# UML/UEDK:
# An Open Platform for CAD/EDA Design Learning

José Augusto D. F. Lima, Ph.D.

Departamento de Informática, Universidade do Minho,
Campus de Gualtar, 4710-057 Braga, Portugal
Email: jal@di.uminho.pt

**Abstract - The UML/UEDK (Universidade do Minho layout Learning, Utilization Environment and Development Kit) is a CAD/EDA tool framework, particularly suited to be used in courses having emphasis in the design automation aspects of VLSI and digital design. This open platform, uses the UMLe-SDK (Universidade do Minho Layout environment Software Development Kit) to provide CAD/EDA design apprentices with basic data-structures, events, methods and graphical user interface (GUI), needed to support and facilitate the development of "autonomous" CAD operations. Complementarily, a web based platform, makes available to both VLSI and CAD/EDA designers, the specifications, the tutorial descriptions, and the code of selected UMLe operations. Any UMLe operation - old or new alike - once installed, becomes available through the menus, layout templates or tool buttons it adds into the UML/UEDK backbone "layout editor". A typical student's learning curve requires the use of existing operations followed by the development of his/her own.**

## 1. Introduction

As far back as 1994, the NTRS Roadmap - a US SIA (Semiconductor Industry Association) publication - states: *"It is strongly felt that limitations in the design processes are rapidly becoming a severe bottleneck in the semiconductor industry. Better understanding of the problem; increased investment; and improved coordination between user companies, universities, and EDA suppliers will be essential to provide the new set of tools and techniques."* Such limitations must be overcome if the industry's staggering rate of growth, yielding a new chip every 18 months, is to be maintained.

Design productivity, namely those aspects related to designers' qualifications and readiness to effectively use, and design, evolving CAD/EDA tool suites, has become a bottleneck in the growth of the industry. Hence, a potential show stopper, identified in the Roadmaps has been human expertise and design team productivity. Another complementary key aspect, being the need the industry has for "adequate" CAD/EDA tools in order to be able to design each new chip generation.

Given this context, we think universities can contribute to help ease this impending problem in a three fold fashion. Introduce VLSI design courses whose curriculum is relevant to these technology challenges; offer courses where emphasis is put in training students in automation based design; introduce students in the design of CAD/EDA automation tools.

## 2. Objectives and Educational Use

First, the UML/UEDK [1] platform is intended as an effective way of helping students learn - from a pedagogical and an engineering viewpoint - how to design basic VLSI automation tools. Second, it is intended as a teacher's in class tool, providing

teachers with an effective, and flexible, way of illustrating concepts, algorithms, and point tool solutions for physical design problems. Third, it provides a lasting, as well as useful, platform for students to have selected coursework small projects live on pass their own school year, instead of gaining dust in the teacher's office. Also showing fellow students of the same engineering degree - LESI (Licenciatura em Engenharia de Sistemas e Informática) - examples to follow, and results against which to compare their own efforts.

In class we have used the UML/UEDK to illustrate basic channel routing algorithms, floorplanning representations and algorithms, semi-automatic generation of layout cells, logic synthesis, BDD and SOP optimizations, DRC validation of layouts, cross-sections of layouts, generation of traditional as well as alternative FPGA layout structures, etc. Because these UML/UEDK operations are the result of students' CAD/EDA small projects they are most appropriate, and well adapted, to illustrate the corresponding VLSI design and CAD/EDA key concepts of the course. In fact, one would be hard pressed to isolate within professional tools, these key elements, and algorithms, because they are embedded in a complex design flow, and are entangled with other design features. Making its analysis and presentation to complex to be feasible within introductory level courses.

Often the UML/UEDK problem specifications ask students to show key steps of the algorithm implemented in their solution, not only the final result. This allows the student a piecewise approach to the problem, facilitates debugging and correctness check, and has the interesting side-effect of being useful for later demonstration in class. Also, the students seeing the demonstrations and using those UML/UEDK operations, are empowered to handle more complex aspects of these operations because they can test and experience directly running solutions, instead of just being lectured about its features. We call such feedback loop learning reinforcement, "learning by designing".

## 3. Available Resources and Project Description

In order to achieve the desired results, three concurrent resource types, human, computational and pedagogical, must be articulated when introducing the UML/UEDK platform into a computer engineering or computer science curriculum.

From a computational viewpoint the current platform, uses the 0.2.0 UMLe-SDK (Universidade do Minho Layout environment Software Development Kit) implemented in Visual Basic (VB) and supported in the form of COM based DLL automation. Nonetheless, languages like VC++, Delphi, or others, for which an ActiveX component can be created can access this SDK. It provides basic data-structures, events, methods, graphical user interface (GUI), load and save operations, file formats, that can be accessed and are shared by any student built "autonomous" CAD operation. These operations are implemented as VB Add-ins, though .exe implementations (out-of-process) are also supported. An UMLe Add-in is a program, which performs tasks using its own resources as well as the UMLe environment resources, being compiled as an in-process ActiveX .dll component.

This SDK is a development environment, whose user accessible, and visible, interface is the UMLeAx layout editor, see Figure 1. This editor is "simply" an application case study, using the same environment resources available to the student Add-ins'. Moreover, even if independently installed, any UMLe Add-in "external" operation is available to the UML/UEDK CAD user as if native, i.e. as if originally compiled with the UMLeAx editor. Student operations are accessible through UMLeAx: menus, File New templates, or Toolbox buttons.

Besides an MS like Help documentation, the UMLe-SDK resources are: a Data Structure Server (DSS) and an eXtensibility Environment (xEnv) - supporting the shared data-structure and its methods. Plus an Extensibility Object Library (EOL) - where extensibility means the capacity to stretch the functionality of the UMLeAx seamlessly, adding operations, i.e. objects, previously unavailable.

Currently, the UMLe common resources only support VLSI physical design automation operations. But it can be extended to other computer engineering design areas. Indeed, one can see the UMLe computational paradigm as if it was a "dress closet". Currently the "closet" only has one door, that giving access to the physical design automation "clothes". Having the UMLeAx as its main rod, where several "coat hangers" - the Add-ins - permit the user to choose what to wear. The same paradigm, can equally support "closet doors" - DSS and xEnv - in other design areas.

From a human skill viewpoint we have had good results with sophmore students - in a five year degree - having no prior knowledge of VB or CAD/EDA VLSI - but having access to code examples within the UML/UEDK, and being introduced to key VLSI physical design concepts and algorithms. Basic knowledge in data-structures, and basic experience with procedural and object-oriented languages is needed.

A set of recitations should be set aside to present the UMLe environment, show running examples, and allow students to use, test, and "break", the UML/UEDK available operations. Because the goal is to provide hands-on experience in conceiving design automation operations, it is important - with sophmore students - that teachers partitioned each assigned small project in its key aspects. Providing guidance about the relative importance of different solution features. Encouraging students to ask questions as well as use related UML/UEDK operations.

From a pedagogical viewpoint, students should be encouraged to give intermediate presentations. Nonetheless, even in a "final" presentation of a project, teachers should seize the opportunity and focus on good and poor - user and design - features of the solution. Advising students on how to bring about major design improvements through minor code alterations, while identifying any important limitations from a VLSI design-flow usability viewpoint. UMLe project on-line support is provided in *http://gioconda.di.uminho.pt/UMLe*. It archives, UMLe-SDK versions; useful VB components; skeleton add-ins source code, which illustrate basic usage of the UMLe-SDK environment. In *http://gioconda.di.uminho.pt/UMLeSpecs*, the most current specifications of each UMLe-UEDK operation as well as needed book references, are available. Support and content for operational aspects related to those courses whose small projects use the UMLe-SDK, is made available at this site.

## 3. Results and Further Evolution

Assume a four month long course, where a student is given the assignment to implement, for instance, a zero-skew clock routing algorithm. Experience shows the amount of effort (time) students spend in creating the basic data-structure, the GUI, etc., diverts much of their effort away from the design aspects of the zero-skew clock routing itself! Time spent also in detriment of considering tool integration design aspects, run-time efficiency, design usefulness of the solution, or even trying variations over the textbook version. Moreover, little time, if any, is left available for actual testing of the solution presented, even against naive cases, let alone against adequate benchmarks.

The UML/UEDK helps reduce the development time, allowing the student to concentrate most of his/her efforts on the specific problem. Moreover, by requiring students to use UML/UEDK operations implemented by other students - some having

known limitations - our experience shows we accelerate the learning curve, fostering a critical attitude towards one own solutions, and developing "adequate design" intuition.

Some of the UMLe Add-in operations in UML/UEDK are the following. Layout wizards to automate layout cell design (namely BDDs and RAMs). Layout n/p network connectivity extraction (LVN). Capacitance and resistance extraction. Stick diagram editor (SDE) and viewer (SDV). Slicing floorplan representation and solution finding. Layout evaluators, to help users compare alternative layout solutions for cells. Dual Euler path identification in n/p networks, and corresponding single stripe layout. Optimization of layout polygons. Design rule checking (DRC). Algotronix like FPGA editor. Balanced clock routing H-trees. Format converters (MAGIC to UMLe and to CIF). Yoshimura and Dogleg channel routers, and others.

In view of all of the above, we believe the UML/UEDK platform is a useful learning environment which could easily be used within different universities course curriculum. Moreover, it is possible as well as desirable, to extend its CAD/EDA design scope behound that of VLSI physical design. In which case, cooperation between different universities would allow the UML/UEDK platform to proliferate along those aspects of digital design of most interest to the participating courses, teachers and students.

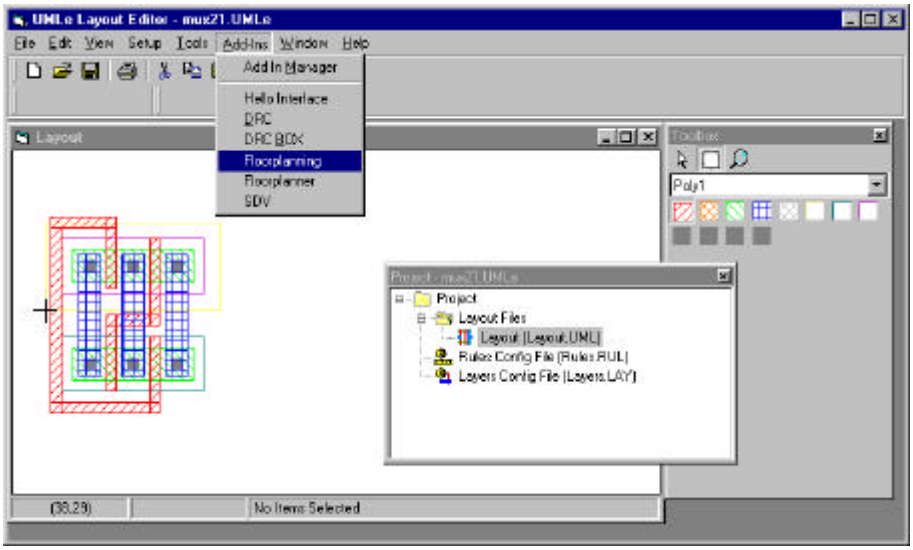


Figure 1: The UMLeAx, showing in the menu two floorplans, DRC, DRC box and SDV Add-ins.

## Acknowledgements

Besides the participating students, we would like to thank research and teaching assistants, Vitor Serafim, Jorge Braga, Vitor Pereira and Vitor Rodrigues for their contribution, in student support and in the development of features of this platform.

Finally, some of this work has only been possible thanks to a grant under FCT project N$^{o}$ 1668/95 of PRAXIS/JNICT.

## References

1. Lima, J., Braga, J., “The UMLe: A VLSI CAD/EDA Learning Environment”, (EDP/IEEE1999) Electronic Design Processing 1999, Monterey, CA, USA.