AMCIS 2003 Proceedings

Americas Conference on Information Systems (AMCIS)

December 2003

Mapping Petri Nets and Metagraphs: A Step Towards Inter-Organizational Workflows

Sharad Barkataki
Purdue University

MAPPING PETRI NETS AND METAGRAPHS: A STEP TOWARD INTERORGANIZATIONAL WORKFLOWS

Sharad Barkataki
Krannert School of Management
Purdue University
barkatak@mgmt.purdue.edu

Abstract

In a world of increasing inter-connectedness between firms, workflow management systems that can inter-operate with each are becoming of increasing importance. In light of the number of different modeling techniques used to model workflows, mapping between different workflow modeling techniques is an important problem. This paper proposes a mapping between Petri Nets and Metagraphs based on the underlying characteristics of the two models. Providing such a map would enable workflow management systems based on different metamodels to understand and communicate with each other to a greater degree than the current state of the art permits. In addition, providing a mapping between these two related modeling techniques would allow designers the ability to use the strengths of each the techniques. Finally, it would be a step in the direction of a Unified Modeling Language which could be used to model both metamodels.

Introduction

The 1990's was a decade of rapid expansion of the internet and interconnectedness between business. Over this period we saw a movement from the traditional industrial base of the economy to a new information based economy, with companies increasingly interconnected to each other. This economy has raised a number of exciting opportunities for business. Concurrent with the growth in the internet we saw the rise of new business paradigms and buzzwords such as business process re-engineering, Electronic Data Interchange (EDI), enterprise resource planning systems (ERP), customer relationship management systems (CRM), innovative supply chain management tools, and the idea of the virtual organization. In this new age, the interoperation of business processes has become increasingly important. For example, it would be useful for suppliers to have real time knowledge of the movement of their products through the distribution system, the hot sellers and the "lame ducks". Thus a manufacturer might know instantly the state of sales of certain products at a retailer such as Walmart, allowing them to adjust their prices and production accordingly.

Another development of the information economy was the increasing use of Workflow Management Systems (WMS) to analyze and automate business processes. These WMS can be based on a variety of metamodels including Petri Nets (van der Aalst and Kumar, 2002; van der Aalst, 1998) and Metagraphs (Basu and Blanning, 2000; 2001). Each of these metamodels has their own distinct advantages and are orientated to handle certain questions in workflow analysis. In the interconnected information economy, it is inevitable that workflows will cross organizational boundaries (Basu and Kumar, 2002; Bussler 1999; Casati and Shan, 2001; van der Aalst and Kumar, 2002). For example, we might have a workflow for a dynamic composite service that requires services from several different firms, necessitating interorganizational coordination and cooperation of their workflows in a dynamic setting. As these workflows cross organizational boundaries, it has become very important that different workflow management systems be able to communicate and understand each other; especially in scenarios where the WMS are using different underlying metamodels. In such an environment, a mapping between the different metamodels would go some way in allowing different WMS to understand and communicate effectively with each other. In addition, any mapping between different metamodels would allow workflow designers to use the relative advantages of the different modeling techniques. Finally, any mapping between the metamodels would be a step in the direction of a Unified Modeling Language being developed to address the problem of interoperability of WMS.

The purpose of this paper is to provide such a mapping between two graph theoretic workflow modeling tools: Petri Nets and Metagraphs. The paper is divided into several sections. In the section on Inter-Organizational Workflows, we survey some of the existing approaches to inter-operable workflows. In the section on Workflow Modeling, we discuss the advantages and disadvantages of two workflow modeling tools: Metagraphs and Petri Nets. In section 4, we provide the mapping between the two metamodels. In section 5, we discuss the future directions of this research.

Inter-Organizational Workflows

Several attempts have been made at addressing the question of inter-operable workflows. In Bussler (1999), the author outlines a number of important elements that inter and intra organizational WMS would need to address. In Klingeman et. al (1998), the authors address the issue of inter-operable workflows by modeling workflows as services which can be outsourced to different organizations. They attempt to determine the optimal level of outsourcing in such an environment, but manage to neglect to issue of how different WMS will communicate and interact with each other. Casati and Shan (2001) address the issue of adaptable WMS in a dynamic environment, given that WMS already understand each other through a centralized e-services platform.

A significant attempt at addressing interoperable WMS has been the work of van der Aalst and Kumar (2002). They propose an architecture and a language called XRL (eXchangeable Routing Language) based on XML to route work through different organizations. They also map the XRL constructs to Petri Net constructs. The XRL framework they propose would thus route work through different organizations by providing a Petri Net representation of the work that needed to be completed at each node. However, this in itself creates two kinds of problem. The first arises even if two nodes are using Petri Net based WMS. In such an environment, though the WMS may understand that a Petri Net has been generated by XRL/flower, there still exists the problem of reconciling the meanings of the transitions and places of the Petri Net representation produced by the XRL/flower workflow engine with their own Petri Net representation. This problem could be rectified by a kind of centralized data dictionary that stores the meanings of the transitions, though the authors do not address the issue. The second problem arises when different nodes rely on different WMS. In such a case, though the XRL/flower engine would generate a Petri Net for the work that needed to be done at a particular node, the WMS at the node would need a means of translating that representation back into its own representation.

The provision of a translation between a Petri Net and a Metagraph would thus enable the XRL idea to work in an environment where the nodes are using different underlying metamodels for their WMS i.e. the particular node is using a Metagraph based approach. Thus, the provision of such a mapping would thus not only allow designers the added flexibility of being able to use multiple modeling techniques to do workflow analysis and automation, but also serve to extend the XRL idea and allow interoperability on a grander scale than was addressed by van der Aalst and Kumar (2002).

Workflow Modeling

Metagraphs

Metagraphs are graphical structures that represent directed relationships between sets of elements. They extend features of both digraphs and hypergraphs. Metagraphs were first introduced as a tool for modeling decision support systems by Basu and Blanning (1994). In that paper, the authors introduced a number of analytical operations that could be performed on these structures to facilitate analysis. In Basu and Blanning (2000), the authors show how this same analytical construct could be used to model workflows. A basic metagraph can thus be defined as follows (Basu and Blanning, 2000): Given a finite generating set $X = \{x_i, i = 1...I\}$, a metagraph is an ordered pair $S = \langle X, E \rangle$ in which $E = \{e_k, k = 1...K\}$ is a set of edges. Each $x_i \in X$ is called an element. Each edge is an ordered pair $e_k = \langle V_k, W_k \rangle$ in which $V_k \subseteq X$ is the invertex of the edge e_k and $W_k \subseteq X$ is the outvertex of edge *k*. The coinput of any $x \in V_k$ is $V_k \setminus \{x\}$ and the cooutput of any $x \in W_k$ is $W_k \setminus \{x\}$. A *conditional* metagraph is a metagraph of the form $S = \langle X_p \cup X_v, E \rangle$ in which X_p is the set of propositions and X_v is a set of variables (i.e. the remaining elements). We can see that the basic metagraph defined is a special case of the conditional metagraph with $X_p = \varnothing$.

We can see from the above definitions and Basu and Blanning (2000) that conditional metagraphs are sets of information elements that are connected to each by directed edges. These edges may have attributes and the information elements themselves could be separated into different types, such as assumptions and data. The information elements in the invertex of the edge represent the information elements that need to be present before an edge can be enabled or done. The elements in outvertex represent the result of moving along the edge.

Basu and Blanning (2000) were able to define a number of operations on metagraphs that allowed varied analysis of workflows based on a single metagraph representation of the workflow. Specifically they introduced procedures that allowed an *information centric* workflow model based on a metagraph to be easily transformed into *task centric* and *resource centric* views of the workflow. The advantages of such an approach were that it allowed designers to easily answer not only questions about relationships between different information elements, but also allowed them to easily answer questions about the relationships between tasks, resources, and the different components. In addition these different views of the *information centric* metagraphs were themselves metagraphs, thus allowing the analysis tools available for metagraphs to be used on them. Consequently, the metagraph is able to provide a comprehensive view of the whole workflow from multiple perspectives using a single construct. The useful operations that are available for metagraphs include the *projection* operation that allows designers to focus on certain parts of the workflow exclusively, while simultaneously allowing designers to hide sensitive parts of the workflow from different types of users. This would be especially important in today's environment where security concerns are very important. In addition, there are *Context* metagraphs that allow the analysis of a process when certain propositions are true or false. Several other operations were defined for metagraphs including the *addition* operation (which could be used to combine lower and higher level metagraphs) and the *multiplication* operation (which could be used to find the lengths of paths between elements). In Basu and Blanning (2001), the authors extend metagraphs to be able to handle attributes related to temporal constraints, thus extending metagraphs to be able to handle scheduling of tasks.

However, despite these advantages, there are few drawbacks to using this approach. Compared to Petri Nets, metagraphs are a relatively new idea and hence the range of rigorous tools available for validation, verification, and performance analysis are limited. Additionally, metagraphs are not well suited to handle cases and states. Thus, monitoring and control and the identification of bottlenecks become more difficult in a metagraph modeling environment. Despite these shortcomings, metagraphs remain a powerful tool for workflow analysis and WMS.

Petri Nets

Petri Nets are graphical and mathematical modeling tools that have been around for a long time. They were first introduced in 1962, and like metagraphs are based on graph theory. Petri Nets eventually found their way into modeling WMS (van der Aalst, 1998). A Petri Net is a directed, weighted, bipartite graph with two types of nodes called *places* and *transitions.* Connections between two nodes of the same type are not allowed. Thus arcs are either from places to transitions or vice versa. In the first case, the place is called an input place. In the second, the place is called an output place. A *marking* (*state*) assigns to each place a non-negative integer k. If a marking assigns to a place p a non-negative integer k, we say that p is marked with k tokens. In a graphical representation, places are generally drawn as circles, and transitions are drawn as boxes. A Petri Net can thus be defined as a 4 tuple, $PN = (P, T, F, M_0)$ where:

P is a finite set of place

T is a finite set of transitions $(P \cap T = \varnothing)$

$F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs (flow relations)

$M_0 \colon P \to \{0, 1, 2, 3, \cdots\}$ is the initial marking

In modeling systems, places can represent a number of different things including conditions, conclusions, input or output data, input or output signals, and buffers, depending on what is being modeled. In the context of WMS used to model business processes, places could thus represent input or output data or conditions and conclusions (Murata, 1989; van der Aalst, 1998). Transitions on the other hand can represent events or tasks. A transition is said to be *enabled* if every one of its input places has a token. We can thus think of input places as the information necessary for a task to be enabled. In the same way, we can think of output places as the output data or conclusion of a fired transition. Once a transition has been enabled, it can be *fired* by a trigger i.e. the transition can be fired either automatically, or by a user, external message or a clock. We can thus define 4 types of *triggers* (van der Aalst, 1998): *automatic, user, message,* and *time.* Once a transition has fired, tokens move from the input places to the output places of the transition. Colered Petri Nets are extensions of classical Petri Nets with colered tokens to represent different cases (Jensen, 1997, van der Aalst, 1998).

The main advantages of WMS based on Petri Nets are many. In fact, several commercial systems (COSA, INCOME, LEU) already rely on Petri Net based systems. The main advantage of Petri Nets are that several powerful analysis techniques already exist and that Petri Net systems are inherently suited to doing case based analysis. Thus, it relatively easy to monitor and control a WMS based on a Petri Net, enabling easier identification of bottlenecks. In addition, simulation tools exist to validate whether a workflow behaves as expected, as well as tools to do performance analysis such as analyzing throughput times. Finally, and most importantly, several tools exist for verifying the correctness of workflows based on Petri Nets. For example, van der Aalst (1998) defines the Soundness property as a means of verifying the correctness of workflows. Checking the *soundness* of a workflow would help to verify the correctness of the workflow, including checking for dangling tasks and dead tasks. In addition, it easy to identify bad constructions in workflows using Petri Net based analysis. Murata (1989) identifies several properties that can be checked on Petri Nets including identifying whether or not a Petri Net is reachable or not, which would help in identifying dangling tasks and bottlenecks. One can also identify whether a Petri Net is bounded and safe, which would guarantee that there are no buffer overflows. In addition, one can check whether a Petri Net is *live* which ensures deadlock free operation no matter what firing sequence is chosen.

The biggest disadvantage of Petri Nets are of course that they do not handle resource management as well as Metagraphs. Other disadvantages include the inability to answer the same kind of questions that Metagraphs can answer about task interactions or do the aforementioned things that Metagraphs do well. Thus, any translation of Metagraphs and Petri Nets to each other would allow designers to take advantage of the relative strengths of both modeling tools.

Mapping Petri Nets to Metagraphs

We will now outline the main elements necessary to map Petri Nets to Metagraphs. Note that we will be talking about *information element centric view* Metagraphs when we refer to Metagraph from now on. When we refer to the elements of WMS based on Petri Nets, we will be relying on the analysis of van der Aalst (1998), unless otherwise stated.

First we note that transitions in Petri Nets and directed edges in the Metagraph represent tasks. Thus, it is quite easy to see that a transition on a Petri Net must be equivalent to an edge in a Metagraph. We will consequently say that edges and transitions are equivalent. The result is stated in the following proposition:

Proposition 1: *Any transition t in a Petri Net must have an equivalent edge e in a Metagraph.*

We have noted previously that transitions require all their input places to contain a token before they are enabled. We should also note that these places can be thought of as the input data or information necessary before a task can be started (Murata, 1989). Thus, we argue that all input places *i* of a transition must be represented as the information elements that are necessary to complete a task. As such, the information elements necessary to complete a task in a Metagraph are equal to the elements in the invertex of the edge. Thus we get the following proposition:

Proposition 2: *Every input place i of a transition in a Petri Net must be represented as an invertex element in the respective edge of the metagraph.*

The output places *o* are the results of the transition firing and the task being completed. They can only be reached after a task has been completed. Thus we get the following proposition:

Proposition 3: *Every output place o in a Petri Net must be represented by an equivalent information element in the outvertex of the respective edge in the Metagraph.*[1]

The next problem we address is the one related to triggers (van der Aalst, 1998). We note that the *automatic* or *user* triggers associated with transitions in Petri Nets, refer to human resources or automated systems. Equivalently, Metagraphs represent resources as labels on the edges. Thus, we present the following proposition:

Proposition 4: *Any user or automatic trigger on a transition t in a Petri Net, is represented by a resource label on respective edge e on the metagraph.*

We model every *message* trigger (external trigger) for a transition *t* as an information element in the invertex of the respective edge *e*. The reasoning for this is that the external trigger refers to some external message or date that a transition needs before it

[1]Both the Murata (1989) and the van der Aalst (1998) papers both explicitly state that places represent data or conditions. Information elements also represent data or conditions (Basu and Blanning, 1994; 2000).

can fire. As such, the external trigger then represents the information necessary before a transition can fire or execute. Thus, we argue that external trigger must then be equivalent to the information elements required before edge *e* can execute i.e. it must be an information element in the invertex. However, we note that if we transform the external trigger into simply another information element, we lose some information in the mapping from the Petri Net to the metagraph. Thus, we propose that to facilitate the mapping of a Metagraph back into a Petri Net, we need to recognize that the external trigger is a special type of information element. Since Metagraphs already incorporate different types of information elements (namely normal information elements and proposition or assumption information elements), there is no loss to defining a third type of information element as an external trigger. Indeed Basu and Blanning (2000) suggest we should add additional types into the generating set to aid in evaluation (for example in terms of Context Metagraphs and so on). Let this new type be called *Message* information elements. Thus we arrive at the following proposition:

Proposition 5: *Every message (external) trigger for a transition t in a Petri Net, is represented as a message information element in the invertex of the respective edge* e_k *in the metagraph i.e.* $x_m \in V_k$

Using a similar argument, we model every *time* trigger for a transition *t* as an information element called *Time* Information element in the invertex of the respective edge *e*. Thus we also have the following proposition:

Proposition 6: *Every time trigger for a transition t in a Petri Net, is represented as a time information element in the invertex of the respective edge in the metagraph i.e.* $x_t \in V_k$

Finally, we are left with trying to map the routing constructs of Petri Nets to Metagraphs. The easiest constructs to map are the AND-Split and AND-Join constructs which represent parallel routing in Petri Nets. An example of a parallel routing scheme is given in Figure1:

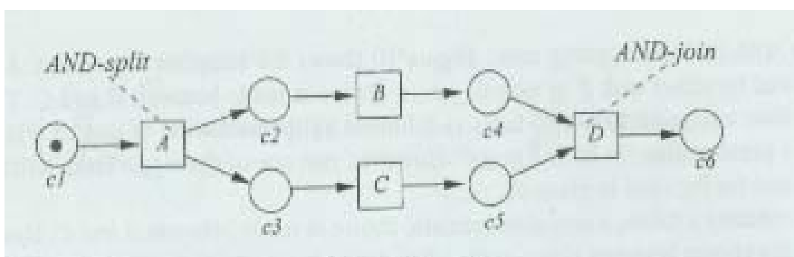

Figure 1. Petri net representation of an AND-Split and AND-Join

The equivalent metagraph representation of the AND-Split would be represented by an edge *A* with an invertex set of information elements given by {*c1*} and an outvertex set of information elements {*c2*, *c3*}. This transformation is consistent with the propositions listed previously. The AND-Join routing construct would consist of edge *D* with an invertex set of information elements given by {*c4*, c5} and an outvertex set of information elements given by {*c6*}.

The second routing constructs we need to study are the Implicit OR-Split and OR-Join with User and Message Triggers as shown in Figure 2:

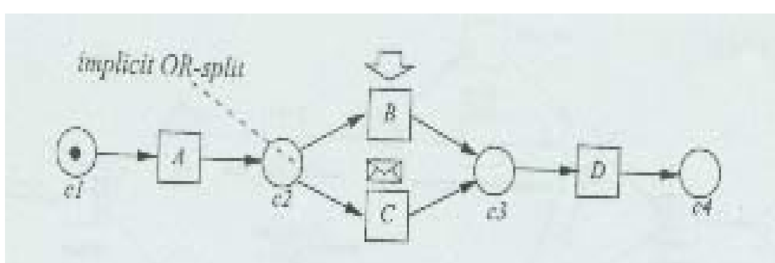

Figure 2. A Petri Net Representation of an Implicit OR-Split and OR-Join with User and Message Triggers

Note that the *message trigger* (the envelope symbol) for Transition C will be represented as an Message information element for edge *C*. Similarly, the *user trigger* (the arrow symbol) for transition *B* is represented as a label on edge *B* in the metagraph representation. We are still working on giving a complete mapping for this routing construct which will be forthcoming in future drafts of this paper.

Finally, we are left with mapping an Explicit OR-Split into a Metagraph representation. An example of a Petri Net representation is given in Figure 3:

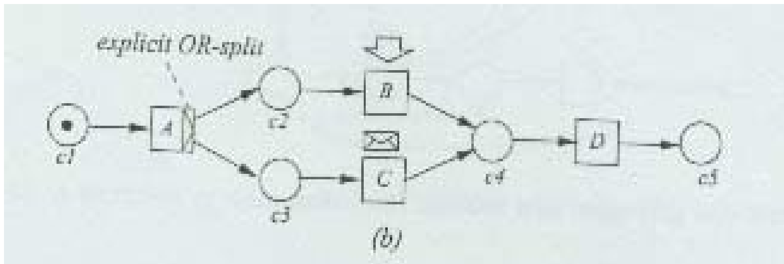

Figure 3. Petri Net Representation of the Explicit-OR Split with User and Message Triggers

We begin by noting that the Explicit OR-Split is a non-deterministic split that is the result of a condition being evaluated once the transition fires and completes. This is equivalent to a condition being evaluated once an edge has been completed. Let the two conditions determining the split be x_1 and x_2. In that case, we need an intermediate step after transition A fires to model the evaluation of the condition. Thus, once transition A fires, we move to a new place called A_Complete. We then evaluate x_1 and x_2 simultaneously. Each evaluation is a separate event and as such an event is equivalent to a transition in a Petri Net (Murata, 1989). Thus, each successful evaluation of the condition is equivalent to an edge. Hence, we have two edges labeled x_1 and x_2 moving out of information element A_Complete. It must be then that c_2 and c_3 are now propositional information elements.[2] Once we have completed the above mappings of the routing constructs, we should have a complete mapping between Petri Nets and Metagraphs.

Contributions and Future Directions

We intend to complete the mapping and illustrate several examples of mappings between the two modeling tools. Once we have finished, we believe there are 4 main contributions of this work. First, we have enabled WMS based on two different modeling tools to be able to inter-operate and understand each other. In an inter-connected virtual world, this is an important problem to solve. Second, we have enabled workflow designers to use the relative strengths of both modeling techniques by easily moving between the two models. Third, we have extended the work of van der Aalst and Kumar (2002) on XRL by both allowing inter-operability of WMS based on differing metamodels and by implicitly providing a map between XRL constructs and metagraphs., given that a mapping is suggested between the proposed XRL and Petri Nets (van der Aalst and Kumar, 2002). Fourth, we have provided a step in the direction of UML since any modeling language designed to model one or the other of the modeling techniques can be implicitly used to design both. Future directions of this paper include extending this mapping to attributed metagraphs and other metamodels used to model workflows.

References

Basu, A. and Blanning, R. (1994a). " Metagraphs: A Tool for Modeling Decision Support Systems." *Management Science*, 40 (12), 1579-1600

Basu, A. and Blanning, R. (1994b). "Model Integration Using Metagraphs." *Information Systems Research*, 5 (3), 195-218.

[2]Similar intuition for modeling the Explicit OR-Split in this way can be derived from Fig. 4 in Basu and Blanning (2002). In that figure, tasks Acceptable Risk Assessment and Marginal Risk Assessment refer to conditions on information element Loan Risk (which is equivalent to information element A_Complete in the example). Similarly, information elements AR and MR are then propositions or assumptions.

Basu, A. and Blanning, R. (1998). "The Analysis of Assumptions in Model Bases Using Metagraphs." *Management Science*, 44 (7), 982-995.

Basu, A. and Blanning, R. (2000). "A Formal Approach to Workflow Analysis." *Information Systems Research*, 11(1), 17-36

Basu, A. and Blanning, R. (2001). "Workflow Analysis Using Attributed Metagraphs." *Proceedings of the 34th Hawai International Conference on System Sciences.*

Basu, A. and Kumar, A. "Research Commentary: Workflow Management Issues in e-Business." *Information Systems Research*, 13 (1), 2002.

Bussler, C. (1999). "Enterprise-Wide Workflow Management." *IEEE Concurrency*, 32-43.

Casati, F. and Shan M. (2001). "Dynamic and adaptive composition of e-services." *Information Systems*, 26, 143-163.

Georgakopoulos, D. and Hornick, M. (1995). " An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure." *Distributed and Parallel Systems*, 118-153.

Jensen, K. (1997). "A Brief Introduction to Colored Petri Nets." *Working Paper University of Aarhus*.

Murata, T. (1989). "Petri Nets: Properties, Analysis and Applications." *Proceedings of the IEEE*, 77 (4), 541-579.

van der Aalst, W. (1998). "The Application of Petri Nets to Workflow Management." *Working Paper Eindhoven University of Technology*.

van der Aalst, W. and Kumar, A. 2002. "XML Based Schema Definition for Support of Inter-organizational Workflow." *Forthcoming in Information Systems Research*.