \cup \boldsymbol{FJ}^0 \} \quad (5)$$

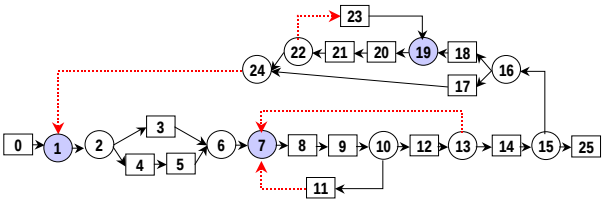
$\boldsymbol{FS}^0$ = { 7, 8 } and $\boldsymbol{FS}^0$ = { 10, 13, 15 } for the workflow in Figure 2 and 4, respectively.

At every iteration, our method identifies $\boldsymbol{FN}^n$, $\boldsymbol{FJ}^n$ and $\boldsymbol{FS}^n$. When all the nodes in $\boldsymbol{FJ}$ are classified into one of $\boldsymbol{FJ}^n$, we can conclude that $\boldsymbol{FN}^{n+1} = \boldsymbol{V} - \cup_{m\leq n} \boldsymbol{FJ}^m$ and no further classification of nodes is required. For instance, $\boldsymbol{FJ}^0 = \boldsymbol{FJ}$ for the workflow in Figure 2, and we can conclude with $\boldsymbol{FN}^1 = \boldsymbol{V} - \boldsymbol{FN}^0 = \{ 11, 13 \}$. Otherwise, there exist more feedback structures in the subgrpah spanned by $\boldsymbol{V} - \cup_{m\leq n} \boldsymbol{FJ}^m$ and further classification of nodes is required.

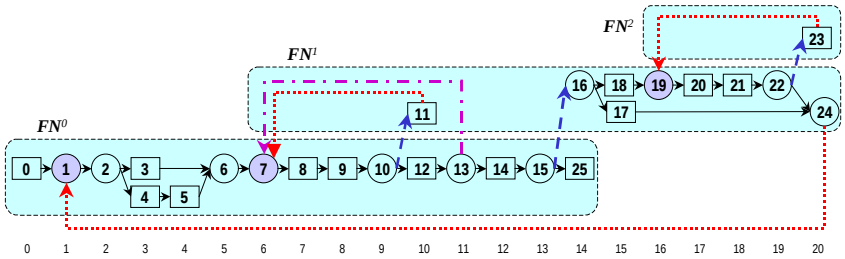
Definition 3. For any $fs \in \boldsymbol{FS}^{n-1}$, let $\boldsymbol{Desc}^n$ (*fs*) denote the set of nodes in $\boldsymbol{FN}^n$ and $\boldsymbol{Desc}^{n+}$ (*fs*) denote the set of nodes in $\cup_{m\geq n} \boldsymbol{FN}^m$, respectively, that can be reached from *fs* by the transitions in $\boldsymbol{T}$.

Definition 4. Let $fs \in \boldsymbol{FS}^{n-1}$ *and* $fj \in \boldsymbol{FJ}^{n-1}$. The n^{th}*-order Feedback Flow* $\boldsymbol{FF}^n$(*fs, fj*) denote the subgraph spanned by the set of nodes { *fs, fj* } $\cup$ $\boldsymbol{Desc}^n$ (*fs*). The $(n+)^{th}$*-order Feedback Flow* $\boldsymbol{FF}^{n+}$(*fs, fj*) denote the subgraph spanned by the set of nodes { *fs, fj* } $\cup$ $\boldsymbol{Desc}^{n+}$ (*fs*).

With $\boldsymbol{FN}^0$, $\boldsymbol{FJ}^0$, $\boldsymbol{FS}^0$, and $\boldsymbol{FN}^1$, the workflow of Figure 2 can be partitioned into, the main flow $\boldsymbol{MF}$ (0, 20), $\boldsymbol{FF}^1$(7, 1) spanned by {7, 11, 1}, and $\boldsymbol{FF}^1$(8, 1) spanned by {8, 13, 1}, requiring no further partitioning. For the workflow of Figure 4, we have $\boldsymbol{MF}$ (0, 25), $\boldsymbol{FF}^1$(10, 7), $\boldsymbol{FF}^1$(13, 7), and $\boldsymbol{FF}^{1+}$(15, 1). Since $\boldsymbol{FF}^{1+}$(15, 1) contains a node of $\boldsymbol{FJ}$ not classified yet, i.e. node 19 $\notin \boldsymbol{FJ}^0$, it contains some cyclic structure and requires further partitioning. In similar way, we can get $\boldsymbol{FN}^1$ = { 11, 16, 17,18, 19, 20, 21, 22, 24 }, $\boldsymbol{FJ}^1$ = { 19 }, and $\boldsymbol{FS}^1$ = { 2 } from $\boldsymbol{FF}^{1+}$(15, 1). Since nodes in $\boldsymbol{FJ}$ are fully classified, we can conclude that $\boldsymbol{FN}^2 = \boldsymbol{V} - \boldsymbol{FN}^0 - \boldsymbol{FN}^1 = \{ 23 \}$ and derive additional acyclic subgraphs of $\boldsymbol{FF}^1$ (15, 1) and $\boldsymbol{FF}^2$ (22, 19). Table 3 summarizes the results of the analyses for the workflow graphs of Figure 2 and 4.



(a) A workflow with nested cycles



(b) Normalized graph

Figure 4. Normalization of a workflow graph with nested cycles

TABLE 3. SUMMARY OF PARTITIONING

Case	Target graph	Classified nodes	Derived subgraphs
Workflow of Figure 2	***G(V,T)***	***FJ*** = { 1 }; ***V*** – ***FN***^{0}= { 11, 13 }, ***FJ***^{0}= { 1 }, ***FS***^{0}= { 7, 8 }.	***MF*** (0, 20) ***FF***^{1} (7, 1) ***FF***^{1} (8, 1)
Workflow of Figure 4	***G(V,T)***	***FJ***= { 1, 7, 19 }; ***V*** – ***FN***^{0}= { 11, 16, 17,18, 19, 20, 21, 22, 23, 24 }, ***FJ***^{0}= { 1, 7 }, ***FS***^{0}= { 10, 13, 15}.	***MF*** (0, 25) ***FF***^{1} (10, 7) ***FF***^{1} (13, 7) ***FF***$^{1+}$ (15, 1)
	FF^{1+} (15, 1)	***FN***^{1}= { 11, 16, 17,18, 19, 20, 21, 22, 24 }, ***FJ***^{1}= { 19 }, ***FS***^{1}= { 22 }; ***FN***^{1}= { 23 }.	***FF***^{1} (15, 1) ***FF***^{2} (22, 19)

Let q be the maximum degree of feedback and r be the average number of subgraphs in $\boldsymbol{FN}^{n}$ that need to be further partitioned, in the given workflow graph. Identification of $\boldsymbol{FN}^{n}$ (with $\boldsymbol{FS}^{n-1}$ *and* $\boldsymbol{FJ}^{n-1}$) from $\boldsymbol{FF}^{(n-1)+}$ (fs^{n-1}, fj^{n-1}), where $fs \in \boldsymbol{FS}^{n-1}$ and $fj \in \boldsymbol{FJ}^{n-1}$, is rather straight forward with $R_{\boldsymbol{FJT}}$ (fs^{n-1}, fj^{n-1}) that has complexity $O(M')$, where $M' < M$ is the number of transitions in the cyclic subgraph $\boldsymbol{FF}^{(n-1)+}$ (fs^{n-1}, fj^{n-1}). Therefore, complexity of this step will be $O(qrM)$.

5. Concluding Remarks

In this paper, we proposed a novel method of partitioning a cyclic workflow graph into the subgraphs of acyclic flows. We showed that how to identify the main flow and consecutively partition off subgraphs of acyclic feedback flows. Each finally derived subgraph of a feedback flow matches for the corresponding elementary cycle in the given workflow graph. The proposed method allows a cyclic workflow model to be analyzed further, if necessary, with several smaller subflows, which are all acyclic. For instance, graph reduction technique [9], if applied to a cyclic workflow graph to be partitioned into s subflow graphs, is allowed to handle cyclic workflow models and even improve the performance from $O((M+N)^2 \cdot N^2)$ ([1], [9]) to $O((M+N)^2 \cdot N^2 / s^3)$.

References

[1] Aalst, W. M. P. van der, "An alternative way to analyze workflow graphs," *14th Int. Conf. On Adv. Info. Sys. Eng.*, pp. 535-552, 2002.

[2] Aissi, S., P. Malu, and K. Srinivasan. "E-business process modeling: the next big step," *IEEE Computer*, vol. 35, no. 5, pp. 55-62, 2002.

[3] Basu, A. and R. W. Blanning, "A formal approach to workflow analysis," *Information Systems Research*, vol. 11, no. 1, pp. 17-36, 2000.

[4] Basu, A. and A. Kumar, "Research commentary: Workflow management issues in e-Business," *Information Systems Research*, vol. 13, no. 1, pp. 1-14, 2002.

[5] Business Process Management Initiative, *Business Process Modeling Notation*, Working Draft 1.0, August 2003, available at http://www.bpmi.org/

[6] Choi, Y. and J. L. Zhao, "Matrix-based abstraction and verification of e-business processes," *Proc. the 1st Workshop on e-Business*, pp. 154-165, 2002.

[7] Delphi Group, *BPM2002: Market Milestone Report*, available at http://www.delphigroup.com/.

[8] Gondran, M. and M. Minoux, *Graphs and Algorithms*, John Wiley & Sons Ltd., 1984.

[9] Lin, H., Z. Zhao, H. Li, and Z. Chen, "A novel graph reduction algorithm to identify structural conflicts," *Proc. of the 35th Hawaii Int. Conf. On Sys. Sci.*, IEEE Computer Society Press, 2002.

[10] Sadiq, W. and M. E. Orlowska, "Analyzing process models using graph reduction techniques," *Information Systems*, vol. 25, no. 2, pp.117-134, 2000.

[11] Sheth, A. P., W. M. P. van der Aalst, and I. B. Arpinar, "Processes driving the networked economy," *IEEE Concurrency*, vol. 7, no. 3, pp. 18–31, 1999.

[12] Workflow Management Coalition, *Glossary. Document Number WfMC-TC-1011*, 1999.