Proceedings of the 1994 Winter Simulation Conference
ed. J. D. Tew, S. Manivannan, D. A. Sadowski, and A. F. Seila

INTEGRATING BPR WITH IMAGE-BASED WORK FLOW

Robert M. Shapiro

Meta Software Corporation
125 CambridgePark Drive
Cambridge, Massachusetts 02140, U.S.A.

ABSTRACT

Almost every BPR engagement begins with a 'process capture' phase. The BPR team goes to the process owners and with their assistance develops a complete description of how the business is currently conducted. The results of this are used to select those processes within the business which offer the potential for dramatic savings if re-engineered. These may then be modeled in more detail and analyzed, from the point of view of evaluating the benefit. The analysis may require the use of both Activity-Based_Costing tools and simulation.

Ultimately, the implementation of the re-engineered processes may employ a Work flow system. Building this **should be more than guided** by the BPR effort. The process capture, if done with the appropriate software tools, can feed directly into both the BPR analysis (ABC, simulation, etc.) and the work flow implementation. This greatly reduces the cost and time to employ simulation and build the implementation. We describe this approach.

1 INTRODUCTION

In the past few years we have acquired experience using IDEF (Marca and McGowan 1988) diagrams to capture work flows in the following application domains: income tax returns processing, human resources management, sales order processing, command and control, check processing, insurance application processing, insurance claim processing, loan origination processing, engineering design release, project management, production planning and enterprise wide re-engineering.

The main reasons for using IDEF are:

• Use of graphics - IDEF diagrams are created by drawing rectangles that represent activities and by wiring them together with arrows representing input and output relations between them.

• Use of hierarchies - IDEF is a top-down methodology where a process is specified by systematically decomposing higher level activities into more detailed sub diagrams until the meaning (e.g. behavior) of the lowest level activities is sufficiently precise.

• Easiness of use - IDEF is easy to learn since it has very few primitives and makes extensive use of graphics. Most courses that teach how to read IDEF diagrams take half a day. Most courses that teach how to create IDEF models typically take 2-3 days.

• Widespread - IDEF has proven itself mature as a process description language. Tens of thousands of analysts throughout the US and Canada have been trained in its use. The US Government uses it widely, especially for business process analysis. The US Department of Defense has mandated its use for the Corporate Information Management Program. In Europe IDEF is used for the early specification phase of software design, particularly in the aerospace industry and in some of the larger projects (e.g., Columbus) undertaken by the European Space Agency.

IDEF on a laptop works well as a process capture tool. Members of the BPR team can visit the process owners and create together with them an accurate representation of how the organization currently does business: the **as-is** model. Costing information may be included at this stage by using an Activity-Based-Costing glossary associated with the model. Behavioral information may be included by using a further extension of the glossary, combined with techniques for resolving the ambiguities normally contained within IDEF models (Pinci and Shapiro 1993).

IDEF may be translated directly into a variety of simulation languages. Automated bridges have been developed for generating from an IDEF model the behaviorally equivalent **CP-net** (Jensen 1992, Jensen and Rozenberg 1991), **ServiceModel** model or **WITNESS** model. Previous papers (Shapiro, Pinci and Mameli 1993, Pinci and Shapiro 1991) describe the translation of IDEF diagrams into CP-nets both conceptually and from the point of view of implementation. In this paper we focus on the bridge to **ServiceModel**.

Historically, IDEF (or some other less formal method) has been employed for process capture, without the capability of going automatically to simulation. Thus, a simulation expert needed to take the process capture model and program from scratch the simulation. This disconnect between process capture and simulation has obvious consequences in terms of cost and time. The very same issues arise in moving from a BPR effort to a Work Flow implementation.

We use a set of conventions for modeling work flows in IDEF0. A computer program (Meta Software Corporation 1993) deduces the behavioral details automatically for models following these conventions. This eliminates the need to write (and debug!!) the simulation model. It is our hypothesis that many work flow problems can be described using this approach. In any case, the model generated by the program provides the starting point for further elaboration or modification.

In our approach, a Work Flow model describes how 'Entities' flow through an organization. The organization consists of interrelated activities which require resources for their execution. The resources are typically various categories of staff and equipment. Input files characterize the staff and equipment available for a simulation, as well as the time-ordered set of 'Entities' to be processed. The results of simulation are a set of reports which evaluate the performance of the organization and point to bottlenecks (delays) in processing as well as inefficient use of resources (idle resources) . If the simulator has animation (e.g. ServiceModel) the ongoing simulation may be observed dynamically.

The IDEF model also provides the routing information required by a Work Flow system. Thus it can be used to automatically generate the flow logic (often called maps or sheets) that drive entities through the work stations of the system. Suitably extended to include the data structures (forms and fields) required in the process, the IDEF model provides data object descriptions that allow application programs running on clients seamless access to repository information coordinated or controlled by the work flow server.

The remainder of the paper is organized in the following way. In section 2 we present the basic constructs in IDEF for specifying behavior. In section 3 we discuss the bridge from IDEF to ServiceModel(see Pinci and Shapiro 1993 for a description of the bridge to Design/CPN). In section 4 we discuss the potential bridge to several work flow products. Finally, in the last section we discuss future work and draw some conclusions.

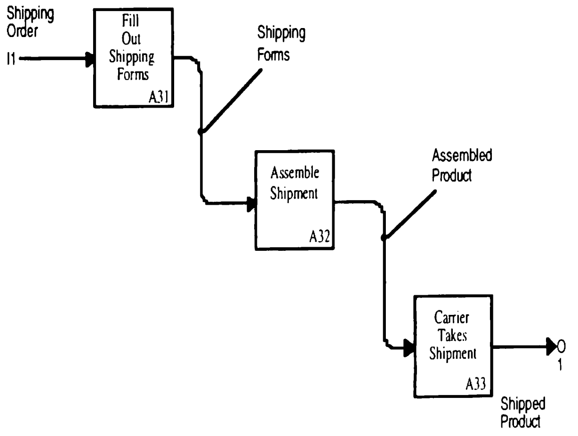
2 BASIC CONSTRUCTS

The basic paradigm for understanding the behavior of an IDEF activity is as follows. When all the inputs required by an activity are present and all the resources required are available, the activity consumes the inputs, producing outputs based on the inputs and delaying the availability of the outputs and resources according to the time required for the activity to be performed.

The Simulation Clock advances only when there is nothing more that can happen at that time. This means that an activity takes place repeatedly so long as there are inputs present and resources available. This interpretation of behavior allows true concurrency in work flows. Sequentialization is a consequence of constraints.

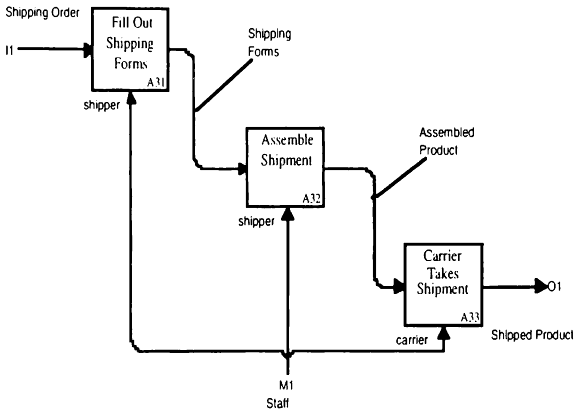
In what follows we present a series of IDEF diagrams that illustrate the rudiments of behavior for work flow models. The diagrams are from a description of a simple Sales Order Processing model. Our objective here is to convey an intuitive understanding of the relationship between IDEF diagrams and their behavioral semantics.

First we examine a simple case: a sequence of activities with **one input and one output**.



Each order flows through a sequence of three steps. The time to ship an order is the sum of the times of the three steps. Since there are no resources required, orders flow through unconstrained by resource limitations. A million orders can be processed in the same time it takes to process one order.

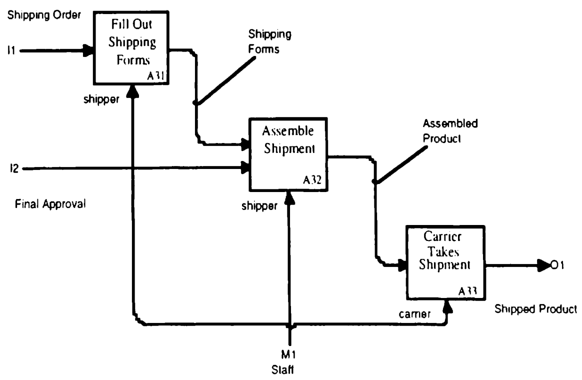
Now we add a **resource**.



Each activity is constrained by the availability of staff. If there is only one shipper, the shipping forms can be filled out for only one order at a time. The same shipper cannot fill out a form and assemble a shipment at the same time. Orders may now be delayed because staff is not available. On the other hand, staff may be idle because there are no orders to ship.

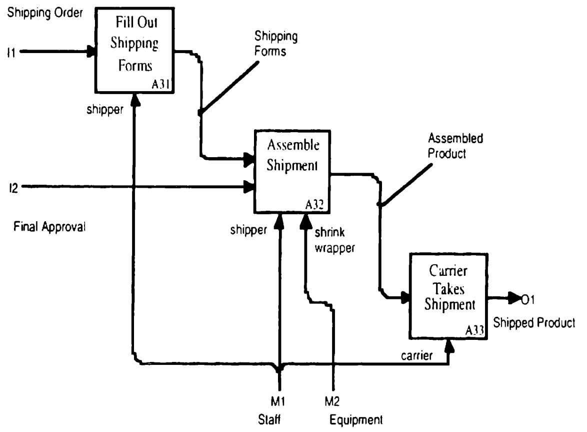
The <u>time to service</u> an order is determined by the difference between its arrival time and its shipping time. This may become large if a lot of orders arrive at the same time. The <u>processing time</u> for an order is the sum of the times of each step it goes through. This remains the same in this model irrespective of loading or staffing. The <u>delay time</u> for an order is the total delay it experiences. In this simple case <u>time to service</u> = <u>processing time</u> + <u>delay time</u>. Where multiple parts of the same order can be processed concurrently, this relationship no longer holds.

There can be **multiple** inputs.



Assemble Shipment may now be delayed for another reason. Both the Shipping Form and Final Approval are required inputs. Thus, processing delays can occur because multiple inputs arrive at different times. The critical path is the longest, the critical input is the latest.

There can be **multiple** resources.



Now Assemble Shipment requires two resources: a staff person who is a shipper and a piece of equipment, the shrink wrapper. This activity cannot take place until both are available. Various causes of processing delays are possible for this activity. The lack of one resource may force the other resource to be idle.

Activities can generate **multiple outputs**. This allows concurrent processing of different parts or copies of an order (i.e.. entity).

The complete behavioral functionality includes:

- Complex Constraints
- Calculation of Duration Time
- Calculation of Size
- Conditional Outputs
- Output Transmission Time
- Priorities
- Branching and Joining

The reader is referred to (Pinci and Shapiro 1993) for a discussion of these features.

3 BRIDGE TO SERVICEMODEL

The content of the IDEF model is used to determine the ServiceModel that embodies the behavior. We refer the reader to (ProModel Corporation 1994) for a detailed description of ServiceModel. Design/IDEF 3.1P1 and ServiceModel 2.0, available fall 1994, implement the bridge.

3.1 Model Structure

An IDEF0 model is a hierarchic collection of sub models. Each sub model consists of:

1: a small number of activities, some of which refer to other sub models (decompositions).
2: a well defined interface to a higher level model (the ICOM ports).
3: an arrow structure which interconnects the activities and ports in a syntactically correct manner.

The top level sub model (the A-0 page) refers to inputs, controls, outputs and mechanisms which lie outside the model. These play a special role which we describe in the sequel.

Behavioral detail is specified for leaf (non decomposed) activities. Only these activities will appear in the *flat* ServiceModel model. The arrow structure(except for mechanisms: see sequel) defines the paths for objects(entities) flowing between activities. The general branch/join arrow structure of IDEF0 is restricted in behavioral modeling (see WFA documentation) so that all joins must precede all branches. This guarantees, for each arrow structure, the existence of a common segment. Each such common segment can be thought of as a **place**(buffer, queue) where entities reside while traveling between activities. The software indicates where all such places will be From this we have:

1: every leaf activity is a ServiceModel **LOCATION**. Associated with each location will be **Operation logic** specific to the activity that transforms the input data. See sequel. Each activity/location has a name based on the IDEF box number and the activity text.

2: every place is a ServiceModel **LOCATION.** Associated with each location will be **Operation logic** for handling the branch/join arrow structure for that location. See sequel. Place names are derived from IDEF labels, but the translator guarantees by construction that they are unique and do not match any ServiceModel reserved words.

Activity/Locations are the active elements in simulation. They change the values of the attributes associated with the entities that pass through them. They make probabilistic choices of what routes to follow. The arrow structure provides the information for interconnecting Activity/Locations and Place/Locations. IDEF0 arrow structures are ambiguous in respect to issues of fan in and fan out. These are resolved in WFA using the WFA **Fan Info** dialog. From this we have

1: the arrow structure determines the **Routes** between locations, the **routing rules** and routing **exit logic.**

In the sequel we discuss each of these in detail.

3.2 Role of Mechanisms

In WFA mechanisms are treated in a special way. They are thought of as the resources needed in order to perform the activities. Resources are required at the beginning of an activity and are unavailable for other activities while in use. Thus a mechanism arrow expresses the use of a specific resource each time that activity takes place, followed by the return of that resource to the resource pool upon completion of the activity (note that the back arrow is not required or allowed).

Each mechanism on the A-0 page defines a resource category. Each activity that requires resources connects to one or more of these mechanisms via the arrow structure. Attached labels on mechanism arrows define resource subtypes. The **WFA DATA** dialog allows inspection of category and subtype data.

1: each **subtype** maps to a ServiceModel **RESOURCE**. The default quantity is one. Within ServiceModel quantities, **SHIFTS** and other resource related information may be edited as usual.

All mechanisms attached to a leaf activity cause the associated activity/location to contain in its operation logic a **JOINTLY GET** of the designated resources followed eventually by a **FREE ALL.** This logic can be extended by the user to include Boolean logic, quantities greater than 1 and priorities

3.3 Role of Inputs and Controls

In WFA inputs and controls are treated almost identically. Each input and control on the IDEF A-0 page defines a source for objects that flow through the model during simulation. Thus:

1: each top level input or control maps to a ServiceModel **ARRIVAL**. (The user may prefer to provide arrival information using **ARRIVAL FILES**).The name of the location is determined by the top level label. The entity is chosen from the list of subtypes defined in the IDEF model. All data values (Qty each, First Time, Occurrences, Frequency) are set by default to one The user may edit these in the customary manner.

2: each arrival statement feeds entities to a corresponding place/location, which contains in its operation logic, the generation and assignment of a unique value to the **id** attribute of the entity (see sequel).

There is only one circumstance in which inputs and controls function differently. The I1 input to an activity is regarded as the **dominant input** and all other inputs and controls are subordinate to it. In case there are no inputs, the C1 input plays this role. We discuss this in the sequel.

3.4 Role of Subtypes

The **WFA DATA** dialog allows the user to introduce subtypes associated with the objects flowing on input, output and control arrows. Subtypes play a role in steering objects through the model.

1. each subtype corresponds to a ServiceModel **ENTITY name**. The presence of a specified subtype for a particular input or control of an activity is reflected in the corresponding operation logic for the place/location that supplies that particular input or control. The operation logic tests to make sure the Entity matches the specified subtype All Entities have variable attributes described in the sequel.

3.5 Display Information

The graphical layout of the hierarchical IDEF0 model is of little value in the flat world of ServiceModel models. Instead, the logical structure of the model is used to determine a default layout of the locations for ServiceModel . This layout is made visible in Design/IDEF and may be printed out. (Future releases of the product will allow rearrangement of the layout within the IDEF tool.) The layout uses heuristics. The layout determines the positions of the locations and the routes between them. These may be altered within ServiceModel in the usual way.

Activity/locations are depicted as rectangles. The text within is created by taking the activity box ID and concatenating to it the text within the activity box.

Place/locations are depicted as ellipses. The text within is the label text.

In both cases, all non-alphanumeric characters are replaced by underscore (_).

3.6 ServiceModel Details

The IDEF model provides detailed information for the following ServiceModel constructs:

- Attributes(Entity) and Variables
- Locations: Icon, Name.
- Entities: Icon, Name.
- Resources: Icon, Name.
- Process Information: Entity, Location, Operation Logic, Routing Table.

For a complete description the reader is referred to (Shapiro 1994).

4 BRIDGE TO Work Flow

The content of the IDEF model is used to generate the routing of jobs (work objects) through the tasks (work flow performers) required to accomplish the intended purpose of the system. Explicit representation of forms and fields on forms required as inputs and/or outputs of activities provides definition of the data objects that must be stored in the work flow repository.

4.1 Candidate Work Flow Systems

The following work flow systems have been studied in detail:

- Wang Open Workflow
- Plexus FloWare
- XSOFT InConcert
- FileNet Visual Workflo

Each of these systems support hierarchical routing specifications. One (FloWare) focuses on routing and provides no management of data structures and access paths to the repository. Several of them are limited in terms of the ability to support concurrent work streams within a job, with rendezvous coordination required within the same sub model as the fork or split. Only InConcert offers functionality sufficient for completely handling the input/output relationships expressible in IDEF.

4.2 Interpretation of Resources

The staff resources in these work flow systems are typically people logged in on clients running the work flow operating software. The roles these people can play determine in part what work will be routed to them. The 'in-basket' of persons in the same category may list the same jobs. As soon as a person selects a job it disappears from all in-baskets.

Special equipment may also be characterized as resources. Work flow performers that do not require staff (e.g. printers) operate automatically off of shared queues.

5 CONCLUSIONS

The Work Flow Analyzer greatly reduces the time involved in creating a simulation model suitable for identifying bottlenecks and idle resources in a complex work flow. It accomplishes this by assigning to the activities of an IDEF model a simple behavioral interpretation, based on the arrow structure in IDEF and the behavioral semantics of transitions in hierarchical Colored Petri Nets.

This makes it possible for a computer program to generate, from the IDEF model, the entire structure of the simulation model, including all inscriptions and the additional logic required to load input files.

As a result. a much broader group of people will now be able to build models to study the performance characteristics of work flow systems.

It remains to be seen whether the specific interpretation of behavior is general enough to handle a high percentage of work flow problems. Our objective in the near future is to extend the functionality based on the experiences of early users of the approach.

A natural extension of this effort is to automate the construction of a work flow system that implements the business process described by the IDEF models created during the process capture phase of Business Process Re-engineering.

Tuning a Work Flow system can then be accomplished in the following way. Duration and frequency data collected by the logging mechanism in the Work Flow operating system are exported to the simulation model. What-if experiments then examine the consequences of changing staffing levels and schedules, equipment speeds, and business rules. The Work Flow system is changed accordingly. Continuous process improvement becomes a reality.

REFERENCES

Jensen, K. 1991. Coloured Petri Nets: A High-level Language for System Design and Analysis. In: *Advances in Petri Nets 1990*, ed. G. Rozenberg, 342-416. Lecture Notes in Computer Science Vol. 483, New York: Springer-Verlag.

Jensen, K. 1992. *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use. Volume 1: Basic Concepts*. EATCS Monographs on Theoretical Computer Science. New York: Springer-Verlag

Jensen, K. and Rozenberg, G. (eds.). 1991. *High-level Petri Nets. Theory and Application*. New York: Springer-Verlag.

Marca, D. A., and McGowan, C. L. 1988. *SADT*. New York: McGraw-Hill.

Meta Software Corporation. 1992a. *Design/IDEF User's Manual*. Cambridge, Massachusetts: Meta Software Corporation.

Meta Software Corporation. 1992b. *Design/CPN User's Manual*. Cambridge, Massachusetts: Meta Software Corporation.

Meta Software Corporation. 1993. *Work Flow Analyzer Tutorial*. Cambridge, Massachusetts: Meta Software Corporation.

Pinci, V. O., and Shapiro, R. M. 1991. An Integrated Software Development Methodology Based on Hierarchical Colored Petri Nets. In *Advances in Petri Nets 1991*, ed. G. Rozenberg, 227-252. Lecture Notes in Computer Science Vol. 524. New York: Springer-Verlag.

Pinci, V. O., and Shapiro, R. M. 1993. Work Flow Analysis In Proceedings of the 1993 Winter Simulation Conference, ed. G. W. Evans, M. Mollaghasemi, E. C. Russell, W. E. Biles, 1122-1130.

ProModel Corporation 1994. ServiceModel for Windows: User's Guide and Reference Manual.

Shapiro, R. M., Pinci, V. O., and Mameli, R. 1993. Modeling a NORAD Command Post Using SADT and Colored Petri Nets. In *Functional Programming, Simulation and Automated Reasoning,* Lecture Notes in Computer Science. New York: Springer-Verlag.

Shapiro, R. M. 1994. The Relationship between Work Flow Analysis Models and ServiceModel. Design/IDEF3.1P1 . Cambridge, Massachusetts: Meta Software Corporation.

AUTHOR BIOGRAPHIES

ROBERT M. SHAPIRO is Chairman of the Board, CEO, and Founder of Meta Software Corporation, a privately held company that develops, sells and supports modeling and simulation tools for business process re-engineering. A 36 year veteran of the information systems industry, he has pioneered the development of better and faster computer-based simulation using Petri net technology. He has had a number of his writings published in books and journals.