ABSTRACT

The "NMP-CADLAB framework" incorporates a OO/ER4 high performance user interface management system, a distributed data handling system for heterogeneous computer networks and a rich set of services for step-wise tool integration. The paper gives an overview of the framework development and its relation to the existing "-CAD and CADLAB software.

Section 1 of this paper describes the general problem area, ending up with a basic conceptual model for frameworks and environments (for a more detailed discussion of this topic see /7/). With this as background, section 2 explains the NMP-CADLAB Framework and its services in more detail. The paper finishes with a short report on status and planning (section 4) and conclusions (section 5).

1. FRAMEWORKS AND ENVIRONMENTS

1.1. Problem

When looking at the designer's tool environment and the people giving support to him, a number of problems become apparent.

Typically the designer will find quite a diversity of design tools at his command, which are difficult to use, incomplete and inefficient in their combined application. They are lacking a consistent user-interface, uniform control philosophy and integrated data-handling.

Looking into the designer's support department, which is supposed to supply the designer with an efficient, safe design environment and which has to provide mechanisms to ensure that all relevant data can be maintained and handled in an consistent way, we find that supporters are seriously lacking tools and services to aid them in this difficult situation.

In the management area analogous problems are seen. Presently possibilities for configuring environments to specific projects are severely limited. Functionality for configuring teams, responsibilities, allowed tools and demanded procedures is missing. It is difficult to store and retrieve management information such as status, cost and planning data and to associate it with the respective design information. Tools (in the case that they are available) often resides isolated on the manager's PC.

1.2. Basic Definitions

Let us define the following terms:

Definition 1: "CAX" is a place holder for CAD, CAE, CAM or CASE. "Services" are any kind of aids that support a user in doing his job (i.e. services include methods, tools, data etc.).

Definition 2: A "CAX-framework" is a general computer-based infrastructure, which offers services for efficiently building, maintaining and configuring "integrated CAX-environments".

Definition 3: An "integrated CAX-environment" is a collection of computer-based, mutually coupled services for supporting all phases and aspects of the CAX-process in a user-friendly, continuous way.

Note, that the term "integrated CAX-environment" often is used to refer to various specific types of environment. We suggest to distinguish "standard" environments, which are produced for a wide market containing standard tools and services, from "custom" environments which, in addition, offer special tools and services tuned to the specific need and application domain of a certain company or design department. Moreover, it makes sense to distinguish "generic" environments, which contain the environment services in a generic non-configured form, from "instantiated" environments, which are instantiated and contain all services in their final executable form. In a similar way, we should distinguish "generic" from "instantiated" frameworks or framework components.
1.3. Requirements

It is clear that the requirements we have to place on frameworks are strongly influenced by the requirements integrated environments have to fulfill.

The main requirements for integrated CAX-environments (see also /8/) are summarized by the following key words:

A CAX-environment shall have a consistent and uniform tool handling with design process control facilities. The set of tools, forming the instantiated and functionally complete environment, shall have consistent and intuitive user interfaces.

A CAX-environment shall have an integrated handling of distributed data allowing for concurrent access. The CAX-environment shall be open and configurable, where environment software is of high performance and portable.

More specifically, we have to demand from the underlying CAX-framework:

A CAX-framework shall supply building blocks for homogenous user interfaces, integrated data-handling systems, tool integration and design process control.

A CAX-framework shall be open and flexible, supporting a spectrum of integration depths (from low to high), have relevant interfaces that are open and public, include configuration mechanisms and be scaleable for various team and project sizes.

A CAX-framework shall provide security and robustness (against break-down and misuse) by providing protection and exclusion mechanisms, management of access privileges, transaction management and recovery services. Further allow concurrent access (multiuser/multitasking) and resource sharing.

A CAX-environment shall have general availability, and provide high performance and efficiency directed for global optimum but also provide local tuning options.

1.4. Conceptual Model

In order to discuss, how integrated environments can efficiently be built with the aid of frameworks let us consider a conceptual model of an integrated CAX-environment (see Fig. 1).

The model considers an environment as a layered architecture of services with the most application specific services at the top and the most general services at the bottom. Hence, in the lower level layers, we find those services, which have been identified as being common to a large number of standard, custom, generic as well as instantiated environments.

The main layers considered here are:

1. Hardware and Native OS Layer
2. Kernel Environment Support Layer
3. Application Integration and Development Layer
4. Application Layer

As indicated in Fig. 1., these layers may contain sublayers.

To a first approximation the given model can also be seen as that of a multi-layered virtual machine, where higher levels of implementation are supported by lower ones /9/. However, in deviation from the given drawing this view would demand that some of the integration services which have now been associated with layer 3 have to reside on top of layer 4.

As already pointed out above, we have to make a distinction between generic and instantiated environment components. This implies, that the services displayed in Fig. 1 do not simply represent executable, instantiated code. Rather a major part of the services is given by - what we will call - "instantiation mechanisms", which, when applied to "instantiation information" as input will produce "instantiated components" as output.

The given conceptual model is suggestive of how an environment can be economically built up, in so far, as it factors out more basic lower level services on top of which a diversity of more specific higher level services and hence environments can be built. Thereby different environments are mainly characterized by their different application specific services. Thus, the remaining lower level services make up the framework.
Note, that we suggest, that a complete framework - besides offering layers 1 to 3 - should also contain the domain-neutral end-user services of layer 4 (for reasons of economy). Note also, that our definition of a framework includes hardware services, which - in the given model - have been concealed inside layer 1. The focus of framework developments, however, is in the area of "generic framework software", which should be as hardware-independent as possible.

2. The NMP–CADLAB FRAMEWORK

2.1. Software Architecture

The NMP-CADLAB framework consist of a family of components. One component may vertically span several of the conceptual layers described in the previous section. One customer environment may consist of a selection of components from the framework product family. Most of the components may be divided further into a run-time and a development part. The following paragraphs describes briefly the services offered by the different framework components.

2.2. Portability Services and Virtual OS

The framework is developed according to ISO and IEEE Software Quality Assurance Plans. The framework provides a virtual operating system according to the POSIX /10/ and X/OPEN /11/ definitions. This is a thin layer available to the application (and framework) developer. This layer also guarantees the portability of the framework itself. High-level communication services on remote procedure call level are provided by building on the NCS (Network Computing System) family of products from Apollo Computer Inc.

2.3. General Data handling Services

The framework offers an ODBMS, Object-Oriented Data-Base Management System. The ODBMS is optimized for integrated CAX-environments where large and long-lasting transactions are common. It provides heterogeneous distributed access to and storage of data. The ODBMS builds on the ERA (Entity-Relationship –Attribute) and OO (Object-Oriented) approach. In literature this kind of DBMS is often called structural object oriented. Data can be described and accessed on a very detailed entity level (e.g. rectangle, x-coordinate) or on a "black-box" or "file" level ("clusters" or "complex objects"). "Clusters" may be versions managed by the ODBMS.

2.4. User Interface Services

The framework offers both graphics libraries and development tools to make flexible and modifiable user interfaces to application tools. The generic name for this component is UIMS, User Interface Management System. The UI (User Interface) of a tool may be divided into three layers, the presentation component (i.e. graphics libraries, "widgets"), the dialog component (i.e. dialog structure definition) and the application interface component (i.e. the interface to the behaviour or functionality of the application tool itself). The UIMS provides run-time and development support based on this architecture.

Figure 3 has been introduced by the Open Software Foundation (OSF) and shows where OSF's standard (Motif) and UIMS fits. WM stands for Window Manager and is a utility in the framework. X-Windows is used as a basis /12/.

FIGURE 2. The ODBMS architecture

The application developer is offered two sets of interfaces to the ODBMS, a procedural and a descriptive one. The procedural interface, IDM /6/, acts both as a DDL (Data Definition Language) and DML (Data Manipulation Language). The descriptive interfaces /5/ are C-DDL and NQL (Navigational Query Language). NQL is an embedded language, currently supporting C as its host language. The ODBMS architecture is outlined in figure 2.
2.5. Capability Services

Based on the UI and DH-services layer, a set of capability services are offered. Typically this is a set of interfaces to the Design Management Model (DMM) /16/, /18/. Packaging results into Domain Neutral Services. The services available, are the procedural interfaces used by services on the next higher levels in the conceptual model.

2.6. Application Integration and Development Services

For developing specific classes of applications having the same base schema, toolboxes or libraries are offered that bridge the User Interface and Data Handling parts together. Such toolboxes always build on some conceptual model (or schema). Examples of toolboxes offered by the framework is and OIS ("Integration Services" /14/), which allows easy "black-box" ("encapsulation") integrations of tools.

The framework offers a methodology and support for developing applications. There are consistent and common conflict and error handling components.

2.7. Domain Application Neutral Services

The Capability Services may be packaged into domain neutral applications, according to some process and user model (i.e. "instantiated"). An example packaging (which also implies an example UI) is provided with the framework.

Examples of such services (i.e. tools) are Query Tools on Design Management Data, a general Browser to query and modify contents of the ODBMS and ODBMS utilities (e.g. managing and monitoring of the ODBMS, setting access rights, backup etc.). Other examples are a Configuration and Version Management tool and a tool to support a project (e.g. development phases, task assignment, change request handling).

A "desktop component" is an example of another generic service. It provides an entry to end-user environments from which tools may be started. a ("CAD-shell").

3. BACKGROUND

NMP-CAD is the name of a project jointly sponsored by the Swedish Government, through the Swedish Telecom Group and the Swedish Defence Administration, and Swedish industry through Ericsson Telecom, Ericsson Radar Electronics, SAAB, FFV and TeleLOGIC (a subsidiary of TeleSoft Inc.). The framework development started in the late 1985s, and continues with a second phase starting in 1988 with additional sponsors. There are about 25 engineers (M.Sc degree) working in the development group. TeleLOGIC, a subsidiary of the Swedish Telecom group, is responsible for support and marketing of the framework product version, which is called Base/OPEN. Here another 10 people are working since late 1988.

The Computer Aided Design Laboratory CADLAB is a joint venture of Nixdorf Computer AG and Paderborn University. Founded in 1985, Cadlab provides research and development in the area of computer aided design of integrated circuits and systems with a special focus on frameworks and tool integration. There are eight project groups with an approximate total of 55 scientists and engineers, 10 M.S. students and 70 student assistants, half of them being engaged in the framework domain.

Motivated by the apparent need of open integration platforms, both NMP-CAD and CADLAB independently took up framework developments early 1986. The developments were based on a careful analysis of available research and development results (a.o. /1, 2, 3, 4/) leading to the NMP-CAD "Base/OPEN Framework" /5/ and the CADLAB "CWS Framework" respectively. In 1987, a first cooperation started in the context of the European /6/ "SECT" initiative ("Software Environment for CAD Tools"), where NMP chaired the development of working standards for User Interface Systems and CADLAB chaired the Data Handling Standards group. The large similarity of the NMP-CAD and CADLAB approaches was recognized and further extended.

In order to maximally exploit the common resources and in order to offer a broadly-supported optimized framework to the user, NMP-CAD and CADLAB decided to merge their framework developments. This joint "NMP-CADLAB Framework" provides an efficient platform for tool integration and development in the CAD, CAE, CASE, CAM areas. It incorporates a user interface management system, a distributed data handling system for heterogeneous computer networks and a rich set of services for step-wise environment integration.
4. STATUS AND PLANNING

The functionality of the NMP-CADLAB framework is already available today. However, at present it is partially distributed over the current NMP-CAD and CADLAB frameworks. A first version of the merged NMP-CADLAB framework will be available Q1 1990 for pilot use. A full release will be available in Q1 1991 on hardware from Apollo, Digital, HP, Nixdorf and Sun. The NMP-CADLAB framework will also be made available to the JESSI Framework project.

Note, that a product version of Base/OPEN containing significant features of the NMP-CADLAB framework is available today through the Telelogic company [17]. The NMP-CADLAB framework will be upwards compatible with this product.

5. CONCLUSIONS

With their existing frameworks, NMP-CAD and CADLAB have already shown that efficient framework implementations are possible and useful. The new NMP-CADLAB Framework will offer an even broader range of optimized services for efficiently building, maintaining and configuring integrated environments in the CAD, CAE, CASE and CAM areas. The NMP-CADLAB Framework provides an open platform for various levels of tool integration and development. It thus constitutes a major step towards a broadly applicable standard framework.

6. ACKNOWLEDGEMENTS

A list of all those who have contributed to this work would be too long. We give our acknowledgements to all developers at NMP-CAD and CADLAB without whose participation this work should not have been possible. Especially we would like to thank Sören Altemark, Gerd Kachel, Thomas Kathöfer, Johanna Lorentzi, Julia Miller, Kit Molander, Bernd Nelke, Håkan Söderström and Charles White for their effective assistance in technically guiding the preparation of the merger, and Peter Weissglas, Conny Björkvall and Wilfried Hauenschild for their central contributions to defining the overall concepts of the cooperation.

REFERENCES