

Vela Project

Globally Distributed Microsystem Design – Proof of Concept: Demo Summaries of the 1998 DAC Exhibition in the University Booth

June 1998

1998-TR@CBL-07-DACdemo, Version 1.0

Dimitri Antoniadis, <http://www-mtl.mit.edu/bin/user?daa>

Franc Brglez, Editor, <http://www.cbl.ncsu.edu/~brglez/>

Anantha Chandrakasan, <http://www-mtl.mit.edu/~anantha/index.html>

Wayne W.-M. Dai, <http://www.cse.ucsc.edu/personnel/faculty/dai.html>

Giovanni De Micheli, <http://akebono.stanford.edu/users/nanni/>

Paul Losleben, <http://www-snf.stanford.edu/staff/losleben/>

Teresa H. Meng, <http://havefun.Stanford.edu/~meng/>

Richard Newton, <http://www-cad.eecs.Berkeley.edu/~newton/>

Jan M. Rabaey, <http://infopad.eecs.Berkeley.edu/~jan/>

Robert Reese, <http://www.erc.msstate.edu/~reese/index.html>

Reprinted, with permission, from
<http://www.cbl.ncsu.edu/publications/>

Publications at this site are occasionally revised,
please check for the latest version under the same title.

For more information about CBL, visit
<http://www.cbl.ncsu.edu/>

or write to
info@cbl.ncsu.edu

For a companion viewgraph document, see
http://www.cbl.ncsu.edu/publications_misc/1998-Talk-DACdemo

About This Document

Acknowledgments. The work presented in this document has been supported by by contracts from the DARPA/ARO (P-3316-EL/DAAH04-94-G-2080), and a grant from Semiconductor Research Corporation.

Abstract – *The Vela Project brings together a team of co-PIs that spans 6 university locations across the USA: MIT, MSU, NCSU, Stanford, UCB and UCSB – all participants in a unique proof-of-concept experiment that aligns system designers, library and tool providers, and system integrators. The overriding objective of this experiment is to design a single-chip multimedia processor at a level of complexity that will be typical for next-generation systems and demonstrate the feasibility of constructing and supporting a distributed design environment.*

The near-term objectives of this project include organization of real-time demos of distributed design activities, modelled on the Internet-based patterns of use that are beginning to emerge among all project participants. These demos have taken place during the forthcoming Design Automation Conference in San Francisco, in the the University Booth, on June 15, 16, 17, 1998. The main objectives of this report are:

- *to document the formal structures of interaction between the project participants;*
- *to provide a concise summary and context for each demo;*
- *to capture enough detail in the description of each executable demo, so that demos can be critically tested and improved by Vela project participants as well as collaborating partners elsewhere.*

Demos will all aspects of this project, including (1) design driver, (2) design libraries, (3) web-based point-tools and APIs, and (4) web-based workflows, enabling users to configure and visualize executable graphs that link tools and data across the Web, while maintaining the consistency and the ease of execution for each and every sequence of user-configured tasks.

Keywords: distributed design, web-based design environments, APIs, design of experiments, benchmarking.

Note. This document has been published for viewing on the Web in PostScript and HTML formats. The latter is annotated with active hyperlinks for each displayed URL, thus facilitating quick browsing of this and related documents. To access the HTML document, see <http://www.cbl.ncsu.edu/publications/1998-TR@CBL-07-DACdemo/> and click on [HTML](#). For a companion viewgraph set, see http://www.cbl.ncsu.edu/publications_misc/1998-Talk-DACdemo/

Citation. If you choose to cite this report, please add the following entry to your bibliography database:

```
@techreport{1998-TR@CBL-07-DACdemo,
author = "F. Brglez (Editor)",
title = "{Vela Project -- Globally Distributed Microsystem Design --
Proof of Concept: Demo Summaries of the
1998 DAC Exhibition in the University Booth}",
institution = "{CBL, CS Dept., NCSU, Box 7550, Raleigh, NC 27695}",
number = "1998-TR@CBL-07-DACdemo, Version~1.0 ",
month = "June",
year = "1998",
note = "{Also available at {\tt http://www.cbl.ncsu.edu/publications}}"}
}
```

Contents

1	Introduction	1
2	Vela Project Navigation Rules and Data Structures	2
3	Project Status and Demo Summaries	7
3.1	Design Driver and Web-Based Tools: Demo Summary	7
3.2	Distributed Design Libraries and Services: Demo Summary	9
3.3	Web-Based Floorplanner: Demo Summary	11
3.4	Application-Centric Computing: Demo Summary	11
3.5	Reliable Distributed Design Data Management: Status Summary	12
3.6	Collaborative Design Workflows and Design of Experiments on the Internet: Demo Summary	13

List of Figures

1	Vela Home Page – linking nominal resources, including Six Tasks Home Page.	3
2	Vela Six Tasks Home Page – linking tasks and co-PIs associated with each task.	4
3	Vela Single Task Home Page example – under public access, linking task-specific resources.	5
4	Vela Single Task Home Page example – under protected access, linking task-specific resources.	6
5	Conceptual view of a testbed and an instance of application-specific implementation.	19
6	Executable playback of a recorded collaborative session on the Internet.	20

1 Introduction

The Vela Project brings together a team of co-PIs that spans 6 university locations across the USA: MIT, MSU, NCSU, Stanford, UCB and UCSB – all participants in a unique proof-of-concept experiment that aligns system designers, library and tool providers, and system integrators. The overriding objective of this experiment is to design a single-chip multimedia processor at a level of complexity that will be typical for next-generation systems and demonstrate the feasibility of constructing and supporting a distributed design environment. This environment will be molded to meet the needs of the design driver and will be used by a critical mass of distributed but collaborating participants concurrently, presenting a unique opportunity to evaluate the promises and the challenges of the proposed approach to distributed system design.

In the long term, the goals of this project are:

1. Demonstrate, using the most up-to-date internet-based technologies, the feasibility of a large-scale collaborative and distributed design effort for electronic systems with tools, libraries and personnel residing across the USA;
2. Identify challenges and opportunities for the next generation of design technology, given the rapid pace of advances in semiconductor technologies and the emergence of Internet-II;
3. Provide an effective technology transfer paradigm for DoD missions that will increasingly rely on COTS technologies in a distributed design environment.

The near-term objectives of this project include organization of real-time demos of distributed design activities, modelled on the Internet-based patterns of use that are beginning to emerge among project participants. The first of these demos took place during the Design Automation Conference in San Francisco, in the the University Booth¹, on June 15, 16, 17, 1998. Three of the servers in this project, from MSU, NCSU, and UCB, were moved to the University Booth that now supports, for the second year, the connection to the Internet. The QoS of the Internet connection was surprisingly high so the connection of these servers to their home bases across the country allowed each team to execute demos whose performance matched the performance experienced in the home office. The main objectives of this report are:

- to document the structure of interaction between the project participants;
- to provide a concise summary and context for each demo;
- to capture enough detail in the description of each executable demo, so that demos can be critically tested and improved by Vela project participants as well as collaborating partners elsewhere.

The major milestones of this project are:

DAC98 Demo: (1) focus on analysis and tentative approaches to working with distributed participants, servers, tools, and data; (2) demonstrate work-in-progress ranging from innovative web-based point tools, encapsulation of commercial tools for remote execution without relying on user-accounts, collaborative workflows that invoke a number of distributed tools and execution sequences, to the definition, abstractions, and analysis of the design driver chip.

DAC99 Demo: (1) focus on integration architectures and testing, by all participants, of distributed servers, tools, and data into well-defined executable and collaborative design workflows – while supporting the design driver; (2) demonstrate feasibility and limitations of migrating most if not all access of tools, data, and design workflows to platform-independent interfaces, i.e. accomplish most if not all design transactions through the nominal web-browser.

This report is organized into two main sections:

- Vela Project Navigaton Rules and Data Structures;
- Project Status and DAC'98 Demo Summaries.

For a companion viewgraph document, see http://www.cbl.ncsu.edu/publications_misc/1998-Talk-DACdemo/
For a complementary point of view of the on-going work in this and related projects, see the recent IEEE Spectrum article on the IC design on the World Wide Web [1], and an earlier article in EE Times [2].

¹The University Booth is organized by a team of ACM/SIGDA volunteers, this year led by Dr. Richard J. Auletta from University of Denver.

2 Vela Project Navigation Rules and Data Structures

The Vela Project navigation rules and data structures rely on four types of Web pages:

- (1) Vela Home Page – linking nominal resources, including Six Tasks Home Page, shown in Figure 1;
- (2) Vela Six Tasks Home Page – linking tasks and co-PIs associated with each task, shown in Figure 2;
- (3) Vela Single Task Home Page – under public access, linking task-specific resources, shown in Figure 3;
- (4) Vela Single Task Home Page – under protected access, linking task-specific resources, shown in Figure 4.

The *Vela Home Page*, maintained at <http://www.cbl.ncsu.edu/vela/>, supports links to a number of nominal resources and project information updates. The most important of these is the link to the *Vela Six Tasks Page* which itself maintains links to home pages of the six Vela tasks and the home pages of project co-PIs. The home page of each of the six tasks consists of two parts: one with public access, and the other with protected access. The protected page of each task is accessible only to Vela project participants who share a common password to gain this access.

While the Web pages as shown in Figures 1 and 2 are relatively static, the Web pages in Figures 3 and 4 that serve as the home page for each of the six tasks are changing as the project evolves. Specifically, Figures 3 and 4 illustrate the respective public/restricted home pages for tasks being completed at NCSU/CBL. Notably, we expect a thorough testing, by all Vela participants, of demos posted under these pages before making them accessible to the public pages under *Demos and Resources*. Similarly, to facilitate interaction and testing of each demo at each site, we expect all Vela project participants to organize their task home pages in a manner similar to one shown in Figures 3 and 4.

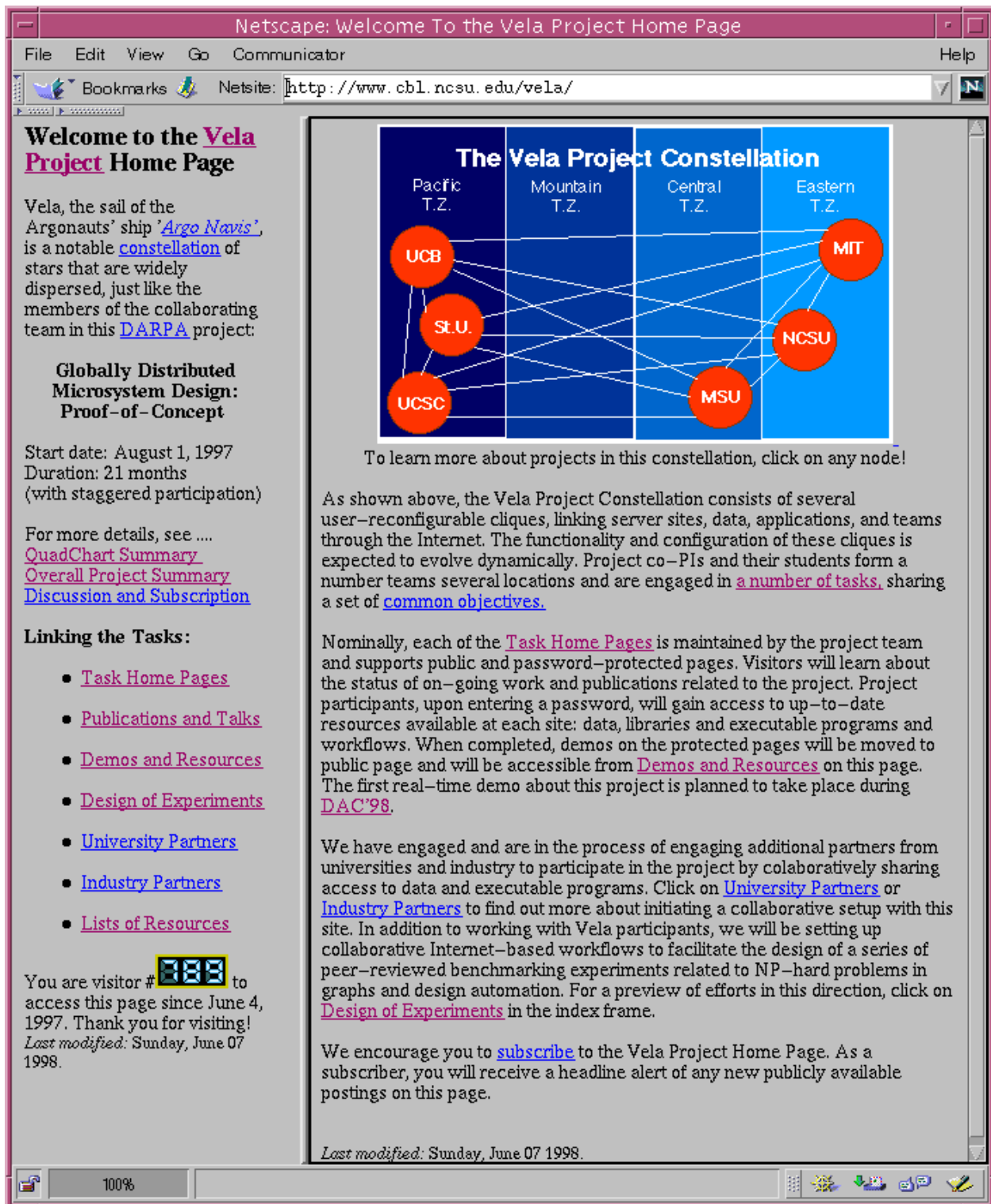


Figure 1: Vela Home Page – linking nominal resources, including Six Tasks Home Page.

Welcome to the Navigation Page about the Vela Projects!

Click on any link to find more details about each project, project co-PIs and project team members.

Project Name

Project Co-PIs

Team Members

Design Driver

[Jan M. Rabaey](#)

[Dimitri Antoniadis](#)

[Anantha Chandrakasan](#)

[Teresa H. Meng](#)

Distributed Design Libraries and Services

[Robert Reese](#)

Floorplanner

[Wayne W.-M. Dai](#)

Application-Centric Computing

[Giovanni De Micheli](#)

[Paul Losleben](#)

Reliable Distributed Design Data Management

[A. Richard Newton](#)

[Andrew Mayer](#)

[Mark Spiller](#)

Collaborative Design Workflows and Design of Experiments on the Internet

[Franc Brglez](#)

[Hemang Lavana](#)

[Debabrata Ghosh](#)

[Justin Harlow](#)

[Nevin Kapur](#)

Figure 2: Vela Six Tasks Home Page – linking tasks and co-PIs associated with each task.

Welcome to the Task Page on "Collaborative Design Workflows and Design of Experiments on the Internet"!

Major tasks of this project are:

- Coordinate co-PIs from a number of participating universities in a unique proof-of-concept experiment that aligns system designers, library and tool providers, and system integrators;
- Introduce and put to test includes a prototype of a user-reconfigurable, Internet-based collaborative workflow design environment. This environment supports visualization and automated task execution of sequences that invoke any number of distributed point tools and data, accessible either through the Web, RMI, telnet, ftp, ssh. Tools may be university-based and experimental, or commercial;
- Introduce 'Design of Experiments' methodology to leverage reconfigurable workflows and support web-based experiments that benchmark heuristic algorithms solving NP-hard problems;
- Organize a joint real-time collaborative demo of project accomplishments with all participants during DAC'98 and DAC'99 Exhibitions;

For a global view of the project, related publications and talks, see the [Vela Project Home Page](#).

If you a project co-PI [click here](#) and identify yourself with a name and a password to gain access to the restricted pages.

If you a collaborating partner [click here](#) and identify yourself with a name and a password to gain access to the restricted pages.

This access will allow you to work collaboratively, through a web-browser, on demos and designs-in-progress, share executable workflows, stand-alone programs, libraries, data files, and databases. When completed, demos on the protected pages will be moved to [Demos and Resources](#) in the public area of this page, which itself is linked to the [Vela Project Home Page](#). We encourage you to [subscribe](#) to the Vela Project Home Page. As a subscriber, you will receive a headline alert of any new publicly available postings on that page.

brglez@cbl.ncsu.edu

Figure 3: Vela Single Task Home Page example – under public access, linking task-specific resources.

Vela coPI-only page at CBL

This page is password protected, maintained at CBL, and is accessible only to the coPIs of the Vela project and their students. From this page, you may also engage in active [discussions](#) with other members of this group.

Major directories:

- **Demos:** are based on executable cgi-scripts, Java applets and Tcl/Tk applets. Preliminary demos are available for testing on this page before they are made accessible to everyone. At DAC '98, we will feature the following:
 - Mutate (REUBEN)
 - Place_Eval (REUBEN)
 - Partitioner-Xilinx (REUBEN)
 - Partitioner-SIS-Xilinx (REUBEN)
 - Synopsis-Partitioner-Xilinx (JavaCADD/REUBEN)
 - [WebWiseTclTk](#) (Tcl-plugin 2.0)
 - Cell-Characterization-Plot (OmniDesk)
 - [PosterNotes](#) (cgi-script/MhonArc)
 - [CoWTeX](#) (Java applet/cgi-script)
- **Saved data/results:** generated on execution of cgi-scripts and/or applets on the web server are available for download in this [area](#).
- **Data Libraries:** General purpose libraries are available [here](#).
- **Others:** All other information related to this page follows.

Notes:

- **Java applets** require Java-enabled web browsers for execution of the downloaded applet.
- **Tclets** (Tcl/Tk applets), on the other hand, rely on installation of [Tcl Plugin 2.0](#) in your web browser for its execution.
- URLs for any **demo** must be available from this page.
- URLs for any **data generated** must be accessible from this page.
- The directory structure for the demos and the data maintained at CBL site is available [here](#).

Figure 4: Vela Single Task Home Page example – under protected access, linking task-specific resources.

3 Project Status and Demo Summaries

This section summarizes the status of the Vela Project as of mid-June 1998, concurrent with the demos that took place in the University Booth during DAC'98 for three days: from June 15 – June 17, 11 AM – 2 PM. Most of the demos shown to visitors during DAC'98 Exhibition are now documented and accessible for execution through any web-browser from any location. For a complementary point of view of the on-going work in this and related projects, see the recent IEEE Spectrum article on the IC design on the World Wide Web [1], and an earlier article in EE Times [2].

The section is organized by project prime tasks and presented in a logical sequence that introduces the design driver, design libraries and services, web-based point tools and APIs, workflows of distributed tools and data, and the design of experiments in CAD algorithms and tools. The nominal titles of these projects are:

- Design Driver and Web-Based Tools;
- Distributed Design Libraries and Services;
- Web-Based Floorplanner;
- Application Centric Computing;
- Reliable Distributed Design Data Management;
- Collaborative Design Workflows and Design of Experiments on the Internet.

3.1 Design Driver and Web-Based Tools: Demo Summary

Project PIs: Dimitri Antoniadis (MIT), Anantha Chandrakasan (MIT), Teresa Meng (Stanford U.), Jan Rabaey (UCB)

Project Home Page: <http://apsara.mit.edu/vela>

The design driver is a single chip multimedia processor consisting of a video decompression unit, an ARM processor, and an embedded power supply. The design driver is being designed and simulated in a distributed fashion, with participants primarily located at MIT, UC Berkeley, and Stanford University.

The primary focus at MIT is the development of a Java based hierarchical block/schematic editor, WebTop. The editor serves as the front-end for the design driver specification and provides netlisting capabilities for various formats including Spice, Verilog, and PowerPlay. A Verilog RTL power estimation tool, Pythia, uses a collection of Verilog-PLI access and utility routines in order to traverse the circuit hierarchy, extract user-provided parameters, and monitor signal transitions. Each net in the design is annotated with three different capacitance values: gate, junction, and routing. Capacitive non-linearities are taken into account during the power estimation process. The power estimates are within 10% of PowerMill.

At UC Berkeley, macromodels to predict the behavior of the design driver have been developed for the ARM processor, IDCT, and DC-DC converter. These models take various parameters as input (operating frequency, "c" code, input data, etc.) and produce profiles of utilization (operation counts) and estimates for power dissipation. PowerPlay is a system level exploration tool, taking the design driver netlist from WebTop and estimates the overall power dissipation.

At Stanford University, a DC-DC converter and a simulator has been developed to predict the transient behavior of a low power DC-DC converter. The DC-DC converter is adaptive, based on desired performance. The simulator takes a desired voltage waveform and produces an output of the actual waveform. The DC-DC converter is used with the ARM module of design driver.

Executable Demos

All demos about the design driver are initiated on the MIT server at <http://apsara.mit.edu/WebTop>. From this URL, the user accesses the hierarchical schematic editor, WebTop, which is the entry point for the design driver. The tool has a number of primitive cells with which a designer can build other cells. The schematic editor supports extraction to various formats (Verilog, Spice, PowerPlay, etc.). The multimedia processor has been described using WebTop at various levels of abstraction. From the description, selected modules or the entire design can be simulated using remote tools. For example, the PowerPlay "conceptual"

estimator developed at UC Berkeley is used to estimate the power dissipation of the entire module taking into account the efficiency of the switching regulators. Similarly, a Verilog netlist of the video decompression module is extracted and its power dissipation is estimated using Pythia, a power estimation tool developed at Berkeley. The design libraries are stored remotely at various servers and can be accessed through a library manager built into WebTop.

The execution of any of the three demos currently accessible from the Web must be preceded by loading of all required data and libraries as follows:

- (1) access <http://apsara.mit.edu/WebTop>
- (2) From the WebTop main window, select from menu
Library -> Show LibMgr (to invoke Library Manager)
- (3) From Library Manager window, select from menu
File -> Load Cell -> Load From URL (to invoke a browser)
- (4) Click on "filter" to display the browser content for
<http://apsara.mit.edu/CellServer/> and click on "Load All"
- (5) Re-invoke browser and click on "filter" to display the browser content for
<http://apsara.mit.edu/CellServer/idctp/> and click on "Load All"
- (6) Re-invoke browser and click on "filter" to display the browser content for
http://apsara.mit.edu/CellServer/remote_demo/ and click on "Load All"

PowerPlay: High Level Power Estimation of Media Processor (executed at UC Berkeley).

An important aspect of the design process is to be able to evaluate and modify designs during the conception of a system. Furthermore, throughout the design process, it is necessary to be able to abstract key properties in order to analyze large, highly heterogeneous systems. The system design tool, PowerPlay, is designed to provide intuitive and flexible support to the exploration process of system design through the use of abstraction of key components of design objects.

PowerPlay was created leverage off the functionality of the World Wide Web. With PowerPlay, designers can create hierarchical views of their systems using a variety of models ranging from simple equations to CAD tools located anywhere on the WWW. Any design tool can be embedded into PowerPlay through the use of simple web forms. Systems can be explored through the use of hyperlinked spreadsheets and various forms of graphing.

To execute a demo of PowerPlay from within WebTop, proceed as follows:

- (1) From the WebTop Library Manager, select the `media_processor` cell and select/click on "schematic" (this will put the schematic of the `media_processor` cell into the WebTop main window).
- (2) In the WebTop main window, select the two `dc2dc` generators, ARM processor and IDCT chip, and from menu Extract, select PowerPlay Selected Extract (to invoke Extracted Parameters Window).
- (3) In the Extracted Parameters Window, make sure to select PowerPlay, and click on "submit".
After a minute or two, PowerPlay (estimation running at U.C. Berkeley) should return with a power breakdown for various components.

PowerPlay can also be accessed at <http://infopad.eecs.berkeley.edu/PowerPlay> directly.

Pythia: RTL Power Estimation of Video Decompression (executed at MIT).

Pythia uses a collection of Verilog-PLI access and utility routines in order to traverse the circuit hierarchy, extract user-provided parameters, and monitor signal transitions. Each net in the design is annotated with three different capacitance values: gate, junction, and routing. Capacitive nonlinearities are taken into account during the power estimation process. The power estimates are within 10% of PowerMill.

To execute a demo of Pythia from within WebTop, proceed as follows:

- (1) In the WebTop main window, select the idct chip and descend the hierarchy (double click on selection, click on edit [to edit the subcell], and then say "ok" when it asks to discard changes)
- (2) from menu Extract,
 - select Pythia Selected Extract (his will perform a hierarchical extraction and invoke Extracted Parameters Window).
- (3) In the Extracted Parameters Window, make sure to select Pythia, and click on "submit". Since Pythia estimates power at a RTL, it is significantly slower. The invocation of Pythia might take a few minutes, we have saved the results in the following location:
 - `http://apsara/pythia-results/blks14/show.html`
 - (do not submit design, unless you want to wait for a few minutes).

DC-DC Converter: Remote Simulation (executed at Stanford).

To minimize power consumption, the supply voltage can be dynamically adjusted according to the processing load and temperature variations to operate at the lowest possible voltage level. An efficient DC-DC switching regulator capable of rapidly delivering the specified voltage with minimal ripples is required. The dynamic switching regulator that employs a variable switching frequency from 1 MHz to 10 kHz and a non-linear feedback loop has been designed. The converter efficiency for voltages ranging from 1.5V to 3.5V while delivering a power level from 10 mW to 200 mW is measured to center around 88% (85% to 93%).

To execute a demo of DC-DC converter simulation from within WebTop, proceed as follows:

- (1) From the WebTop Library Manager, select the media_processor cell and select/click on "schematic" (this will put the schematic of the media_processor cell into the WebTop main window).
- (2) In the WebTop main window, select Stanf_Power_Generator
- (3) from menu Extract,
 - select Stanford Selected Extract (to invoke Extracted Parameters Window).
- (4) In the Extracted Parameters Window, make sure to select Stanford Power Estimator, and click on "submit".

A plot of the DC-DC converter transient response will appear. The simulation is run at Stanford.

3.2 Distributed Design Libraries and Services: Demo Summary

Project PI: Robert Reese (MSU)

Project Home Page: <http://www.erc.msstate.edu/mpl/vela>

The future of ECAD design libraries and services will be of a distributed nature, with vendors providing libraries (layouts, characterization data, etc.) and services (characterization, place/route, generators) via the WWW. MSU is tasked with being the library and service provider for the rest of the VELA group; initial approaches have concentrated on using Java Remote Method Invocation (RMI) within a Java-based client/server architecture. Features aimed at improving collaboration between library and service users are being included in the design of the client/server architecture. The client/server architecture is being integrated by NCSU into the Vela collaborative workflow environment.

Distributed Design Libraries. MSU has developed two standard cell libraries for use with the MOSIS silicon brokerage service. These two libraries are compatible with the 0.5u, 0.8u, and 1.2u processes offered through MOSIS. Via a JavaCADD server running at MSU, a user can synthesize a VHDL behavioral model to either library. The resulting EDIF netlist can then be placed/routed and a GDSII file returned. The synthesis (Synopsys) and place/route (Mentor Autocells) tools are accessed via the JavaCADD server technology.

The Vela Project will produce a library of complex building blocks, including core processors, multimedia accelerators, memories and bus interfaces. Power, area, and timing predictors for the industry-standard Synopsys DesignWare macroblock library as mapped onto a target cell library whose characterization data

is based upon the MIT 0.18um spice models will be encapsulated so as to allow remote execution by higher-level predictor/estimator tools such as PowerPlay from UC Berkeley.

JavaCADD Services. A Java-based server and client used for distributed ECAD services has been put to extensive tests [3]. The server/client combination allows user access to batch-oriented ECAD services (synthesis, place/route, etc) without having to provide full login-access for the users. The JavaCADD GUI uses templates stored at the server for GUI definition, so new services can be added without modifying the Java source code of the GUI. Java Remote Method Invocation (RMI) is used for client/server communication. Both client and server are Java applications, and share the operating system independence unique to Java. JavaCADD has been used by undergraduate classes at MSU to access ECAD applications and computing resources located at the MSU/NSF Engineering Research Center. JavaCADD allows access to these resources without having to grant full login privileges to this large, transient user population.

Executable Demos

The JavaCADD demos given at DAC'98 exercised key points of the JavaCADD server and client programs. The server was running on an 8-processor UltraSparc located at MSU, with remote tool configuration and monitoring supplied by the client running on the local Sparc workstation on the DAC floor.

Synopsys Synthesis and Mentor Layout Tools in JavaCADD (executed at MSU).

Synopsys Synthesis and Mentor standard cell place & route tools are executed remotely at MSU via the JavaCADD client interface. This demo is now executable from <http://www.erc.msstate.edu/mpl/vela>

UCSC-Based Floorplaner in JavaCADD (executed at UCSC).

This demo was set-up on the the DAC floor and required about 30 minutes of work to get the JavaCADD server running on the host DAC machine with a JavaCADD interface for the UCSC-Based Floorplaner tool. This was an unplanned, spontaneous and opportunistic setup to demonstrate the ease of encapsulation of batch-oriented tools with the JavaCADD client/server, no knowledge of Java is required for this setup. An enhancement of this demo is now executable from <http://www.cse.ucsc.edu/research/vela/>

Synopsys, Partitioner and Xilinx on the Web (executed at MSU and NCSU).

JavaCADD client was also configured as a program node in a reconfigurable, executable, and collaborative workflow REUBEN [4], invoked at NCSU. A flow of distributed tools and data, Synopsys at MSU, netlist partitioner and Xilinx place & route at NCSU, has been executed and multi-cast to two workstations where collaborating students could negotiate the execution of the workflow and view the initial data and final results. This demo is now executable from <http://www.cbl.ncsu.edu/demos/>

Impact of JavaCADD for the Vela Project. Any batch-oriented tool can be encapsulated using JavaCADD for WWW-access with minimum effort. This includes user configurations of any number of JavaCADD clients as program nodes in NCSU's workflow environments such as REUBEN and OmniFlow. While the same functionality can be achieved via CGI scripts and WWW server, some advantages of the JavaCADD server (a Java application) are:

- With CGI, one often requires permission from the local administrator before adding executable CGI scripts. CGI scripts are also run on the same machine as the WWW server, which means the CAD tool will also run there (unless one adds an rsh or something in the CGI script to spawn the tool on the other machine). The JavaCADD server can be run on any machine, under any user, without having any special root access. A test server has been setup easily on the host machine at DAC. The JavaCADD server can also be run under NT.
- One can access remote tools from either a browser interface or from the JavaCADD client. The advantage of the JavaCADD client (a Java application) is that one need not be concerned with the browser compatibility (ie. will it work under Netscape 4.0X? or Internet Explorer 4.??).

3.3 Web-Based Floorplanner: Demo Summary

Project PIs: Wayne W.-M. Dai (UCSC)

Project Home Page: <http://www.cse.ucsc.edu/research/vela/>

The main goal of this research is to prototype a floorplanning tool that supports a large and complex design whose components are distributed and designed at remote locations. The floorplanner will support rectilinear blocks, soft regular blocks, and general non-slicing structures. The floorplanner is to be timing driven, power-sensitive, and noise-aware, and support incremental updates or do a complete plan from scratch. We plan to use the Web to coordinate the design evolution and coordination among the remote locations.

Most of previous floorplan studies are based on slicing structure. For building block layout with multi-layer technology, channel routing will be replaced by area routing gradually as blocks are packed together to minimize the area. This technology shift makes non-slicing floorplan more and more attractive. We have developed and implemented a new representation which can represent the solution space of the general floorplan (rectilinear shape and non-slicing structure) completely and evaluate the solution quickly. This representation is crucial for the efficiency of the stochastic optimization methods. A paper on the theoretical foundation of this floorplanner will be presented during ICCAD'98.

Executable Demos

Two demos were supported by the current version of the floorplanner, as summarized below.

Floorplanner using Java RMI (executed at UCSC).

As the first step, we demonstrated rectilinear block packing. The block information and the net lists were passed from the client (workstations at DAC university booth) to the server (workstations at the University of California at Santa Cruz) and the floorplanning results were sent back to the client and displayed locally for efficiency.

Floorplanner in JavaCADD/RMI (executed at UCSC).

This demo was set-up on the the DAC floor and required about 30 minutes of work to get the JavaCADD from MSU server running on the host DAC machine with a JavaCADD interface for the UCSC-Based Floorplanner tool. This was an unplanned, spontaneous and opportunistic setup to demonstrate the ease of encapsulation of batch-oriented tools with the JavaCADD client/server. An enhancement of this demo is to be posted for from <http://www.cse.ucsc.edu/research/vela/>

The on-going effort includes working with JavaCADD - a Java-based server/client architecture, developed by MSU, to obtain information of fully encapsulated system components and evolving design information (i.e. module size, netlists, and timing, power and noise budget) over the web from each design team. Such encapsulation will also support user configurations of any number of JavaCADD clients as program nodes in NCSU's workflow environments such as REUBEN and OmniFlow.

The floorplanner will be invoked automatically via the Internet by the change of any system components. The floorplanner will support incremental updates or do a complete plan from scratch. The up-to-date floorplan of the processor chip will be viewable at all time.

3.4 Application-Centric Computing: Demo Summary

Project PIs: Giovanni De Micheli (Stanford U.), Paul Losleben (Stanford U.)

Project Home Page: <http://akebono.stanford.edu/users/aneesh/Vela/>

The main goal of this project is to create an integrated environment for synthesis and validation tools, that can be accessed using the WWW. The tools will be both local and remote as well as both research prototypes and commercial standard tools. An example of such tools is PPP, a Web-based environment for Low-Power Design. Its Graphic User Interface is a set of dynamically generated HTML pages that can be accessed through any Web-browser. Three sets of tools are embedded within PPP for Low Power Synthesis, Power Simulation and Power Optimization.

About PPP. Power dissipation is a critical metric for design evaluation. Some of the current power estimation techniques include electrical-level simulators, logic-level simulators, and switching activity analysis.

Although these methods offer promising results, they do not assume any model for gates and do not deal with multiple input transitions that are not aligned perfectly in time.

PPP exploits a BDD based symbolic model for describing the charging and discharging of parasitic capacitances and the flow of short circuit current. Based on the algorithms presented by Benini et al. [5], PPP always achieves an accuracy of within 6% from SPICE. VERILOG-XL is used as the simulation platform to maintain compatibility with other design environments.

The User Interface of PPP is a set of dynamically generated HTML pages that can be accessed through any Web-browser. File transfer utilities allow users to upload the input files on the server and download the final results or the status logs. The Web-based interface for the tool allows the user to run the tool remotely through his/her Web-browser, thus eliminating the need for any software installation. PPP has been designed by the Stanford Synthesis Group.

Executable Demos

There are three sets of tools for Synthesis for low power, Power Optimization, and Power Simulation purposes. Demos presented at DAC'98 can be re-executed through any web-browser from the PPP home page at <http://akebono.stanford.edu/users/PPP/>

PPP, Low-Power Synthesis Tool.

This tool performs automatic Gated-Clock generation, using the algorithms presented by Benini et al. [5]. Starting with a FSM specification in the kiss2 format, the tool finds idle conditions for which the clock can be stopped, thus saving power, without sacrificing functionality. The output after synthesis is a mapped verilog netlist using the gate level simulator in PPP.

PPP, Power Simulation Tool.

This tool embeds an accurate simulator for power and current estimation that operates at the gate level, based on the symbolic model of CMOS gates proposed by A. Bogliolo et. al. The outputs of the simulator are power statistics, current waveforms and average/instantaneous power/energy dissipation data.

PPP, Waveform Display and Editor.

This tool outputs several data files including power and current statistics. This information can be presented in waveforms either as a GIF file or using the Java Waveform Editor. The Java based Waveform Editor is a recent addition to PPP, specifically designed to overcome the static nature of the GIF format. With the Java Waveform Editor, the user can choose a particular waveform and zoom in to any portion of the graph and have the graphs updated dynamically. Some of the waveforms that can be displayed in the editor include the average and instantaneous current consumption, the peak power distribution, and the signal transitions on the inputs of the circuit.

3.5 Reliable Distributed Design Data Management: Status Summary

Project PIs: Richard Newton (UCB), Jan Rabaey (UCB)

Project Home Page: <http://www-cad.eecs.berkeley.edu/weld/vela/>

The basic question we are addressing is: "How can one provide reliable design data management for long transactions and large data sets in a distributed, potentially unreliable, environment?" We are continuing to develop and evaluate the proxy-based, component-oriented approach first introduced in our earlier WELD work but extended to include a DCOM-based distributed implementation. We address the scalability of such approaches in the face of very large (up to a terrabyte) data sets, a variety of tools, and long transactions (over 24 hours in some cases). Can we develop an architectural approach that delivers BASE response time and interactivity with ACID robustness and overall guarantees? Finally, and perhaps most importantly, what is the specific role that caching can play in this overall architecture-does it help or hinder? We expect this work will involve the implementation and both theoretical and experimental evaluation of a number of prototype systems.

Status Summary

This part of the project started recently. Based on our initial evaluation of the Chip Hierarchical Design System prototype of the SEMATECH and Si2 CHDStd API and architecture last year, we plan to undertake this work assuming a CHDStd-like interface and data model. We hope the research will provide feedback as to the suitability of such a data model and interface architecture for distributed design. The CHDStd subset we choose will form an API available to the entire VELA group for evaluation and integration.

3.6 Collaborative Design Workflows and Design of Experiments on the Internet: Demo Summary

Project PI: Franc Brglez (NCSU)

Project Home Page: <http://www.cbl.ncsu.edu/vela/>

Major task of this project is the coordination of co-PIs from 6 universities in a unique proof-of-concept experiment that aligns system designers, library and tool providers, and system integrators. One of the technologies to be introduced and put to test includes a prototype of a user-reconfigurable, Internet-based collaborative workflow design environment. This environment supports visualization and automated task execution of sequences that invoke any number of distributed point tools and data, accessible either through the http, RMI, telnet, ftp, ssh. Tools may be university-based prototypes, or commercial. ‘Design of Experiments methodology leverages reconfigurable workflows and supports web-based experiments that benchmark heuristic algorithms solving NP-hard problems.

Collaborative Design Workflows on the Internet. The goal of this project is to create a re-configurable desktop environment that provides the user with the ability to create *executable* directed dependency graphs as workflows of data, program, decision, script, and workflow nodes. Data and programs can reside anywhere on the Internet, and execution of all nodes can be scheduled automatically, regardless of the data-dependent cycles in the graph. Such an environment can serve the needs of distributed design teams as well as support an infrastructure for the design of experiments in the context described in next section.

We have evolved two generation of executable workflows. The more mature version, called REUBEN, executable through a web-browser under X-Windows and implemented in *Tcl/Tk* [6] and *Expect* [7], supports (1) workflow composition and reconfiguration while accessing and executing programs, data, and computing resources across the Internet, (2) synchronous and asynchronous peer-to-peer interaction between members of any team during workflow composition and execution, (3) synchronous and asynchronous peer-to-workflow interaction between any team member and any object in the workflow, (4) recording of all transactions and re-executable playback [2, 4, 8, 9, 10].

Our latest effort introduces OmniDesk and OmniFlows, a workflow environment similar to REUBEN, but now executable through a web-browser exclusively [11]. To execute a fully featured OmniDesk/OmniFlow without compromising security, we introduce WebWiseTclTk [12, 13], a significant extension of Tcl2.0 plug-in capabilities [14, 15]. While only Tcl2.0 plug-in must be locally installed, the execution of OmniDesk/OmniFlow itself can be accelerated by also installing WebWiseTclTk locally.

Current efforts by Vela Project participants have lead to development of a number of state-of-the-art web-based tools. While complex on its own, these tools ultimately behave as point tools; including the ones with hard-wired links to other web tools and data. There is always a case of one more tool to be executed in a data-dependent sequence relative to other tools – and where user-interaction is required to link data output from one tool to the input of another tool. The success of the workflow environment such as REUBEN and OmniDesk/OmniFlows will be measured to what extent it can facilitate user-configuration of executable design flows that *automatically link* the distributed point tools and data in the Vela Project.

Design of Experiments on the Internet. The *Design of Experiments* is a well-established discipline in science since the pioneering work of R.A. Fisher in 1920s. However, the introduction of design of experiments to critique and significantly improve on today’s practice of ‘benchmarking’ CAD algorithms and tools is relatively new [16, 17, 18, 19, 20]. We are seeking to raise awareness, through web-based publications of

peer-reviewed design of experiments, of the fundamental questions such as “is the reported improvement due to the improvement of the algorithm or due to chance?”. Our experiments demonstrate that the executable workflow environment driven by the current design driver can also serve as a testbed and an infrastructure for the design of experiments in CAD algorithms and tools.

Executable Demos

The execution of the currently accessible demos in public domain from the web-browser requires that user becomes familiar with basics of navigation in ReubenDesktop environment as follows:

- (1) move the X-window called ReubenDesktop to upper-left corner of your screen.
- (2) move the X-window called WorkflowManager in a position adjacent to ReubenDesktop.
- (3) in WorkflowManager window, double-click on any yellow (data) node to invoke the mini-browser that will allow you to access the directory of files bound to this node. Before execution, data nodes that are outputs of blue (program) nodes will have no data files. To open any data file, double-click on the icon associated with the file of a given name. To CLOSE the mini-browser, double-click on the yellow (data) node.
- (4) if you invoked the workflow in the "edit+execute" mode, you may also EDIT the input data file(s) (or paste any data from another X-window).
- (5) to execute the workflow, select the "Entire Workflow" under the "Execute" menu in the WorkflowManager.
- (6) to quit the workflow display and execution, select "Exit" under the "File" menu in the ReubenDesktop.

Not all demos presented at DAC'98 are accesible to the public, though all are described here. For security reasons, only the first six demos listed below are executable through a web-browser by accessing <http://www.cbl.ncsu.edu/demos/>

Synopsys-Partitioner-Xilinx Workflow.

Here you execute a workflow of tools and data at two collaborating sites: (1) MSU and (2) NCSU. The NCSU site invokes Synopsys logic synthesis at MSU via JavaCADD/RMI. The synthesized netlist, returned to NCSU, invokes ULJ/CBL partitioner, and each partition invokes the Xilinx placement and routing tool. The demo can be executed in two modes: (1) execute-only and (2) edit+execute. The "execute-only" mode does not allow you to edit input data file while in the "edit+execute" mode, you can edit any input data file(s) (or paste any data from another X-window).

Signature Invariant Circuit Mutation.

Such mutations induce an equivalence class of netlists that are useful in the design of experiments to evaluate CAD algorithms and tools. Here you execute the process of mutant generation from a given reference netlist. The mutation process generates a circuit benchmark of the comparable complexity as that of the reference circuit (details are described in [17]). The reference netlist may be user-supplied. However, in the "execute-only" mode, the demo shows you the process for the circuit `roth.blif` only. You can generate mutants for circuits of your own through the "edit+execute" mode. There is a 'browse' button on the "edit+execute" page through which you can upload your circuit. Please go through the documentation on the "edit+execute" page before executing mutation of your own circuit.

For more details, <http://www.cbl.ncsu.edu/experiments/> and [17, 18, 19].

Simple Design of Experiments (SIS).

Here you submit a (small illustrative) equivalence class of netlists to a logic optimization tool (SIS) that interfaces, after invocation of three distinct scripts (algebraic/boolean/rugged), to a common logic evaluator. This evaluator reports the total number of 2-input nodes (inverters have been absorbed in 2-input nodes) and the total number of logic levels. The demo can be executed in two modes: (1) execute-only and (2) edit+execute. The "execute-only" mode does not allow you to edit input data file while in the "edit+execute" mode, you can edit any input data file(s) (or paste any data from another X-window).

For more details, <http://www.cbl.ncsu.edu/experiments/> and [18, 19]

Complex Design of Experiments (TOCO).

Here you submit a (small illustrative) equivalence class of netlists to a prototype placement tool (TOCO) that interfaces, at each placement phase, to a common placement evaluator. This evaluator (`Place_Eval`) reports total/critical wire length, total/critical wire crossing, total/critical wire density, total feedthroughs, as well as height, width, and area of the embedded graph. The demo can be executed in two modes: (1) execute-only and (2) edit+execute. The "execute-only" mode does not allow you to edit input data file while in the "edit+execute" mode, you can edit any input data file(s) (or paste any data from another X-window).

This is an example of a hierarchical workflow, executing and evaluating user-configured phases of placement algorithms on 100's of circuit mutants, archiving results, and generating statistical summaries of performance parameter distributions.

A conceptual view of a testbed for such design of experiments, and an instance of the workflow implementation with `Place_Eval`, is shown in Figure 5.

For more details, <http://www.cbl.ncsu.edu/experiments/> and [21, 22].

Introducing WebWiseTclTk.

The WebWiseTclTk toolkit is an extension to Safe-Tcl plug-in that supports (1) creation of new Web-based Tcl applications with greatly enhanced functionality (as compared to Safe-Tcl), and (2) migration of existing Tcl applications to the Web by merely writing an encapsulation script. Using WebWiseTclTk, the Internet-based Tcl/Tk workflows such as demonstrated here, will soon be executable entirely through the Web browser – obviating the reliance on X-windows.

The current demos are available in the execute-only mode, will test your browser plug-in capabilities, and are essentially illustrative. A real-time interactive demos with executable workflows will be available by the end of Summer 1998.

For more details, <http://www.cbl.ncsu.edu/software/> and [11, 12, 13]

PosterNotes.

This is a web-based environment with features that combine those of mailing lists, newsgroups, and the web [23]. The main purpose of PosterNotes is to facilitate and support threaded communication among the members of our team as well as the Internet-based collaborators and peers. Currently, the `PosterNotesManager` supports the setup for four distinctive types of archives: (1) public discussion group archives, (2) collaborative archives, (3) CBL's group archives, (4) CBL's individual user archives.

For more details, access the PosterNotes homepage at <http://www.cbl.ncsu.edu/DiscussionGroups/>

CoWTeX.

This is a web-based system written in Java to support collaborative team creation and dynamic composition of documents written in LaTeX and LaTeX2e [24]. Each document in LaTeX can be converted to HTML via `latex2html` [25]; documents written in LaTeX2e can include HTML tags for automatic creation of hyperlinks in the converted HTML document. Composed documents are rendered for preview and archival as web-based *.ps, and *.html files. Team members can access the system through a web-browser from anywhere at anytime. Most documents, including this document, now posted under <http://www.cbl.ncsu.edu/publications/> have been generated by collaborating participants using CoWTeX.

CBL team members and Vela Project can access CoWTeX through a password protected web-page.

Complex Partition-Optimize-Place-Route-Flow.

This hierarchical workflow is too complex for a demo via a public access on the the Web. The workflow is an example of a collaborative multi-site design flow that uses university and industry tools across several hosts, as shown in Figure 6 – this figure has also been used in the recent IEEE Spectrum article on the IC Design on the WWW [1].

For experimental purposes, we have captured its entire execution as a recording that can be played back and re-executed for performance testing under different network conditions not only from CBL but also from a server site at Duke University some 40 miles away, and from a server site at UC Berkeley some 4000 miles away [10].

As shown in Figure 6, the workflow invokes a recursive partitioner, a logic optimizer, and a placement and routing tool in a data-dependent cycle that terminates with a number of optimized, placed, and routed devices. The dialog boxes in Figure 6 capture the process of decision making between three remotely located participants. Each participant gets to view the same image of the workflow: program nodes are blinking when any program is executing, and arrows are flashing when data is transferred from one host to another host. However, only one participant at a time controls the execution of any preselected paths in the workflow. The assumption is that no single participants understands the intricacies of all tools, hence the reliance on a team of specialists who may be distributed geographically. The *entire workflow* can also be executed automatically by a single participant and only the final results are viewed and analyzed by a distributed team. This is the scenario we describe briefly next.

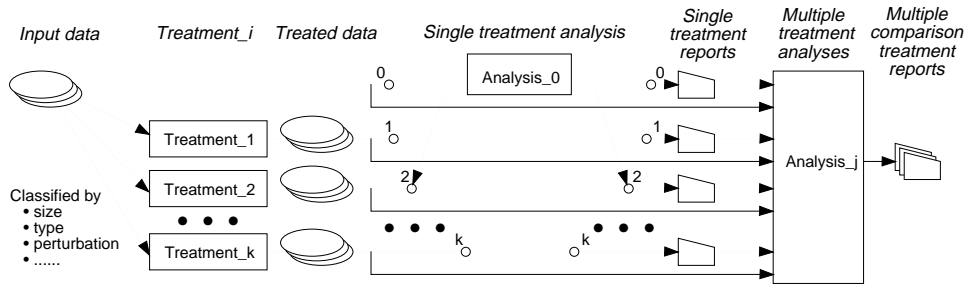
1. Using `OmniBrowser`, invoke the playback session;
2. a netlist is transferred from Host 1 to Host2, read by a `Partitioner` on Host2. Tool outputs are two netlists: a *tentative partition* on Host3 and a *remainder partition* on Host2.
3. The tentative partition is read by a `Optimizer` tool on Host3. This program node is shown as a hierarchical node that expands upon tool invocation as shown in the small figure below. The tool called `SIS` [26] outputs a netlist as an *optimized partition* on Host2.
4. The optimized partition is evaluated on Host2 for a performance metrics (e.g. area) by an `Evaluator`.
5. The output of evaluator is evaluated in the decision node `Partition_Acceptable?`.
6. At this point, the output of the decision node is *NO*, and a script node `Modify_variables` is invoked, changing the partitioning control parameter strategy to less aggressive.
7. Since one of the data inputs to `Partitioner` has changed (data in the node `Modify_variables`), the program node `Partitioner` is re-invoked on the same partition, now with different control parameters. Again, it outputs two netlists: a *tentative partition* on Host3 and a *remainder partition* on Host 2 (the old ones are overwritten).
8. Again, the tentative partition is read by the `Optimizer` tool on Host3 and evaluated by `Evaluator` on Host2. Now, the decision from the node `Partition_Acceptable?` is *YES*.
9. Two actions are invoked by this decision: (1) the optimized partition is transferred to archival directory `Partition_Archival` on Host1, and, (2) the remainder partition is listed and copied to a `New_Netlist`. Since one of the data inputs to `Partitioner` has changed by now (i.e. data in the node `New_Netlist`), the program node `Partitioner` is re-invoked on the new netlist. Again, it outputs two netlists: a *tentative partition* on Host3 and a *remainder partition* on Host 2 (the old ones are overwritten).
10. Again, the tentative partition is read by the `Optimizer` tool on Host3 and evaluated by `Evaluator` on Host2. Again, the decision from the node `Partition_Acceptable?` is *YES*.
11. Again, (1) the optimized partition is transferred to archival directory `Partition_Archival` on Host1, and, (2) the remainder partition is listed and copied to a `New_Netlist`.
12. Again, the program node `Partitioner` is re-invoked on the new netlist, but now outputs a *tentative partition* on Host3 and no *remainder partition* on Host 2.
13. The tentative partition is read by the `Optimizer` tool on Host3 and evaluated by `Evaluator` on Host2. Again, the decision from the node `Partition_Acceptable?` is *YES*.
14. Again, the optimized partition is transferred to archival directory `Partition_Archival` on Host1, but now, the remainder list is empty, invoking a `Place&Route` tool, sequentially, on each of the three archived partitions.
15. The playback stops at this point and returns control to the user, who can do any number of things: examine contents of various data directories generated during the execution of the workflow, including the placed and routed results, with a team of distributed participants; re-execute the entire workflow with different control parameters; reconfigure the workflow for different data directories and/or different partitioners, optimizer and placement tools; etc.

References

- [1] L. Geppert. IC Design on the World Wide Web. *IEEE Transactions on Computer-Aided Design*, pages 45–50, June 1998. Includes a special section “Sailing on the Internet” about the Vela Project, pp. 45–47.
- [2] R. Goering and P. Clarke. Vela project could chart new course for engineering on the Net – Web-based design hoists new sail, 1997. Includes description of a real-time, interactive and collaborative demo of REUBEN in the University Booth during DAC’97 exhibition (June 9-13). Available at <http://www.techweb.com/se/directlink.cgi?EET19970616S0001> and also archived under http://www.cbl.ncsu.edu/publications_others/#1997-DAC-EETimes-Goering-Web.
- [3] D. Linder, R. Reese, J. Robinson, and S. Russ. JavaCADD: A Java-based Server and GUI for Providing Distributed ECAD Services. Technical Report MSSU-COE-ERC-98-07, Microsystems Prototyping Laboratory, MSU/NSF Engineering Research Center, April 1998. Also available at <http://WWW.ERC.MsState.Edu/mpl/publications/papers/javacadd/JCaddTR.pdf>.
- [4] H. Lavana, A. Khetawat, F. Brglez, and K. Kozminski. Executable Workflows: A Paradigm for Collaborative Design on the Internet. In *Proceedings of the 34th Design Automation Conference*, pages 553–558, June 1997. Also available at <http://www.cbl.ncsu.edu/publications/#1997-DAC-Lavana>.
- [5] A. Bogliolo L. Benini and G. De Micheli. Distributed EDA tool integration: the PPP paradigm. In *Proceedings of the International conference on Computer Design*, 1996.
- [6] The Tcl/Tk Home Page. Published under URL <http://sunscript.sun.com/>, 1997.
- [7] D. Libes. *Exploring Expect*. O’Reilly and Associates, 1995.
- [8] Amit Khetawat. Collaborative Computing on the Internet. Master’s thesis, Electrical and Computer Engineering, North Carolina State University, Raleigh, N.C., May 1997. Also available at <http://www.cbl.ncsu.edu/publications/#1997-Thesis-MS-Khetawat>.
- [9] H. Lavana, A. Khetawat, and F. Brglez. Internet-based Workflows: A Paradigm for Dynamically Reconfigurable Desktop Environments. In *ACM Proceedings of the International Conference on Supporting Group Work*, Nov 1997. Also available at <http://www.cbl.ncsu.edu/publications/#1997-GROUP-Lavana>.
- [10] A. Khetawat, H. Lavana, and F. Brglez. Internet-based Desktops in Tcl/Tk: Collaborative and Recordable. In *Sixth Annual Tcl/Tk Conference*. USENIX, September 1998. Also available at <http://www.cbl.ncsu.edu/publications/#1998-TclTk-Khetawat>.
- [11] H. Lavana and F. Brglez. WebWiseTclTk, OmniDesk and OmniFlows: A User-Configurable Distributed Design Environment inside a Web-Browser. Technical Report 1998-TR@CBL-03-Lavana, CBL, CS Dept., NCSU, Box 7550, Raleigh, NC 27695, April 1998. Also available at <http://www.cbl.ncsu.edu/publications/#1998-TR@CBL-03-Lavana>.
- [12] The WebWiseTclTk Toolkit Home Page. Published under URL <http://www.cbl.ncsu.edu/software/#WebWiseTckTk>.
- [13] H. Lavana and F. Brglez. WebWiseTclTk: A Safe-Tcl/Tk-based Toolkit Enhanced for the World Wide Web. In *Sixth Annual Tcl/Tk Conference*. USENIX, September 1998. Also available at <http://www.cbl.ncsu.edu/publications/#1998-TclTk-Lavana>.
- [14] The Tcl Plugin Home Page. Published under URL <http://sunscript.sun.com/plugin>, 1997.
- [15] J. K. Ousterhout, J. Y. Levy, and B. B. Welch. The Safe-Tcl Security Model. Published under URL <http://scriptics.com/people/john.ousterhout/safeTcl.ps>, March 1997. Draft.
- [16] N. Kapur, D. Ghosh, and F. Brglez. Towards A New Benchmarking Paradigm in EDA: Analysis of Equivalence Class Mutant Circuit Distributions. In *ACM International Symposium on Physical Design*, April 1997.

- [17] Debabrata Ghosh, Nevin Kapur, Justin E. Harlow III, and Franc Brglez. Synthesis of Wiring Signature-Invariant Equivalence Class Circuit Mutants and Applications to Benchmarking. In *Proceedings, Design Automation and Test in Europe*, pages 656–663, Feb 1998. Also available at <http://www.cbl.ncsu.edu/publications/#1998-DATE-Ghosh>.
- [18] F. Brglez (Editor). A Brief Tour From The Home Page on WWW Statistical Experiment Archives: Benchmark Descriptions and Posted Solutions to NP-hard Problems. Technical Report 1998-TR@CBL-01-Brglez, Version 1.0, CBL, CS Dept., NCSU, Box 7550, Raleigh, NC 27695, February 1998. Also available at <http://www.cbl.ncsu.edu/publications/#1998-TR@CBL-01-Brglez>.
- [19] F. Brglez. Design of Experiments to Evaluate CAD Algorithms: Which Improvements Are Due to Improved Heuristic and Which Are Merely Due to Chance? Technical Report 1998-TR@CBL-04-Brglez, CBL, CS Dept., NCSU, Box 7550, Raleigh, NC 27695, April 1998. Also available at <http://www.cbl.ncsu.edu/publications/#1998-TR@CBL-04-Brglez>.
- [20] J. Harlow and F. Brglez. Design of Experiments for Evaluation of BDD Packages Using Controlled Circuit Mutations. In *International Conference on Formal Methods in Computer-Aided Design (FMCAD'98)*. IFIP, November 1998. Also to be available at <http://www.cbl.ncsu.edu/publications/#1998-FMCAD-Harlow>.
- [21] N. Kapur. Cell Placement and Minimization of Crossing Numbers. Master's thesis, Electrical and Computer Engineering, North Carolina State University, Raleigh, N.C., May 1998. Also available at <http://www.cbl.ncsu.edu/publications/#1998-Thesis-MS-Kapur>.
- [22] N. Kapur, D.Ghosh, and F. Brglez. Optimization of Placements Driven By the Cost of Wire Crossing. Technical Report 1997-TR@CBL-08-Kapur, CBL, CS Dept., NCSU, Box 7550, Raleigh, NC 27695, Oct 1997. Also available at <http://www.cbl.ncsu.edu/publications/#1997-TR@CBL-08-Kapur>.
- [23] The PosterNotes Home Page. Published under URL <http://www.cbl.ncsu.edu/software/#PosterNotes>.
- [24] The CoWTeX Home Page. Published under URL: <http://www.cbl.ncsu.edu/software/#CoWTeX>.
- [25] N. Drakos. The LaTeX2HTML Translator, February 1998. Published under URL <http://www-dsed.llnl.gov/files/programs/unix/latex2html/manual/>.
- [26] SIS – Release 1.2. UC Berkeley Software Distribution, 1992. For more information, see <ftp://ic.eecs.berkeley.edu/pub/Sis/README>.

(a) Web-based design of experiments: data, treatments (e.g. optimization algorithms), and analyses



(b) Web-based implementation of a specific placement evaluation workflow, reported by Place_Eval

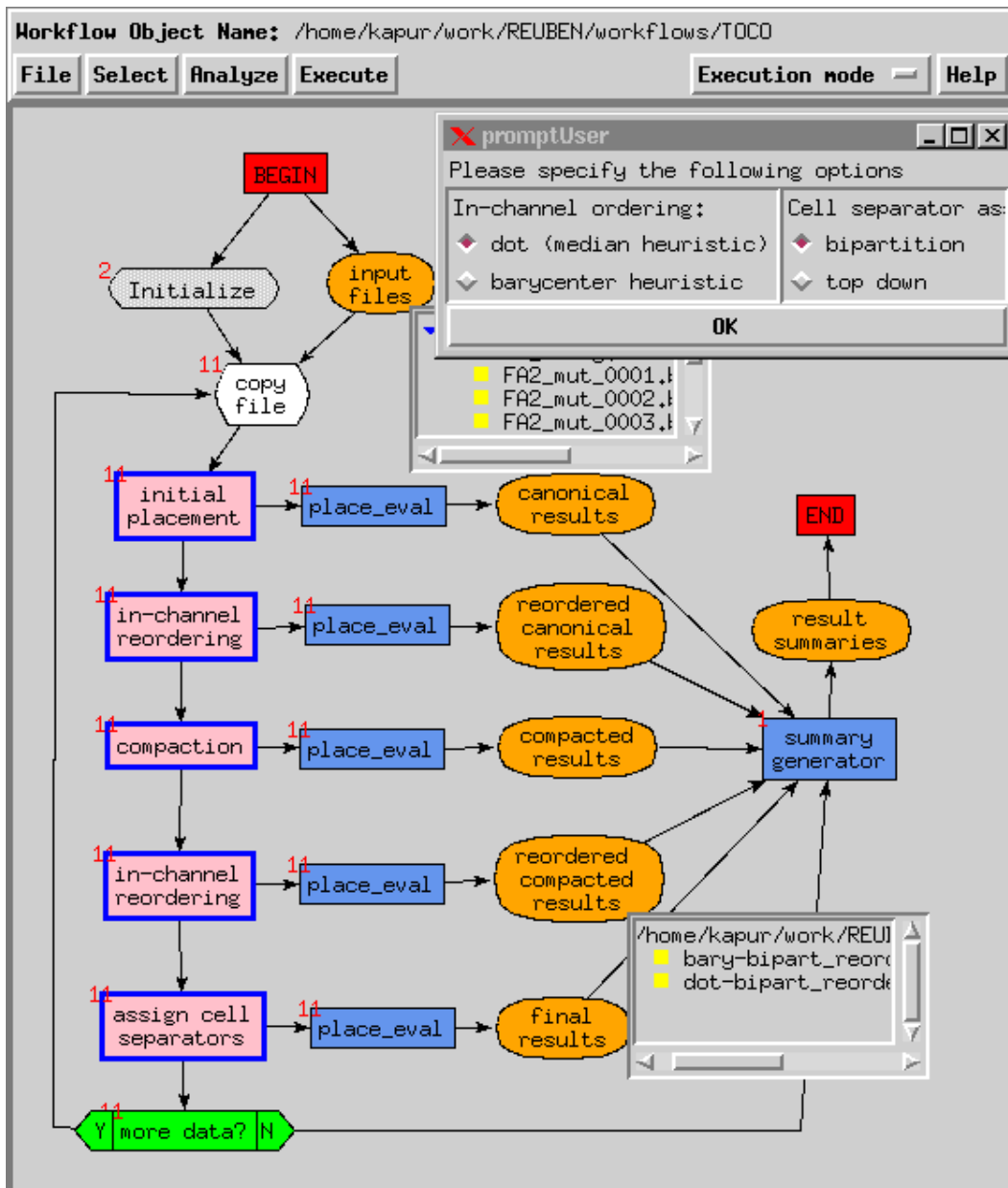


Figure 5: Conceptual view of a testbed and an instance of application-specific implementation.

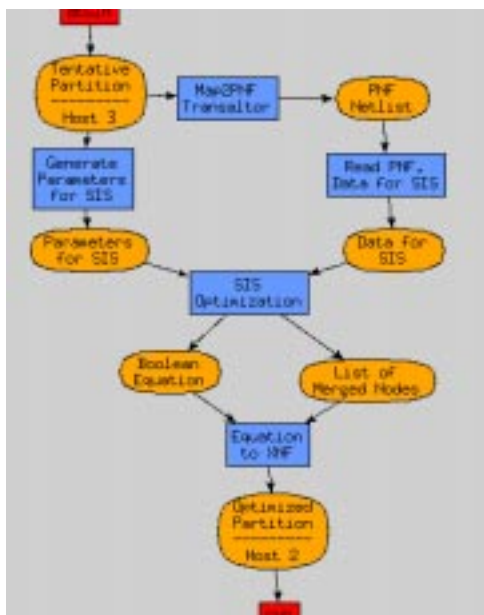
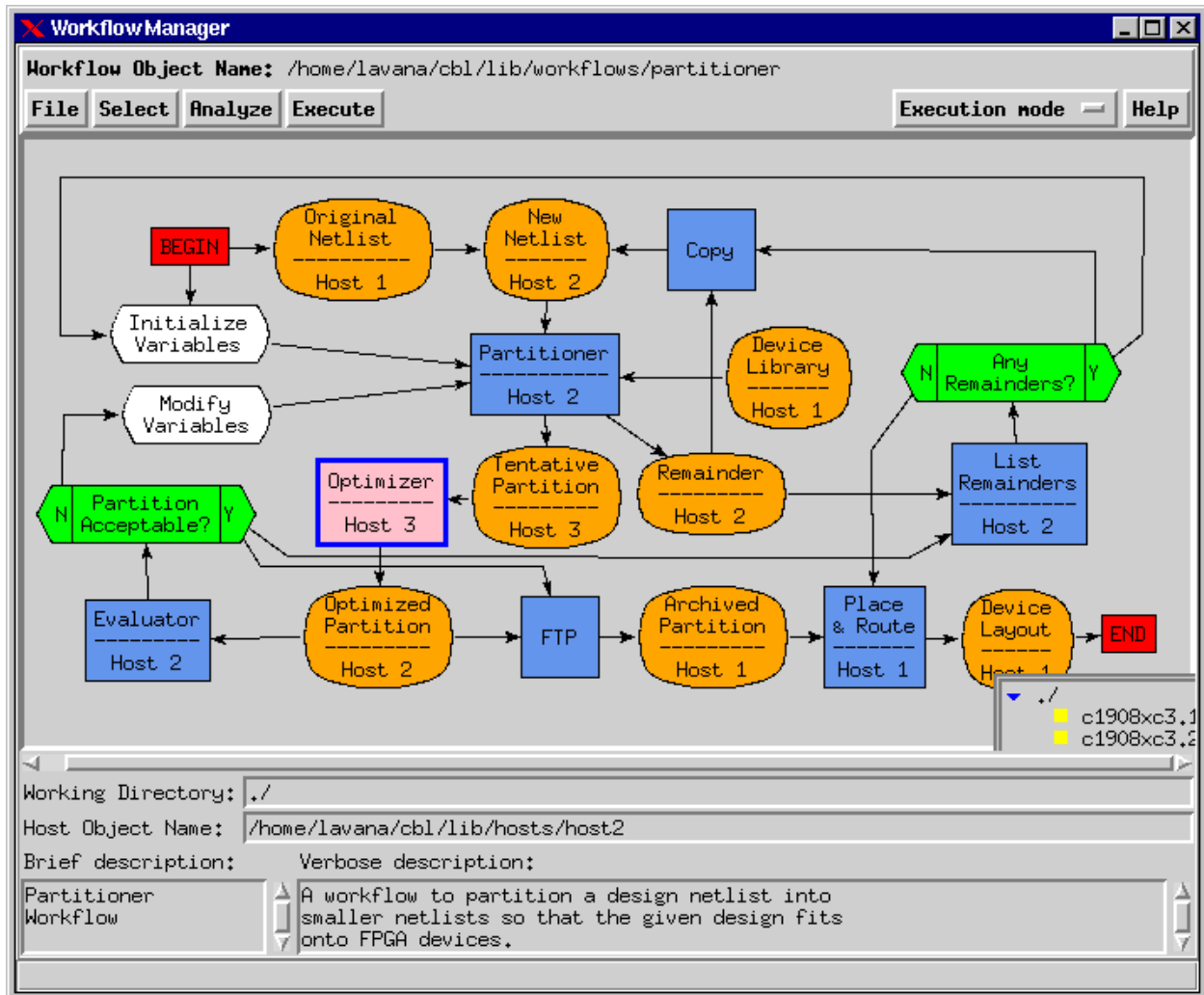


Figure 6: Executable playback of a recorded collaborative session on the Internet.