The UEDK: A VLSI CAD/EDA Learning-by-example Platform

José Augusto Domingues Fernandes Lima
jal@di.uminho.pt

Universidade do Minho,
Departamento de Informática, Braga, Portugal

Abstract

The UMLe-UEDK (Utilization Environment Development Kit) has evolved - as a natural outgrowth - of the UMLe-SDK (Universidade do Minho layout Learning Environment Software Development Kit)[1]. The UEDK is aimed at improving and extending UMLe usefulness as an introductory CAD/EDA apprenticeship "tool". By allowing students, as well as teachers, to learn from utilization as much as from SDK based development, the UEDK facilitates a reflection, over CAD/EDA design features. Indeed, given that Time to Market is critical for product success, and given that the industry faces an increasing demand for chip and tool design engineers, it is useful that interested students may be able to accelerate their learning curve in the CAD/EDA field. Though being a modest contribution, the UMLe-UEDK seeks to help shorten the learning curve gap, facilitating the learning efforts of prospective CAD/EDA engineers.

1 Introduction

The UMLe has been developed as an OPEN environment VLSI CAD/EDA platform, aiming at providing basic data-structures, methods and graphical user interface (GUI), needed to support "independent" development of operations by differentiated groups of students or researchers. These services are supported in the form of COM based DLL automation, compatible with applications supported under Borland, Microsoft, Rational, etc.

UMLe, hence UEDK, aims to facilitate an evolving - and as much as possible seamless - participation of different contributors. Allowing developers to concentrate upon the development of their specific CAD/EDA operation features.

The current environment - the UMLe SDK - has evolved from the UMLe-Alpha - first presented at IEEE/EDP99 [1]. The Alpha solution has a DLL automation data-structure backbone, enhanced with other *.dll* and ActiveX controls. All such UMLe resources can be used by any student built *.exe* CAD/EDA operation. The UMLe-SDK platform uses a Data Structure Server (DSS) together with an extensibility environment (*xEnv*) and an extensibility object library (*xObj*). All of these are shared by the Add-ins implementing each CAD/EDA operation. In fact registered Add-ins act as environment additional tools, appearing to the CAD/EDA user as if native to the environment.

Given the above, and since a reasonable number of UMLe CAD/EDA operations have already been implemented, the next step was to make these CAD/EDA operations available to users. This goal has been achieved by creating the UMLe-UEDK platform.

The objective is for the UEDK - as well as the supported UMLe Add-in paradigm - to constitute a useful tool development environment, aimed at furthering CAD/EDA apprenticeship at the Department of Computer Science and Engineering (Departamento de Informática). Ultimately, it is our expectation, similar objectives embodied in the utilization of the UEDK platform, could be of interest to other teaching and R&D communities.

A distinctive feature of the UEDK platform, versus its UMLe-Alpha predecessor, is the fact that it provides the “user” with Visual Basic Add-in source code examples, from available operations. The inclusion of these examples is intended to provide a faster lane for a quick UMLe-SDK development. Helping developers to focus themselves upon the design of the algorithm, and aspects like, adequacy of available user design options, friendliness of the GUI, quality of the solutions, etc., while abbreviating the programming language and UMLe-SDK programming features learning curve.

2 The UMLe-Alpha: A first approach

The UMLe-Alpha. was the first integrated release of the UMLe where a number of student developed CAD/EDA operations were made available. A UMLe Alpha project was a basic hierarchical layout comprising a *.UMLe* file and the corresponding *.UML* layout files. All operations were implemented as executable files,

and shared the use of a data structure, implemented as a DLL automation. Moreover, access to several *.ocx*, providing project support and integration (layout visualization, project structure, etc) was also provided.
The operations available in the UMLe-Alpha platform were of three major groups. Wizards allowing users to build layouts - RAM cells, Manchester adders, etc - editors/viewers providing their own simple GUI, like a Stick Diagram Editor, a FSM editor, and basic CAD/EDA operations implementing algorithms like Yoshimura and Khu channel router, or a layout evaluation operation, etc.

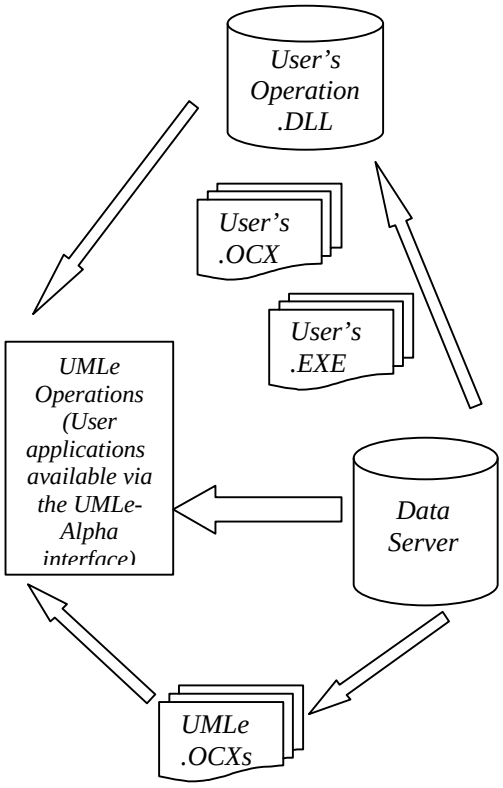

Figure 1: The development architecture of the UMLe-Alpha platform

3 The UMLe-SDK

As the result of the experience gathered in the utilization and support of UMLe-Alpha we concluded a tighter level of integration was advisable. For instance, students often did not comply with expected development practices, e.g. introducing format mismatches or "quick-fixes" into their *.exe* operations. Hence, the UMLe-SDK was created in order to avoid such inconsistencies, and provide a tighter and extended development integration of CAD/EDA operations. This SDK includes a development environment "designed around" the UMLe editor, such that user added operations can be made available as if originally built-in into the editor. Namely, the UMLe-SDK encompasses the following elements:

- A MS like Help, where SDK features and basic development documentation is provided.
- A Data Structure Server (DSS), now available as a server extensibility environment (*xEnviroment*).
- An Extensibility Object Library, where extensibility is the capacity to stretch the functionality of the UMLe editor, seamlessly adding operations previously unavailable.
- Unlike in UMLe-Alpha, operations are implemented as Add-ins, though .exe implementations (out-of-process) are still supported. In fact, an Add-in is a program which performs tasks within the UMLe development environment, and it is compiled as an in-process ActiveX *.dll* component.
- A layout editing capability was added in order to facilitate the support for certain student developed operations, like layout wizards. Also, an Add-in operation is made available in the environment by being added as a command button. Either as a tool button in the editor's menu bar, or as a menu entry in the Add-ins own menu.

Finally, and in order to help the appropriate development of Add-ins, some templates are available. They are the UMLe-Add-in template, and the UMLe template wizard. The latter makes available within the File-New menu a template for layouts of the type created by a specific operation, like the BDD-wizard, or the FPGA-wizard.
To provide an overall presentation of the project, as well as facilitate the access of students web sites and a CD-ROM have been created.

- In *http://gioconda.di.uminho.pt/UMLe*, the UMLe project is presented, access to downloads of SDK versions, as well as on-line documentation, etc, is provided.
- In *http://gioconda.di.uminho.pt/UMLeSpecs*, the most current descriptions of the specifications of each UMLe-UEDK operation are available. Also, support and content for operational aspects related to courses whose small projects are within the UMLe environment, is available at this site.

- The CD-ROM bundles, an SDK version, as well as documentation and layout samples to be used for testing.

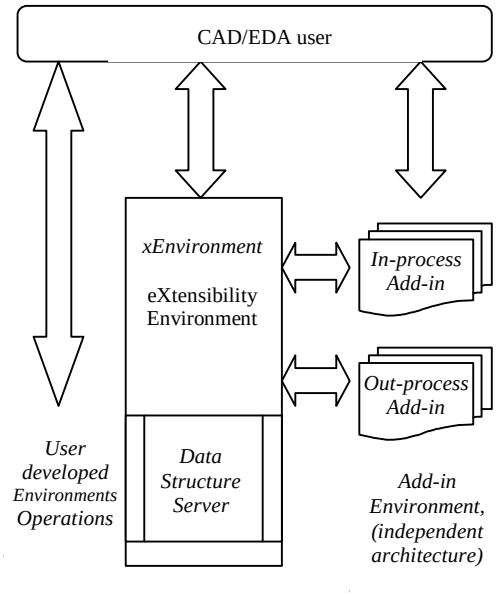

Figure 2: Current UMLe Extensibility Model development platform

Currently the DSS server implements mostly basic data structures needed for layout operations. Namely - and besides other features - it implements the UMLe, and its Add-ins data storage needs, as shown in Figure 3.

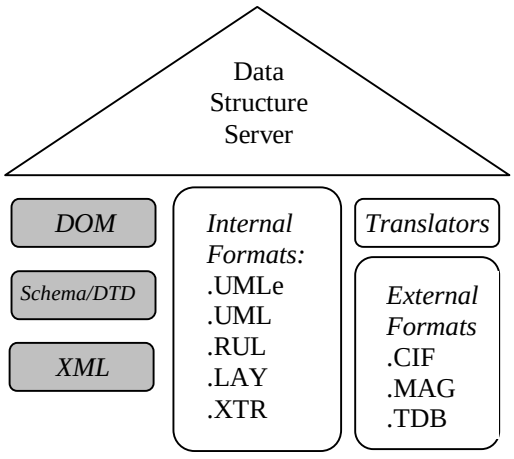

Figure 3: UMLe Data Storage structure, (items currently under development shown shaded).

4 UMLe challenges: the UEDK platform

Two major objectives have been envisioned when the UMLe idea was first proposed. One is to provide students, researchers, with an introductory hands-on approach to learn about CAD/EDA design. The other is to provide a platform where the work resulting from different school years becomes visible, and available, to next year students. Hence taking student's work out of the usual anonymity of course work school projects.

It was also expected that, by providing access and usage, of operations implemented by other students, some healthy awareness - even criticism - of good, and not so good, design options could naturally result. Namely, given that most of the specs of the CAD/EDA operations remain the same across consecutive school years. This self-teaching capability of good, but less than perfect, design solutions provides a learning opportunity based upon the analysis of other student's - if not your own - solutions' short comes.

Given the above objectives, one concludes that in order to be more useful to students, the UMLe project should encompass three complementary aspects. Namely, the technical – in order to improve the knowledge about CAD/EDA algorithms, metrics, cost functions, man machine interface, etc; the pedagogical – in order to provide an effective way of improving the quality of the design solutions - and the cooperative – in order to permit a good level of integration between features and operations developed independently.

Following the more development oriented UMLe-SDK, and in order to achieve the above objectives the natural evolution was the creation of the UMLe-UEDK. A more user oriented platform, intended to enhance the results, as well as resolve the pedagogical and cooperative short comes proper to an SDK.

5 Evaluation and Early Analysis

The different development phases of the UMLe platform have allowed us to learn how the different architectures and the addition of features have impacted upon the student's learning curve as well as the quality of the CAD/EDA operations implemented. In any of the phases some very good operations were developed. Nonetheless, the UMLe-Alpha required less of an effort from the students in terms of grasping the features of the environment

as well in terms of knowledge in VB. It permitted a quick-fix development approach, but was more dependent upon self-compliance of I/O formats, as well as the capability of constructing your own GUI. Alternatively, the UMLe-SDK is a much more elaborate environment, with many more features. Hence, it requires a more in-depth knowledge of the architecture of the DSS, of the xEnvironment and ultimately of the VB. Conversely it enforces strict compliance and compatibility of any operation with existing formats, UMLe built-in methods and classes.. Consequently the learning curve tends to be longer than before but problem free integration is much more likely.

Currently, the UEDK is designed to facilitate the UMLe-SDK environment and VB learning curve by providing standard examples, as well as allowing further development and enhancement of existing UMLe operations. This allows developers to concentrate upon algorithm efficiency and CAD/EDA operational, and design flow aspects, instead as much upon code construction.

6 UEDK Evolution

The current UEDK is supported by the UMLe-SDK, hence it is based upon COM and DLL automation Microsoft technologies. Nonetheless, work for Linux based platforms, namely the UMLe-Anneal and the UMLe-Corba client-server operations has been developed.

- The UMLe-Anneal engine allows students to submit over the web partition data, in order to obtain a TWPP solution. Alternatively you may instead submit your partition problem data to SAGA (the UMLe-Anneal meta-engine). SAGA runs a Genetic Algorithm (GA) engine which returns to the user good Simulated Annealing (SA) parameters for his/her own resolution of the TWPP problem.
- UMLe-Corba has been intended as a feasibility study for the evolution of the UMLe DSS into specialized data servers (possibly multi-platform). A development of this solution is intended to replace the current desktop based UEDK. The objective is to provide more computing power, increased flexibility and a distributed management of UMLe-DSS data-structure operations.

Also, current .UML, .XTR, etc. formats will adopt a DTD specified and XML based description, as illustrated` in Figure 3.

7 Future Work

In spite of evolution UMLe has undergone, there is nonetheless room for UMLe platform improvement. Indeed, in order to achieve a higher emphasis upon design flow, and CAD/EDA design methodology apprenticeship, some relevant changes should be introduced. Namely, and in order to make the platform more flexible, and illustrative of current design trends, Add-ins should run on remote machines, having ActiveX controls implementing their client browser interfaces. Indeed, in spite of the current MS based solution, Linux based important UMLe operations have been developed. Hence, the next step would be to evolve the UEDK into a web based, client server, CAD/EDA learning platform. This approach, allows implementation flexibility, while simultaneously providing UMLe users with a learning ground for a distributed, web based, CAD/EDA design flow. (please refer to Figure 4). Moreover, such a web-UEDK platform would provide users anywhere access to a learn-by-example CAD/EDA environment. Consequently, an expected emphasis of the UEDK platform evolution is the exploration of web based design flows and man-machine CAD/EDA web design interfacing.

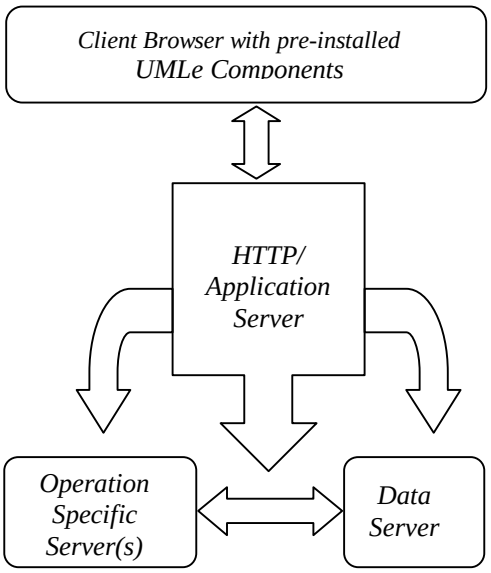

Figure 4: A web based UMLe architecture

Acknowledgments

The above work has had the participation of many students, researchers and teaching assistants. Namely, we would like to thank for

their diverse roles into the development of UMLe and UEDK: Vítor Serafim Pereira, Jorge Duarte Braga, Dario Teixeira, Vitor Pereira, *Ph.D.* and Sérgio Araújo.
Finally, some of this work has only been possible thanks to a grant under *project Nº 1668/95* of PRAXIS/JNICT.

References

[1] Lima, José Augusto, "The UMLe: A VLSI CAD/EDA Learning Environment", IEEE/EDP DATC 1999, Monterey, CA, USA.

Figure 5: View of the UMLe-Alpha first release. Showing a few of the student implemented operations.

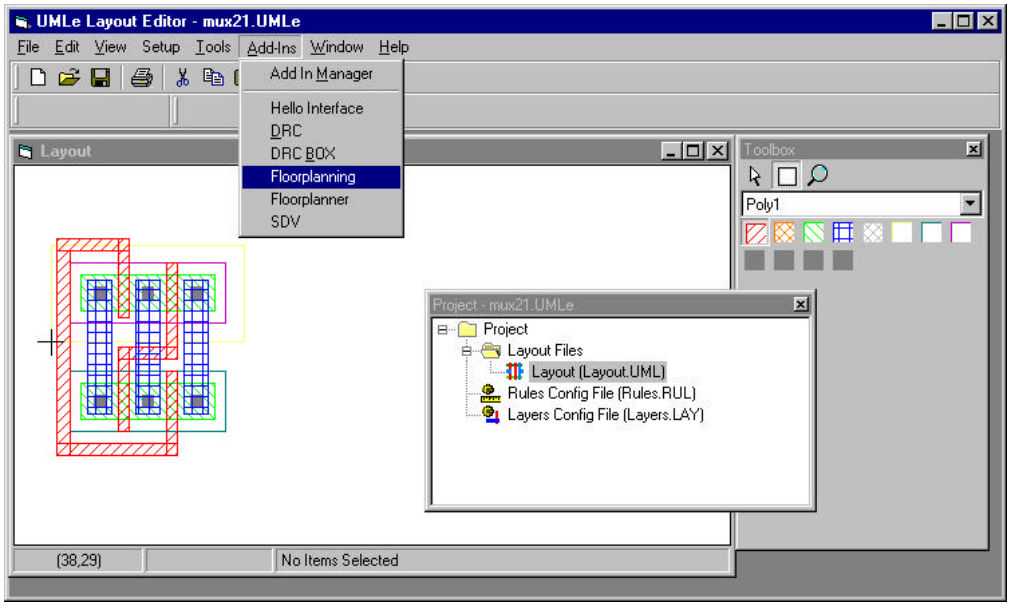

Figure 6: View of the UMLe-SDK editor. Shown is a multiplexer in the layout project window, the toolbar window, and the project window with its .RUL and .LAY configuration files as well as the hierarchy of the layout structure. Also shown are some of the Add-in operations registered: 2 interactive Floorplan operations, a DRC and its DRC box, an SDV- Stick Diagram viewer).

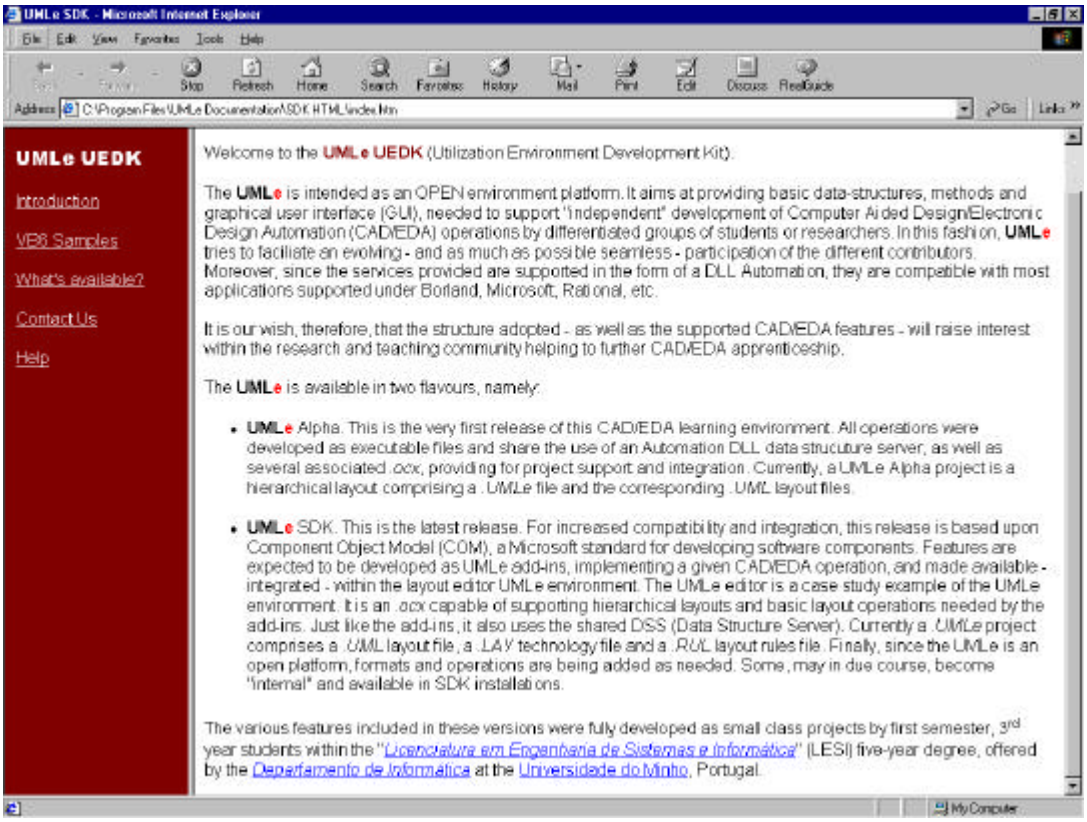

Figure 7: Above, the UMLe-UEDK presentation page provides in "What's available?", any of the UMLe-Alpha, *.exe* operations, as well as any of the *Add-ins* registered in the UMLe-SDK Editor. Below, the UMLe Help system is illustrated, showing the Extensibility Object Library partially expanded.