# Workflow Modeling for Implementing Complex, CAD-Based, Design Methodologies

J. Stavash and J. Wedgwood
Lockheed Martin Advanced Technology Laboratories
Camden, NJ
jstavash@atl.ge.com, jwedgwoo@atl.ge.com

M. Forte
Rockwell International Corporation
Seal Beach, CA
mjforte@naa.rockwell.com

W. Selvidge and M.C. Tuck
Intergraph Corporation
Huntsville, AL
wrselvid@ingr.com, mctuck@ingr.com

A. Bard
Army Research Laboratory
Fort Monmouth, NJ
abard@ftmon.arl.mil

E. Finnie
Aspect Development Corporation
Mountain View, CA
elisa@aspectdv.com

**Abstract**

*A specific goal of the Rapid Prototyping of Application-Specific Signal Processors (RASSP) program is to achieve a 4X improvement in design cycle time, cost of the design, and the quality of the final product. In order to achieve these 4X improvements, Lockheed Martin Advanced Technology Laboratories (ATL) is developing design methodologies that make use of concurrent engineering, design object reuse, and the spiral model of development for rapid prototyping [1]. Lockheed Martin ATL, Rockwell International, Intergraph, and Aspect Development are collaborating on developing workflow models to implement these design methodologies, as well as the data management and CAD tool environment to support them.*

## 1: Introduction

The RASSP team is implementing an integrated workflow/data management development environment that will provide the infrastructure to enable the application of the spiral model and to support reuse. The team's workflows define the steps for a particular design process; tools for execution of each step; personnel resources required; and design objects produced/consumed at each step. Furthermore, the workflows identify tasks that can be executed concurrently. The use of workflows — tightly coupled with design data management, process management, and CAD development tools — is crucial to effective management of the evolving designs (both process and data) and to achieving the full benefit of the RASSP design methodology. It effectively removes the designer from the details of data availability, translation, and tool invocation, which greatly increases productivity.

## 2: Workflow Development

The team is developing workflows for the entire RASSP design process using the IDEF3X modeling method. Rockwell International Corporation developed this method, which is an extension of the IDEF3 process description capture method. The name IDEF originated from the Air Force program for Integrated Computer-Aided Manufacturing (ICAM), from which the first ICAM Definition (IDEF) methods emerged. It is now called Integration Definition. IDEF3 was specifically created to capture descriptions of sequences of activities. It can be distinguished from other process modeling methods because it facilitates the capture of the description of what a system actually does [2].

### 2.1: IDEF3X overview

The IDEF3X modeling method combines the Input, Con-

trol, Output, Mechanism (ICOM) aspect of IDEF0 with the process flow description of IDEF3, along with some additional features to aid implementation by a workflow management tool, such as Intergraph's Design Methodology Manager (DMM). The syntactic elements of an IDEF3X model are similar to IDEF3 and include units of behavior (UOBs), junction boxes, and precedence links. Additional features include:

- Identifying ICOMs by naming the object and its life-cycle state separated by an asterisk (e.g., *Draft*Publication*, where *Publication* is the name of the object and *Draft* is its current state)
- Object state links, which identify the flow of data between UOBs
- Feedback links, which indicate fail-back paths
- Annotating the name of the junction boxes with an *A* or *S* to indicate an asynchronous or synchronous junction
- Annotating the precedence links with a *P:* followed by a two-letter code that indicates the precedence between the parent and dependent UOB.

Precedence links within an IDEF3 process model imply a default precedence of Finish-Start, which means that the parent UOB must finish before the dependent UOB can start. IDEF3X supports additional precedence relationships such as Start-Start, Start-Finish, Finish-Finish, Concurrent, Cascade, Fail Reset, and Fail Cascade. All of these precedence types are supported by Intergraph's DMM tool [3].

## 3: Workflow Implementation

The RASSP team captures the methodology by codifying the hierarchical process steps into workflows, which are implemented in a workflow management tool, such as using Intergraph's DMM. The workflows, in conjunction with Intergraph's Product Data Manager (PDM), DM2.0, and the RASSP Reuse Data Management System (RRDM) from Aspect, Inc., guide the designer through the design process, providing access to the appropriate tools at the proper times, as well as providing the right data in the right format for those tools.

In order to understand the workflows, it is important to understand the level at which the process/data management is implemented. The team has developed workflows for the three main RASSP design processes: System Definition; Architecture Definition; and Detailed Design. System Definition consists of the following subprocesses: System Requirements Analysis and Refinement; Functional Analysis; and System Partitioning. Architecture Definition consists of the following subprocesses: Functional Design; Architecture Selection; and Architecture Verification. Detailed Design consists of the following subprocesses: Chassis Design; Backplane Design; Module Design; ASIC Design; FPGA Design; and Integration and Test [4]. Each of these processes is broken down into the leaf-level workflows. An example of a leaf-level flow is the Field-Programmable Gate Array (FPGA) Design, which is described in Section 4. The process is managed at a higher level than traditional workflows, which derive the process blocks based on tools. In the RASSP workflows, several tools may be available in one particular process step. The workflows are an implementation of the RASSP methodology and can be considered to be tool-independent.

At the onset of a project, the project manager will identify which design processes are required and build the project workflow from leaf-level workflows. The workflows specify the process flow and control precedence. A data template is defined for each process step based on the specified Inputs, Outputs, and Controls. The data template is completed with names of actual data objects that will be managed by the PDM when the workflow is instantiated. An authorization template is defined based on the specified resource Mechanisms. The authorization template is completed with the names of the personnel resources assigned to each process step. A set of workspaces is allocated for supporting the project. Each process step also has pre- and post-conditions associated with it. These conditions identify the data translations required for the data in and out of the process step and may provide automatic E-mail to the responsible resource associated with the task or any other required task that can be scripted to be executed when the task is started or completed. The final piece of information associated with the process step is the tool launch script. The tools required for the process step are specified by the tool Mechanisms. The launch scripts and the data/authorization template information customize the workflows for a particular design environment.

Once a workflow has been developed and the data and authorization templates defined, the correspondence between the process steps and actual tools needs to be reconciled. Some tools may encompass multiple process steps, while others may perform a portion of one process step. From a workflow standpoint, the latter case is easy to deal with because any number of tools can be launched from within one process step. The difficulty arises in the former case, when one vendor's tool spans several workflow process steps and may even span from one hierarchical workflow to another. There are currently three methods of handling this situation:

1. The first method is to modify the workflow so that the multiple process steps, which are encompassed by the vendor's tool, become one. This has obvious drawbacks because it is contrary to the concept of tool-independent workflows.

2. A second approach is for the vendor to modify the tool to match the workflow. This is a potentially expensive and time-consuming process, and possibly not completely achievable.
3. A third approach is to run the vendor tool as usual and have the tool flag the workflow to tell it when the corresponding step in the workflow is complete. That workflow step would then transition appropriately, depending on the exit status passed to it from the vendor tool. The workflow would return a message to the vendor's tool indicating which tasks could now be executed. With this approach, the workflow remains tool-independent. However, the tool-workflow communication is dependent on the process manager. In order for this to be a viable solution, workflow interoperability standards would have to define the communication mechanism.

## 4: Modeling Example

The example discussed in this section is taken from the RASSP Detailed Design workflows. Figure 1 contains a portion of the FPGA design workflow represented as an IDEF3X model. This model contains two UOBs that represent design activities: Functional Design and Functional Verification. Precedence links, object state links, junction boxes, and all of the ICOMs for each UOB are also present. This model was developed using the TopDown Flowcharter tool by Kaetron Software Corporation.

The workflow begins with Functional Design. The purpose of this activity is to refine an FPGA behavioral model using VHDL, with sufficient detail so that it can be synthesized down to the specific FPGA programmable function unit (PFU) level. The inputs for this activity are *Preliminary*FPGA Model* and *Generated*FPGA Requirements Specification*. The control for this activity *is C:Released*RASSP Reuse Library* and the output is *Refined*FPGA Model*. The three mechanisms are:

- *M:Released*Summit VisualHDL*
- *M:Released*Aspect Explore CIS*
- *M:Qualified*FPGA Design Engineer*.

Notice that the FPGA Model is both an input and output object, but its state has changed from *Preliminary* to *Refined*. This reflects the processing performed to refine it from the behavioral level of abstraction down to the register-transfer level (RTL) of abstraction in preparation for synthesis. The exclusive *OR (X)* junction box to the left of this activity indicates its precedence timing. This activity may be executed initially via the *Begin* link or repeatedly via the *P:FR* feedback link from the *Functional Verification* UOB.

The next activity to be performed is *Functional Verification*. This is indicated by the *P:FS* precedence link between the two UOBs. This type of precedence link indicates that the parent activity, *Functional Design*, must finish before the dependent activity, *Functional Verification*, can start. The purpose of this activity is to verify that the FPGA Model meets its performance requirements. This is accomplished by performing a functional simulation. The input for this activity is *Refined*FPGA Model*. The controls are:

- *C:Generated*FPGA Testbench*
- *C:Defined*Target Tester Characteristics*.

The output is *Verified*FPGA Model,* and the mechanisms are:

- *M:Qualified*FPGA Design Engineer*
- *M:Released*MGC QuickVHDL*
- *M:Qualified*WAVES Site Expert*.

Once again the FPGA Model is both an input and output object, but this time its state has changed from *Refined* to *Verified*. If the results of performing a functional simulation indicate the model meets the requirements, the model

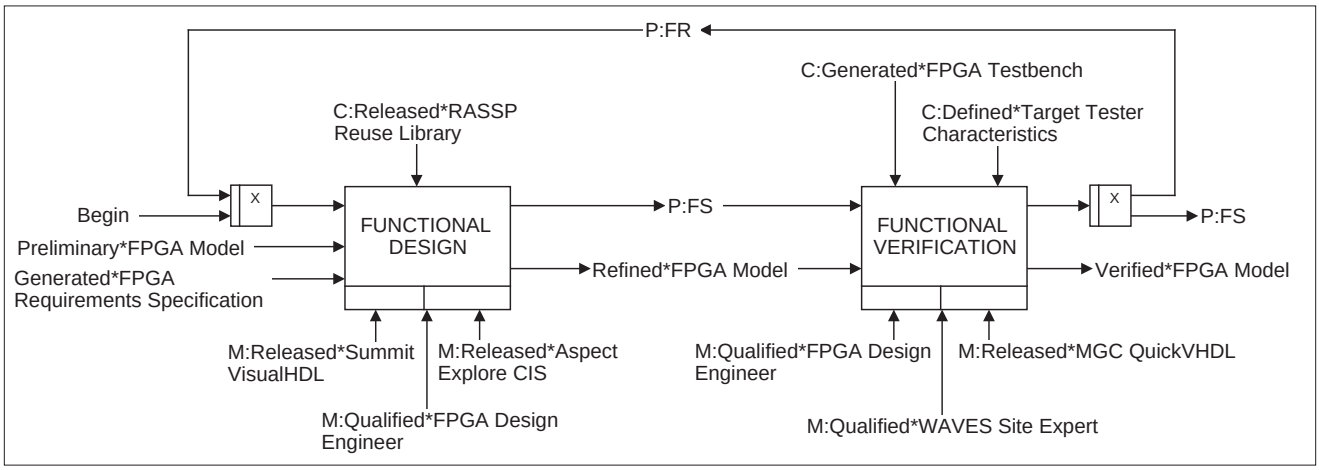


**Figure 1. The IDEF3X workflow model.**

is made available to the synthesis activity (not shown). The *X* junction box to the right of this activity again indicates precedence timing. The remaining activities of this workflow may be executed via the *P:FS* precedence link or the *Functional Design* activity may be executed again via the *P:FR* feedback link. This will occur when *Functional Verification* has not completed successfully and the FPGA model needs to be modified. It is important to note that if the *Functional Design* activity is re-executed, the *Functional Verification* activity will also be re-executed.

## 5: Proposed Features of a Workflow Modeling Language/Tool

A goal of a workflow description language is to represent complex process and data flow in such a way as to enable automatic implementation of the workflow into a process/data manager. This is not possible today for several reasons. There are incompatibilities between workflow modeling methods and workflow management tools that limit the accuracy to which a process can be modeled. The current state of workflow modeling using IDEF3X and commercially available workflow management tools does not fully support a dynamic workflow environment. Below are some examples of the process flows required to implement the RASSP Methodology, which were either difficult to model, difficult to implement, or both. Finally, some project management functions, when combined with the workflows and the workflow manager, can provide a powerful integrated approach to project control and tracking — significantly improving the ability to accurately plan and track projects, as well as to quickly identify and resolve potential bottlenecks.

### 5.1: Model and implementation incompatibilities

The incompatibilities between today's workflow modeling tools and process management tools fall into two classes. The first is the lack of a standard means to represent complex process flows. The second lies with the current state of process manager design. Even in those cases where complex process control can be modeled, it is often not possible to accurately implement it in today's process managers. The RASSP Methodology requires more flexible process dependencies than are available from the standard workflow modeling constructs. Considerable flexibility can be gained by allowing junction boxes to be connected to each other. This can be modeled in IDEF3X, but it cannot be implemented in DMM. On the other hand, the concept of concurrency in the RASSP environment causes difficulties with 'persistent' process blocks, which start in the middle of one workflow but do not need to be finished until the middle of a subsequent workflow. This creates multiple entry points and exit points out of the higher level workflow. Additionally, it is not compatible with IDEF3X, although it may be implementable in a process flow manager.

### 5.2: The dynamic workflow environment

Certain aspects of the RASSP Methodology require a dynamic workflow environment. In order to achieve a 4X improvement, designers need to be able to explore variations on a single implementation of a design, as well as vastly differing implementations. This means that after a designer completes a workflow and creates a valid output data set to pass on to the next workflow, the designer needs to be able to conditionally start an identical workflow to the one just completed. In this new workflow, either the design that was just created would be modified (variations on a theme) or an entirely new data output set would be created (new theme). The creation of the duplicate workflow and the management of the data associated with that workflow should be transparent to the designer. This scenario can not be modeled in IDEF3X nor can it be dynamically implemented in DMM.

A common construct in the workflows is a group of parallel analysis tasks that occur after a major design process. The parallel tasks feed a *select and merge* task, which examines the results of the parallel tasks. If one of the parallel tasks finds that the current design will not meet the requirements, all of the analysis tasks need to be halted, and the workflow would need to be restarted at a place determined by the responsible resource. If all of the parallel tasks are successful, the results are merged and the design is marked as a candidate for more detailed design. The *select and merge* task can begin as soon as any one of the analysis tasks is completed. However, it cannot complete until all of the analysis tasks are completed. The requirement that the failure of one parallel task causes other tasks to abort is difficult to model and to implement. The data file merging is called out in the activity definitions but is not visually modeled.

Optional processes in a concurrent environment are also difficult to model and implement accurately. In the case of an optional process that is concurrent with one or more required process steps, the aim is to follow all of the required paths, and possibly the optional one. A variation on this is the *floating* process step. This is a process that is always available during the execution of a workflow from which there may be no output to any other block in the workflow, such as a block that allows the designer to file a trouble report with the project engineer. In either case, if the optional or floating block is modeled in IDEF3X as a path out of an *OR* junction box, then it is possible to complete the workflow merely by executing this task and not the main

process flow, which is not desirable. Another dynamic construct that would more richly support the fallback concept of the RASSP Methodology and provide critical project management information is the ability to execute a process block differently based on how it was reached. In other words, was the task started because a previous task finished, or was it the result of a failure? Although this can be represented in IDEF0, when an IDEF3X representation is extracted from the IDEF0 model, only one path through the workflow can be shown.

### 5.3: Expansion to project management

The usefulness of the workflows can be significantly enhanced by linking process management tools with project management tools. The ability to run a hierarchical workflow at different levels of the hierarchy will enhance project planning. Higher level simulations can be performed using currently available tools such as ProSim and Witness from AT&T. During the project planning stage, a high-level workflow could be constructed with statistical data culled from the reuse library to provide good estimates of schedules, costs, and manpower. Workflow process steps would be mapped to the Work Breakdown Structure (WBS) of a contract. Transitioning key process steps would automatically notify project management tools that a WBS had been completed. The management and design engineers could experiment with different design scenarios, simulate and evaluate them in a closed-loop environment. Schedules would more closely resemble real life, with planned iterations and fail-paths, which are not represented in today's project planning tools.

The issues identified above are being actively addressed by the Enterprise System team to provide both the tool-independent approach to modeling processes and workflow management tools that handle the dynamic nature of prototyping processes.

## 6: Summary and Conclusions

In this paper, we have presented workflow modeling as a means for implementing complex CAD-based design methodologies and identified the benefits of its use. These benefits include establishing a repeatable design process, enabling concurrent engineering, and removing the designer from the details of data availability, translation, and tool invocation. We identified IDEF3X as the modeling method and described an example from the RASSP detailed design workflows. We identified problems with IDEF3X and the features of an ideal methodology and tool for process modeling.

Using workflows and workflow management tools to implement the highly concurrent, spiral design methodologies required for the RASSP program presents major challenges in both modeling and implementation. The modeling capabilities of IDEF3X need to be enhanced to provide for representation of more flexible workflows. Workflow managers need to be able to implement these enhancements, as well as to interface with project management tools to provide a completely integrated product, process, and project management environment. We are actively working to provide these advanced capabilities on the ATL RASSP program.

## References


1. Boehm, Barry W., "A Spiral Model of Software Development and Enhancement,„ *Software Engineering Project Management,* IEEE Press, 1987, pp. 128-142.

2. Mayer, Richard J., Cullinane, Thomas, deWitte, Paula S., Knappenberger, William B., Perakath, Benjamin, Wells, M. Sue, IDEF3 Process Description Capture Method Report, Knowledge Based Systems, Incorporated, May 1992.

3. Intergraph Corporation, Design Methodology Manager User's Guide for Unix, April 1995, pp. 41-43.

4. Martin Marietta, RASSP Methodology, Version 1.0, December 1994.