VISTA - A FRAMEWORK FOR TECHNOLOGY CAD PURPOSES

R. Strasser, Ch. Pichler, and S. Selberherr

Institute for Microelectronics, TU Vienna
Gußhausstraße 27–29, A–1040 Vienna, Austria
Phone: +43/1/58801-3680, Fax: +43/1/5059224,
e-mail: strasser@iue.tuwien.ac.at, URL: http://www.iue.tuwien.ac.at

KEYWORDS

Computer Aided Engineering (CAE), VLSI & simulation, Optimization, Object-oriented, Intelligent simulation environments.

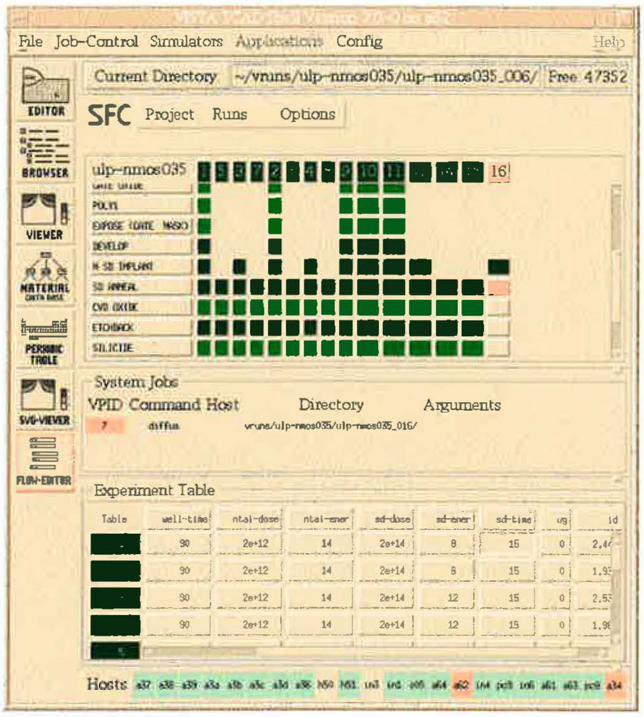
ABSTRACT

This paper describes the design and implementation of the Technology CAD simulation framework VISTA as well as its features. Object-oriented design is a key point in the implementation of VISTA. A Tool abstraction mechanism, job farming, automatic load balancing, parallel execution, a simulation flow controller, gridding services, visualization tools and persistent storage facilities are core components for highlevel simulation modules such as Design of Experiments-, Response Surface Methodology- and Optimization modules. The impact of this Technology CAD framework on the productivity of its users is pointed out.

INTRODUCTION

During the past decade numerous highly effective simulators for the simulation of semiconductor technology (PROMIS, TSUPREM (TMA 1995), ATHENA (Silvaco 1993)) as well as for semiconductor devices (transistors, diodes, ...) (MINIMOS (Habaš et al. 1990), MEDICI (TMA 1994)) have been developed. These simulators deliver reasonable and accurate predictions of process and device performance. Nevertheless, the time consuming efforts to setup, carry out and analyze simulations often prevent designers from performing simulations, neglecting the positive impact of simulation technology on cycle time and cost reduction, respectively. TCAD (Technology CAD) frameworks like VISTA (Halama et al. 1993) (Viennese Integrated System for TCAD Applications) intend to close this gap between the potential capabilities of simulation tools and a technology engineers time restrictions. In the following the framework infrastructure required to perform complex simulation tasks is outlined.

SYSTEM IMPLEMENTATION



Current Directory ~/vruns/ulp-nmos035/ulp-nmos035_006/ Free 47352

System Jobs

```
VPID Command Host        Directory                             Arguments
7    diffus              vruns/ulp-nmos035/ulp-nmos035_016/
```

Experiment Table

Table	well-time	ntal-dose	ntal-ener	sd-dose	sd-ener	sd-time	ug	id
	90	2e+12	14	2e+14	8	15	0	2,4
	90	2e+12	14	2e+14	8	15	0	1,9
	90	2e+12	14	2e+14	12	15	0	2,5
	90	2e+12	14	2e+14	12	15	0	1,9

Figure 1: The VISTA main window with the Simulation Flow Control window provides compact representation of simulation progress and results

The VISTA framework is based on a LISP interpreter called VLISP, which provides object-oriented features similar to the popular JAVA language. This interpreter-based system provides a sound basis for the implementation of a robust and highly portable software system. Object orientation has been found to be a key factor in terms of productivity, maintainability, and robustness during the development of VISTA. Fundamental tasks are provided by a collection of object classes (graphical user interface, operating system interface, basic data types and structures). The facilities of VLISP to execute remote system processes in parallel enable the efficient usage of system resources and thus decrease simulation time. Special care has been taken to separate the framework modules from the user interface without degrading the ease of use. Thus it is easily possible to initiate and monitor complex simulation task via a Graphical User Interface (GUI) or alternatively by issuing commands on a terminal line or executing scripts in batch mode. The rigorous separation of the user interface is expected to facilitate the port of VISTA to newly emerging user interface standards in the future and will therefore significantly extend its lifetime.

TOOL ABSTRACTION

VISTA is an open framework, which means that it is easily possible to integrate any simulation tool by providing interface stubs for different types of simulation tools. These stubs serve as abstractions for the tools inside of the framework. The framework communicates with the tools using typed data. In order to be able to perform simulations with heterogeneous simulation tools, converters have to be registered for new data types. Due to this fact the framework is able to perform automatic conversion of data formats whenever necessary during simulations with heterogeneous tools.

SIMULATION FLOW CONTROLLER

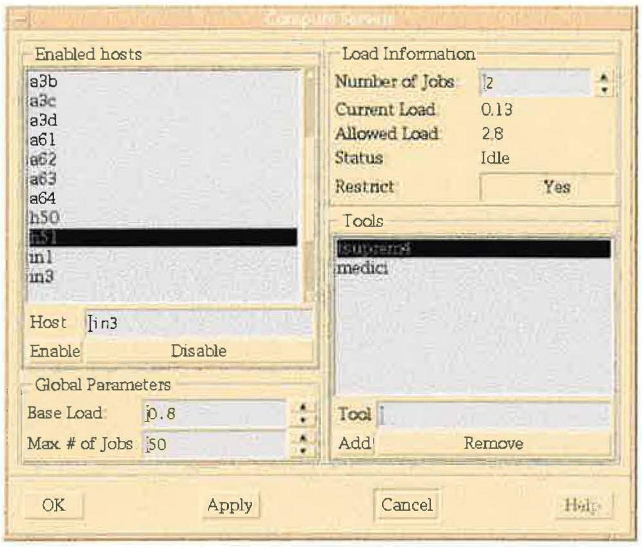
The VISTA *Simulation Flow Controller* (SFC) takes care of scheduling simulations and maintains a persistent result database. Basically the SFC *evaluates* Simulation Flow Descriptions which means that the according simulators are started and error conditions are are handled. It extensively reuses simulation results by maintaining an experiment split tree. Thus re-computations of already existing results are avoided.

Simulation tools have to register their typed simulation output with the SFC. Thus the result of a basic SFC operation can be considered a vector of simulation results, these might be simple numbers or more complex objects like a full *Wafer–state*.

The SFC provides a user interface as depicted in Figure 1 for the simulation results as well as for the simulation progress. This user interface gives a compact overview for even complex experiments like they arise from optimization. On request the user is able the view the output of each particular step of a simulation.

LOAD BALANCING

Since VISTA is able to distribute workload for its computations to several workstations, load balancing is a serious issue. In order to minimize the computation time required for a simulation, VISTA periodically polls for the actual load of each registered node and thus enables automatic load balancing. Individual load limits for each node as well as global load limits guarantee a flexible and highly effective — in terms of resource optimization — distribution of simulation workload. Additionally automatic load balancing avoids overloading


Enabled hosts
a3b
a3c
a3d
a61
a62
a63
a64
h50
h51
in1
in3

Host: in3

Load Information
Number of Jobs: 2
Current Load: 0.13
Allowed Load: 2.8
Status: Idle
Restrict: Yes

Tools
tsuprem4
medici

Global Parameters
Base Load: 0.8
Max. # of Jobs: 50

Figure 2: The Compute Server Panel is used to enable/disable hosts, to configure specific tools for a host, and to specify parameters for load balancing.

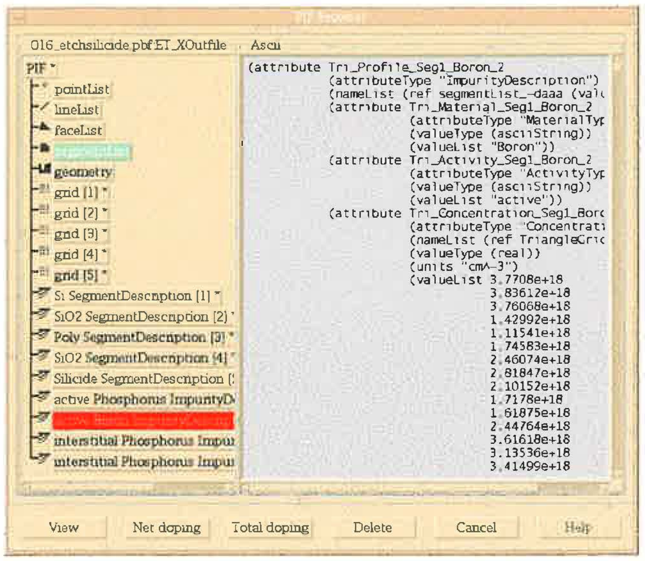
of workstations and therefore prevents resource usage conflicts among concurrent users.

DATA MODEL

The central data object of a process simulation is the so called *Wafer–state* which is a model of a cross section of the simulated piece of wafer. The actual representation of the wafer–state depends on the file format selected. Several formats of wafer–state representations exist. The VISTA native wafer–state representation is PIF (Halama et al. 1993), which is a hierarchical model as depicted in Figure 3. A graphical view of this wafer–state is shown in Figure 4. An example of another wafer state representation handled by VISTA is the *Technology Interchange Format* (TIF) used by TSUPREM.

GRIDDING

VISTA provides gridding facilities through a Delaunay grid generator, TRIANGLE (Shewchuk 1996). This gridder is able to generate high quality grids using selectable quality criteria (minimum angle, maximum area, doping gradient). Due to its triangular nature it is able to create grids for even complicated domains as they result from etching or oxidation simulations. Additionally, TRIANGLE works as a wafer–state gridder, which is used to create a valid data model after simulation tools that do not write a complete wafer model. Typically this is the case for topography simulators that do not care for doping profiles. TRIANGLE merges the new geometry resulting from a topography simulation with preexisting doping profiles. Additionally, due to its adaptive gridding facility, merging of doping pro-


016_etchsilicide.pbf ET_XOutfile
Ascii
PIF
pointList
lineList
faceList
geometry
grid [1]
grid [2]
grid [3]
grid [4]
grid [5]
Si SegmentDescription [1]
SiO2 SegmentDescription [2]
Poly SegmentDescription [3]
SiO2 SegmentDescription [4]
Silicide SegmentDescription
active Phosphorus ImpurityD
interstitial Phosphorus Impu
interstitial Phosphorus Impu
View
Net doping
Total doping
Delete
Cancel
Help

```
(attribute Tri_Profile_Seg1_Boron_2
           (attributeType "ImpurityDescription")
           (nameList (ref segmentList_-daaa (val
           (attribute Tri_Material_Seg1_Boron_2
                      (attributeType "MaterialTyp
                      (valueType (asciiString))
                      (valueList "Boron"))
           (attribute Tri_Activity_Seg1_Boron_2
                      (attributeType "ActivityTyp
                      (valueType (asciiString))
                      (valueList "active"))
           (attribute Tri_Concentration_Seg1_Borc
                      (attributeType "Concentrati
                      (nameList (ref TriangleGric
                      (valueType (real))
                      (units "cm^-3")
                      (valueList 3.7708e+18
                                 3.83612e+18
                                 3.76068e+18
                                 1.42992e+18
                                 1.11541e+18
                                 1.74583e+18
                                 2.46074e+18
                                 2.81847e+18
                                 2.10152e+18
                                 1.7178e+18
                                 1.61875e+18
                                 2.44764e+18
                                 3.61618e+18
                                 3.13536e+18
                                 3.41499e+18
```

Figure 3: An integrated VISTA data viewer.

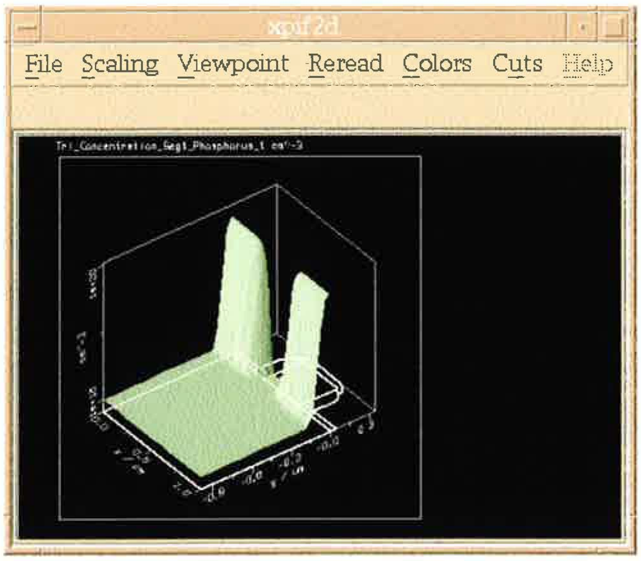


Figure 4: A two-dimensional data viewer for the VISTA native PIF format.

files that reside on different grids is possible ensuring a minimum of accuracy loss.

PERSISTENT STORAGE OF RESULTS

In order to avoid recomputation of simulation results, VISTA keeps a persistent database for simulation results. As disk space can be a serious issue for large scale experiments, only intermediate results at split points of the experiment split tree are kept to keep the amount of required disk space at a minimum.

PROCESS FLOW REPRESENTATION

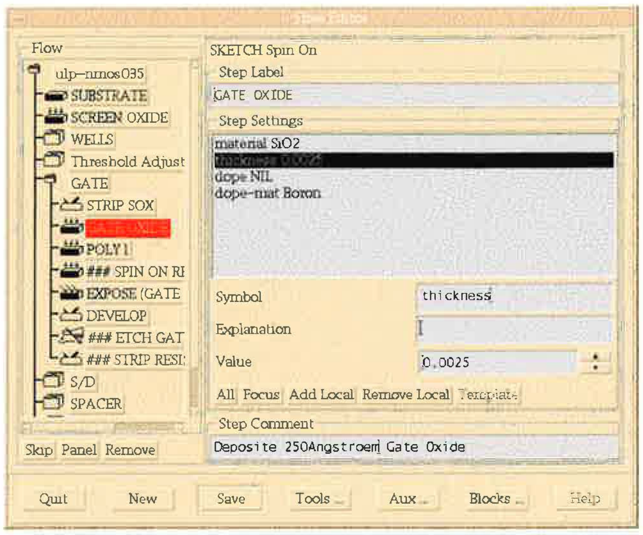
A vital issue of process simulation is the definition of the sequence a wafer undergoes during fabrication - the so called *process flow*. VISTA uses the *Simulation Flow Description* (SFD) (Pichler 1997) format to represent process flows. Basically the SFD encapsulates instances of process statements that are registered with the framework. Furthermore the SFD allows the introduction of a hierarchy in order to facilitate navigation through complex processes. Figure 5 depicts the VISTA flow editor which is provided to easily build and maintain process flow descriptions.

To combine ease of use and a high level of flexibility, a template based approach of simulator control is provided. The input to each simulation tool can be divided into a *control* part and a *data* part. The data part is completely handled by the framework itself, whereas the control part is partially controlled by the framework and partially controlled by the simulation flow description. In Figure 6 a typical template based step for a MINIMOS device simulation is depicted. A template of the control part (namely the input deck) of the simulator input is presented in the text window. Parameters of the template are surrounded by `<(... )>` sequences. Each parameter is defined by its name and its default value. For those parameters which also have a default value, an input field is available (for example `US`, `UG`). If a default value is missing, they are assumed to be hidden (for example `CHANNEL`, `MAT`). Values of hidden parameters are typically provided by the framework, but for some reasons users might want to change them. Generally the user is able to modify the template itself or the input fields of parameters. If additional parameters are introduced by editing the template, they are dynamically rescanned and input fields are created. Parameters of these template are accessible for clients of the SFC and are typically used for automatic process variation.

By providing this template mechanism simulators that require some kind of an input deck can easily be integrated to the framework. Full transparency of the simulator control is achieved, which means that from the users point of view a simulator is basically used in no other way than it would have been done in a standalone simulation. Once integrated, a simulator benefits from the highlevel framework services described later.

SIMULATION METHODOLOGY

The facilities described so far form the basis for application focused TCAD simulations like optimizations or calibration. However, computation time restrictions force the application of sophisticated methods like *Response Surface Methodology* (RSM) (Box and Draper 1987). RSM provides a compact universal numerical model for arbitrary types of data. Instead of requiring computation time consuming simulations the optimization uses a corresponding RSM model for its evaluations. To create RSM models with


```
Flow
ulp-nmos035
  SUBSTRATE
  SCREEN OXIDE
  WELLS
  Threshold Adjust
  GATE
    STRIP SOX
    GATE OXIDE
    POLY 1
    ### SPIN ON RI
    EXPOSE (GATE
    DEVELOP
    ### ETCH GAT
    ### STRIP RESI
  S/D
  SPACER
Skip  Panel  Remove

SKETCH Spin On
Step Label
GATE OXIDE
Step Settings
material SiO2
thickness 0.0025
dope NIL
dope-mat Boron
Symbol        thickness
Explanation
Value         0.0025
All  Focus  Add Local  Remove Local  Template
Step Comment
Deposite 250Angstroem Gate Oxide

Quit  New  Save  Tools ...  Aux ...  Blocks ...  Help
```

Figure 5: The VISTA flow editor

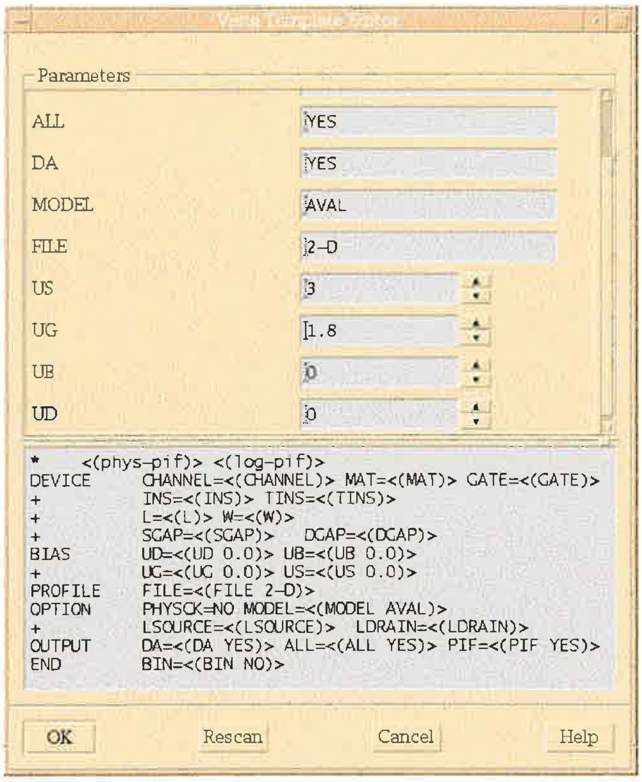
a minimum amount of simulation experiments, *Design of Experiments* (DOE) (Lorenzen and Anderson 1991) support is included within VISTA. DOE is a methodology to explore the input parameter space with a minimum number of samples. VISTA encapsulates these tasks by so called *agents*(Figure 7) which provide communication interfaces with external DOE, RSM and optimizer client applications. These agents communicate with the clients via standard input and standard output. Therefore various implementations (in form of separate programs) for these tasks can easily be plugged in. Figure 1 shows the split tree resulting from of a *CCC* DOE design of a CMOS process.

TASK ENCAPSULATION

In order to provide a uniform interface to basic operations like the evaluation of a SFD (which means an actual process simulation), RSM evaluation or optimization a unified class of *Evaluable Entity* (EVE) LISP objects is provided, which serves as an abstraction of the operations behind. Figure 7 illustrates the encapsulation of an optimizer by an EVE interface. More complicated tasks are created by plugging EVEs together as shown in Figure 8. These virtual interfaces allow for an easy substitution of simulation modules by other ones. A typical example of this is the substitution of a *real* process simulation by an RSM model which was generated out of a set of process simulations before. As depicted in Figure 8 the optimizer requests evaluations from the RSM model instead of the process simulation itself.

CONCLUSION

A TCAD framework that provides all necessary components and functionality for state of the art TCAD investigations has been presented. This framework has successfully been applied to problems arising in industrial technology development laboratories. Especially optimization capabilities represent a highly valuable feature of the VISTA framework. TCAD frameworks have become a key technology for the semiconductor industry.


```
Parameters
ALL      YES
DA       YES
MODEL    AVAL
FILE     2-D
US       3
UG       1.8
UB       0
UD       0

*          <(phys-pif)> <(log-pif)>
DEVICE     CHANNEL=<(CHANNEL)> MAT=<(MAT)> GATE=<(GATE)>
+          INS=<(INS)> TINS=<(TINS)>
+          L=<(L)> W=<(W)>
+          SGAP=<(SGAP)>   DGAP=<(DGAP)>
BIAS       UD=<(UD 0.0)> UB=<(UB 0.0)>
+          UG=<(UG 0.0)> US=<(US 0.0)>
PROFILE    FILE=<(FILE 2-D)>
OPTION     PHYSCK=NO MODEL=<(MODEL AVAL)>
+          LSOURCE=<(LSOURCE)>  LDRAIN=<(LDRAIN)>
OUTPUT     DA=<(DA YES)> ALL=<(ALL YES)> PIF=<(PIF YES)>
END        BIN=<(BIN NO)>

OK   Rescan   Cancel   Help
```

Figure 6: A GUI for simulator templates

FURTHER RESEARCH

Future development of VISTA will especially cover aspects of client/server computing. The ongoing progress of the world wide web and its impact on existing soft-

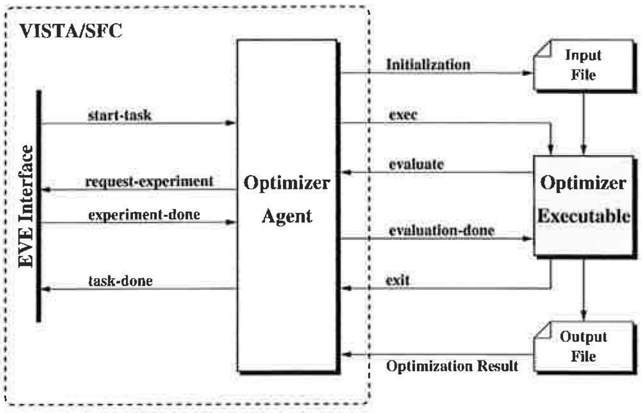


Figure 7: Basic operations are wrapped by so called EVE interfaces

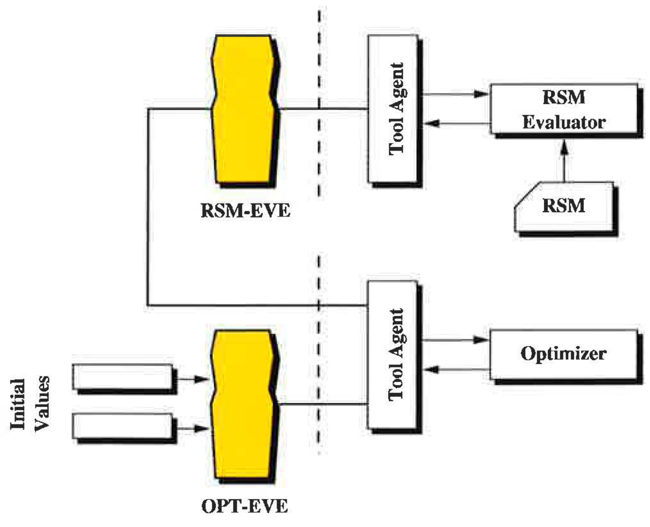


Figure 8: Basic tasks are plugged together via EVE interfaces to perform complex operations like optimization

ware will have influence on future development decisions. One could imagine that VISTA acts as some kind of simulation request server which handles requests from client applications on a remote Internet site. In such a scenario components requiring heavy user interaction are located on the client side and the core simulation modules reside on the server side as depicted in Figure 9. The strict separation of the user interface and the functional components of the framework strongly support this development.

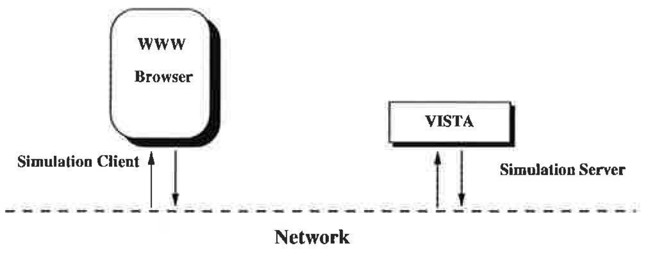


Figure 9: Client/Server architecture for future simulation scenarios

ACKNOWLEDGMENT

This work was significantly supported by the "Christian Doppler Forschungsgesellschaft", Vienna, Austria; Austria Mikro Systeme, Unterpremstätten, Austria; Digital Equipment Corporation, Hudson, USA; Hitachi, Tokyo, Japan; Motorola, Austin, USA.

REFERENCES

Box, G. and Draper, N. 1987. *Empirical Model-Building and Response Surfaces.* Wiley.

Habaš, P.; Heinreichsberger, O.; Lindorfer, P.; Pichler, P.; Pötzl, H.; Schütz, A.; Selberherr, S.; Stiftinger, M., and Thurner, M. 1990. *MINIMOS 5 User's Guide.* Institut für Mikroelektronik Technische Universität Wien, Austria.

Halama, S.; Fasching, F.; Fischer, C.; Kosina, H.; Leitner, E.; Pichler, C.; Pimingstorfer, H.; Puchner, H.; Rieger, G.; Schrom, G.; Simlinger, T.; Stiftinger, M.; Stippel, H.; Strasser, E.; Tuppa, W.; Wimmer, K., and Selberherr, S. 1993. The Viennese Integrated System for Technology CAD Applications. In Fasching, F.; Halama, S., and Selberherr, S., editors, *Technology CAD Systems* pages 197–236 Wien. Springer.

Lorenzen, T. and Anderson, V. 1991. *Design of Experiments.* Marcel Dekker.

Pichler, C. 1997. *Integrated Semiconductor Technology Analysis.* Dissertation Technische Universität Wien.

Shewchuk, J. R. 1996. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In *First Workshop on Applied Computational Geometry* pages 124–133. Association for Computing Machinery.

Silvaco 1993. *ATHENA: 2D Process Simulation Framework.* Silvaco. User's Manual.

TMA 1994. *TMA MEDICI, Two-Dimensional Device Simulation Program, Version 2.0.* Technology Modeling Associates, Inc. Palo Alto, CA.

TMA 1995. *TMA TSUPREM-4, Two-Dimensional Process Simulation Program, Version 6.2.* Technology Modeling Associates, Inc. Palo Alto, CA.

BIOGRAPHY

Rudolf Strasser was born in Ried im Innkreis, Austria, in 1970. He studied electrical engineering at the Technical University of Vienna, where he received the degree of 'Diplomingenieur' in 1995. From spring 1992 to autumn 1993 he held a research position at the Campus-based Engineering Center of Digital Equipment Corporation, Vienna, Austria. He joined the 'Institut für Mikroelektronik' in April 1995. In summer 1996 he was with the Advanced Products Research and Development Laboratory at Motorola, Austin. He is currently working towards his doctoral degree. His scientific interests include semiconductor technology, grid generation and software engineering.