21

Integrating Applications into a Virtual Prototyping Environment

Uwe Jasnoch, Holger Kress, Joachim Rix
Fraunhofer-Institut für Graphische Datenverarbeitung
Wilhelminenstr. 7
64283 Darmstadt - Germany -
Tel. : +49 6151 155 (245 | 212 | 221)
Fax: +49 6151 155 299
Email : (jasnoch | kress | rix)@igd.fhg.de

Abstract

The paper will show and discuss different strategies of integrating applications into a Virtual Prototyping Environment. The ability to rapidly prototype a proposed design in a Virtual Environment is becoming a key contributor towards fulfilling the business requirements embodied in a short time-to-market, in cost-effective and high quality manufacturing, and in easy support and maintenance. The ability of integrating already existing applications into such an environment is fundamental. The paper first clarifies the different notations used for the different forms of integration. Afterwards a communication system as a basis for integration is proposed, which tackles most of the problems concerning the communication between heterogeneous applications. Finally, different strategies to integrate an already existing CAD system are shown and discussed.

Keywords

Virtual Prototyping, Integration Strategies, Communication

1 INTRODUCTION

Virtual Prototyping Environments gain more and more importance. In complex virtual prototyping environments, frameworks are often used as the software basis for these environments. As in CoConut , a framework is a computing environment which for itself is not very useful unless a number of applications are integrated. The phrase "tool/application integration" has

J. Rix et al. (eds.), *Virtual Prototyping*

been widely used in the beginning. It has different meanings to different people, as stated in . Some distinguish between loose integration and tight integration, black-box integration and white-box integration, or integrated tools versus attached tools „. In the past, a better understanding of integration led to the distinction between three "kinds" of integration: white-box, grey-box, and black-box integration .

These kinds have to be seen in the context of three basic areas: control, user interface, and data. This leads to a three-dimensional quantity of integration as shown in figure . As a mini-

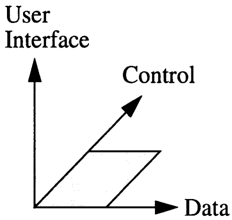

Figure 1: Integration Measurement

mum, a framework has to have some control over the application in order to start and stop the application. For data integration, a certain knowledge about the data structure is necessary. The dotted area of figure indicates a possible degree of integration, where the framework has the knowledge of control and some knowledge of data. In this example, the application does not use any user interface service. With the help of this figure, the different kinds of integration become more clear: white-box integration means the maximum expansion on a certain axis. Grey-box integration is an expansion between the maximum and the origin. Black-box integration means no expansion on a certain axis. But at least, to have a minimum control over the application, some expansion on the control axis is necessary.

Nevertheless, white-box integration is only possible for applications, where the source-code is modifiable or when applications are designed with the knowledge of the provided services in mind. For applications, which are not designed to operate in a framework environment, the phrase "tool/application encapsulation" has been established for the integration. From the data integration point of view encapsulation means to have a wrapper for converting the data between the framework schema and the application schema and the conversion of the access functionality. Data integration itself, is the ability to share information throughout the environment . For an open design environment, data interchange between several applications is fundamental. This means in the real environment, on-line data interchange via the data management service of the underlying framework. On the other hand, a flexible Communication System is needed, to enable the control integration of the new application.

This paper first discuss some major issues of the communication system as a basis for the control integration of an application. Afterwards, the integration of an already existing CAD system into CoConut is discussed as an example of a complex integration task. The paper ends with a conclusion.

2 THE COMMUNICATION SYSTEM AS A BASIS FOR CONTROL

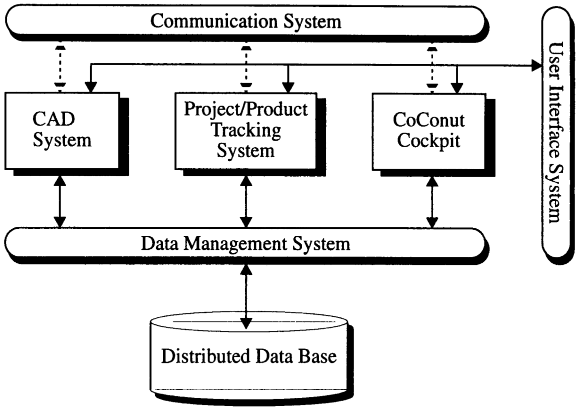

Figure 2: The CoConut Environment

INTEGRATION

The task of the communication system is to deliver messages between different application and/or framework components through the environment. Hereby, the different scopes (the user scope, the environment local scope, and the environment global scope) of the messages must be supported by the communication system, according to the purpose of the message. One major requirement is the independency between the applications. This means that an application must be able to send a message, without knowing the receiver in detail. The application only requests a certain task or gives information about an event, without the knowledge of the concreie partner. This independency is the basis of the integration of new applications in a heterogeneous environment like CoConut.

In a heterogeneous environment the application for solving a certain task could change for the different platforms. This does not mean changing only the code due to the different architecture, this also includes the usage of a completely different application. In our first prototype environment, the design task is handled on an IBM with the CAD system CATIA, while on a SUN this task is solved by the CAD system COSMOS. CoConut must be able to decide based on the integrated applications and the environment to start the correct one after receiving a message from an application indicating the request for designing an object. Taking this into account, the independency between applications regarding the communication is substantial.

One of the basic applications in CoConut is the cockpit. The cockpit starts the necessary applications and monitors the environment state for a user. Whenever an application needs an additional support by another application, it only sends a message and the cockpit knows how to react. If an application which solves the request does not run, the cockpit will start it. Figure presents an overview of the environment and the integrated application in this context. Here, there we do not distinguish between the dependencies of computer architectures and applications. These are hidden as much as possible for the end-user by the environment.

The basis for this controlling actions of the cockpit is the message service provided by the communication system. With the help of this service, the cockpit is informed about the requests and knows on the other hand about currently running applications. When a new application is integrated, there exists an assignment between the messages this application can react on and which messages this application could send. As a minimum, the messages for starting and stopping the application should be present. If these messages are not present, a wrapper must be provided delivering these messages. The benefit of this loose coupling between the application is that the cockpit could decide by its own which application could solve the request based on the current environment.

3 INTEGRATION STRATEGIES FOR A CAD SYSTEM

We have chosen the CAD system CATIA (Dassault Systemes, France) as an example for an integration of an existing application. The system is a characteristic representative for a grey-box integration strategy. As a commercial system the source code of the system is not available and therefore changes to the basic architecture of the system are impossible. Consequently, the integration strategy is focused on the provided functionality of the system's API and the services of the framework.

Three areas of integration will be considered: the integration on the user level, the integration aspects of the information model and of the data management.

Integration on the user level

One major goal of the framework is the consistent handling of heterogeneous applications in the Virtual Prototyping Environment. This is achieved by the encapsulation of the applications and by the control mechanisms of the framework. The consequence for the user is, that his application is not longer a pure single user system. In order to fulfil the integration issues the framework provides different services which have to be used. These services simplify the interaction and communication within the environment.

The most obvious service is the initial login-procedure entering the framework's cockpit. This procedure gives the user access to his personal working environment and makes his presence known to the other participants in the environment. After this he is able to start applications using the start-up service of the cockpit. To launch the CAD system, several ways of interaction are possible. Considering the idea of an integrated working environment, an interaction can only be done in a specific context. For a CAD system the context is the design of an assembly or a single part. Therefore the start-up procedure can be activated e.g. in an application which visualizes the product structure of the current product. The CAD system is started and the cockpit initiates the loading of the selected assembly or part into the CAD system. This example shows, how the usual dialogue using a load menu and selecting files is altered into a context-dependent selection of objects. The load procedure is done completely by the framework without further interaction of the user. This includes conversion between different formats and the extraction of the data out of the global into local data repositories.

For interactions during a design session the user interface of the CAD system is extended using the API functions. CATIA provides two interfaces for this: the IUA (Interactive User Access) language and the GII (Graphics Interactive Interface) interface. Both can be used to provide a specific user interface component for the communication within the environment, e.g. additional load operations, interim and end of session write operations, or communication services.

Integration aspects of the information model

The integration on the data level depends on the availability of a common data repository based on a standardized information model, which can be accessed by all involved persons. Therefore, CoConut is based on an underlying object-oriented model, which was developed by TC184/SC4 of the International Standardization Organization (ISO). This model is part of the ISO standard 10303 (STEP) . STEP is an acronym for „Standard for the Exchange and the Representation of Product Model Data" and characterizes the emerging standard for the exchange of product model data. STEP is the only standard which offers an information model covering various phases in the life-cycle of a product.

In the CoConut approach we use an information model which is especially developed for the area of the design of mechanical products with a three-dimensional shape representation. This model is standardized as document number 203 of ISO-10303 with the title „Application Protocol 203: Configuration Controlled Design" . In detail, the application protocol specifies the structure and the versions of complex mechanical products and their use in assemblies. A configuration management model gives the possibility to define different configurations of the same product. The design process is described as a chain of actions which are started by a start or change request. These requests are approved by authorized persons who are described by an organization model. The following shape representations are covered by the geometrical and topological model of the application protocol: wireframe representation, surface representation, facetted boundary-representation, advanced boundary-representation and solid models. Other information about security classification, contracts, documents or certification is also covered by the model. As the application protocol defines all necessary information used in the design process of mechanical products it can serve as the common information model for the CoConut environment, which ensures the integrity and consistency of the product data.

Each application in the environment handles only subsets of this information model. A project management application manipulates a subset which includes the information elements for e.g. project progress, approval status, and person and organization related aspects. The Product Structure Tracking System uses the information about the assemblies and parts of the product, and the CAD system deals with all information elements which are related to the shape of the product. This includes primarily the geometric and topological representation of the shape. Therefore the underlying information model has to allow application-dependent subsetting and the data management system must be able to handle these subsets in a consistent and uncontradictory way. For this purpose the data management system supports the data sharing with specific data converters for every application. The converters transform the application data from their native format to the neutral STEP format of the environment and vice versa.

Integration aspects of the data management

The data management system depends highly on the services of the object-oriented data base management system (OODBMS) of the environment. The OODBMS provides logging mechanisms, access to the data, and ensures the consistency of the data sets. In a heterogeneous environment different clients of the data base can run on various hardware platforms and enable a distributed access to the common data base. Figure shows a configuration in which the data sharing is completely handled by the OODBMS. No further communication links are necessary.

In case of grey-box integrated applications which are not able to directly interact with the OODBMS, other integration strategies must be provided. Figure gives an overview over the proposed solutions.

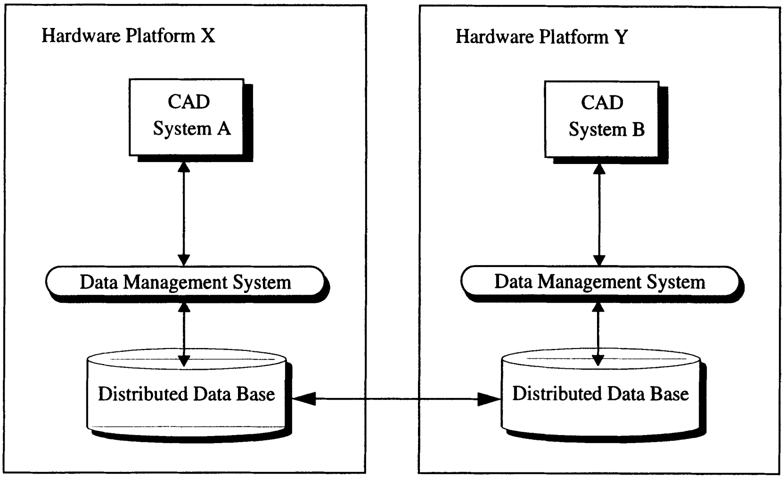

Figure 3: Data Management using a Distributed Data Base on heterogeneous Platforms

The API of the CAD system CATIA offers a set of functions called CATGEO subroutines used to add, modify or read basic data of the CAD system. This API is used to establish the connection to the data base. Three basic ways have been identified: a RPC interface, a physical file exchange, and the access to STEP working form repositories via NFS.

The RPC interface gives the functions of the data management system access to the functionality of the API subroutines. The approach follows the standard RPC implementation on heterogeneous platforms.

In the second case a sequential, ASCII file is used to exchange the CAD data among the different platforms. The STEP working form management system converts the native CAD data into instances of an intermediate, internal format. Accessing the instances of the internal format, the file formatter/reader generates the physical file according to the ISO specification 10303-11 . This file can be transferred via standard file transfer mechanisms. The receiving system generates with the file reader the internal format and transfers the data to the data management system.

The working form management system can store the instances of its internal format in an own data repository. This repository can be directly reached via NFS to pass the data to the data management system.

4 CONCLUSION

The paper described the communication system and the integration strategies for existing applications in CoConut, a system which supports the realization of parallel product development methods in the design phase. The described system contains several major features. One feature is the availability of a common, distributed data repository, which stores the relevant data of the design process according to the Application Protocol 203 *Configuration Controlled Design* of the ISO 10303 Standard STEP. The adaption of standards is a central objective in

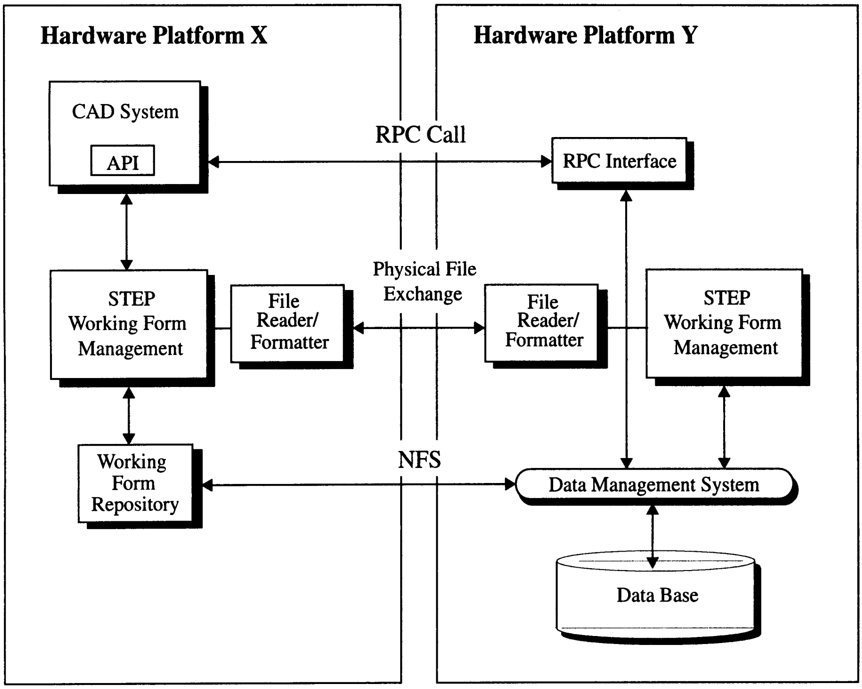

Figure 4: Data Management using a Data Base on a single Hardware Platform

CoConut and ensures the openness of CoConut for future developments.

The CoConut system represents an open, extendable framework for the integration of engineering applications. Future extensions to the system can address the underlying information model as well as the integration of applications which support specific tasks in the process chain of product development, such as kinematic or FEM analysis. The underlying data model can be extended by integrating additional STEP schemata. Thus, the whole data model will develop in the direction of an integrated product model.

The described CoConut environment integrates existing applications such as a CAD system in a heterogeneous computing environment. Nevertheless, the concept is open for the support of the whole engineering process which embraces the Virtual Prototyping Process.

5 REFERENCES

CFI Architecture Committee: „ *CAD Frameworks Users, Goals, and Objectives*“, Version 0.92, 1990.

CFI Architecture Committee: „*Framework Architecture Reference*“, Version 0.87, 1991.

European Computer Manufacturers Association: „*Reference Model for Frameworks of Soft-*

ware Engineering Environments", ECMA TR/55, 1991.

U. Jasnoch, H. Kress, K. Schroeder, M. Ungerer: *„CoConut: Computer Support for Concurrent Design Using STEP"*, in Proc. of the IEEE Third Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, Morgantown, 1994.

T. Kathöfer et. al.: *„The Architecture of the Object Management System within the Cadlab Framework*", in Rammig, Waxman (Eds.): Electronic Design Automation Frameworks, North Holland, 1991.

N. Kraft: *„Embedded Tool Encapsulation*", in Rammig, Waxman (Eds.): Electronic Design Automation Frameworks, North Holland, 1991.

M. Lacriox, M. Vanhoedenaghe: *„Tool Integration in an Open Environment*", Proc of the European Software Engineering Conference 1989, p 311-323, 1989

B. Prasad, R. S. Morenc, and R. M. Rangan: „Information Management for Concurrent Engineering: Research Issues", Concurrent Engineering: Research and Applications (1993) 1, 3-20, Academis Press, London 1993

ISO IS 10303-1: „Industrial automation systems - Product data representation and exchange - Part 1: Overview and fundamental principles"; International Organization for Standardization; Geneve (Switzerland); 1994

ISO DIS 10303-21: "Industrial automation systems - Product data representation and exchange - Part 21: Implementation methods: Clear text encoding of the exchange structure"; International Organization for Standardization; Geneve (Switzerland), 1993.

ISO DIS 10303-203: *„Industrial automation systems - Product data representation and exchange - Part 203: Application protocol: Configuration controlled design"*; International Organization for Standardization; Geneve (Switzerland); 1993

K.J. Cleetus: „Virtual Team Framework and Support Technology"; published in „Concurrent Engineering: Tools and Technologies for Mechanical System Design"; ed. by E.J.Haug; Springer 1993

6 BIOGRAPHY

Uwe Jasnoch received his university diploma in Computer Science from the Technical University of Darmstadt in 1989. Then, he was a software engineer with Philips Kommunikations Industrie AG for one year. Afterwards, Uwe Jasnoch was a researcher with the Interactive Graphics Systems Group at the Technical University of Darmstadt, where he was involved in several R&D projects. Since 1992, he has been a researcher with the Fraunhofer Institute for Computer Graphics in the Industrial Applications Department. His main research topics are data modeling, open environments, and consistency management.

Holger Kress is a researcher in the Industrial Applications Department of the Fraunhofer Institute for Computer Graphics since 1991. He is currently involved in research projects in the area of product modeling, groupware, and CAD frameworks. He received a masters degree in mechanical engineering from the Technical University of Darmstadt in 1991. His research interests include concurrent engineering, product modeling, and computer supported cooperative work.

Dr. Joachim Rix is head of the department for Industrial Applications of the Fraunhofer Institute for Computer Graphics (IGD) in Darmstadt, Germany. From 1991 to 1992 he was Associate Manager of the Fraunhofer Computer Graphics Research Group (today; Fraunhofer Center for Research in Computer Graphics, Inc. (CRCG) in Providence, RI). Mr. Rix received his Diploma and Ph.D. in Computer Science from the University of Darmstadt. His topics of interest are in Computer Graphics, CSCW, CAD, and Product Modelling. This includes the integration and use of computer graphics with its presentation and interaction techniques in industrial applications, like CAD, CAM, Concurrent Engineering , and Groupwork Computing. Mr. Rix is member of the standards Committee of ISO/IEC JTC1/SC24 „Computer Graphics“ and was Rapporteur of the study group „PREMO“ (Presentation Environments for Multimedia Objects). Since 1985 he is member of the national and international committees (DIN NAM 96.4, ISO TC184/SC4) developping STEP (Standard for the Exchange and Representation of Product Model Data). Since 1994 he holds the position of a deputy convenor of its WG 3 „Product Modeling“. Since 1981 Joachim Rix is member of the Eugrographics Association.