Panel Section

Engineering Information Systems:

Builders and Designers Perspective

Nick Roussopoulos

Department of Computer Science
University of Maryland
College Park, Maryland 20742

List of panelists:

Anthony Gadient,	VHSIC Program Office, USAF
Randy Katz,	University of California, Berkeley
Katie Rotzell,	MCC
Mary Loomis,	CALMA/GE
Raymond Lorie,	IBM Research
Gio Wiederhold,	Stanford University

1. Introduction

The primary goal of an **Engineering Information System (EIS)** is to provide a framework for the integration of engineering processes through computer aided engineering An engineering process consists of the conceptualization, design, development, testing, qualification (verification), fabrication, and maintenance of engineering products for which form, fit and function is controlled The management of the complex and continuously evolving databases consisting of data objects associated with engineering processes is the key concept in this framework and the integration of special purpose engineering tools with database technology will be the foundation of it

An EIS must support **reuse, evolution** and **trace** of information These are the most common practices in any engineering process and they constitute the foundation of technology evolution Reuse of designs and the acquired know-how is currently limited to small groups of people at most Technology transfer among these groups is often so time consuming that it cannot catch-up with its own evolution Dynamic models which can acquire new information and/or trace back to the development of the technology are needed

This extended abstract examines the database support requirements of an EIS We classified them into four categories object data types, database design methodologies, information management and control, interoperability of environments, and architectures

2. Object Data Types

An EIS deals with abstract data types representing objects, **Object Data Types, ODT** Many of these require specialized software for storage, display, and processing ODTs hide the internal structure of compound objects from external access operators and leave all the intra-processing to special purpose operators Objects and inter-object relationships are stored, accessed, and managed by a database system that is suitable for handling meta-data Schematics, graphs, images, rules, windows, texts, spreadsheets, voice, video, etc , are some of these ODTs and simulators, plotters, tailored access methods, etc , make up some of the

software for processing This list is by no means exhaustive and it only samples generic types of software which abounds current engineering applications

Redundant multiple representations of the same data is necessary to support the multiplicity of tools and users Managing collections of these advanced objects and their meta-data is different than managing atomic data entities in a database system Often meta-data is needed to be described by meta-meta-data and so on Typically, meta-data, referred to as the intension of the database, is used to interpret and control the data, the extension, of the level below The database is thought of as having multiple specification levels each of which serves as the intension of the next level's extension Convenient languages with recursion for accessing and manipulating objects at all levels of this hierarchical specification are needed

3. Database Design Methodologies

Unlike data processing environments which only capture a static snapshot of the world, EIS databases must capture the whole history of the information from the design of engineering products to maintenance The database is developed incrementally and is maintained for trace of information It must support cooperative design based on the fundamental practice of **reuse** of existing designs and previously obtained know-how To reuse a piece of design, a designer must be made aware of its existence and browse through its associated information

4. Information Management and Control

Long transactions, update propagation of dependent information and library like check-in/out control are distinct characteristics of engineering databases Techniques for dealing with long lasting processes are different than those dealing with rapid transactions found in today's data processing applications The required concurrency and consistency protocols must account for and control the so called engineering change Declarative and executable models are needed to capture **update dependencies** of objects Update dependencies can be thought as the knowledge and control needed to maintain the integrity and the consistency of an evolving database

Maintenance of versions must be based on **differential files**, logs of the changes made after a consistent state of the database and maintained separately from the main files Differential files save storage but, most importantly, make version control and regeneration of previous versions manageable

5. Interoperability of Environments

The large number of tools and special purpose processors, such as schematic analyzers, graphics editors, simulators, etc , that are part of the EIS environment will require distributed programming that is beyond the capabilities of an advanced central or distributed database system Remote Procedure Call (RPC) is a programming primitive that makes access and control of distributed systems easier by providing a simple primitive RPC is treated just like a local procedure call The advantage of RPC over a fully integrated distributed environment is that the former does not require integration but leaves the local environments independent and only integrates a very small subset of them, just what is absolutely necessary for the communication We call this **environment interoperability**

Atomicity and serializability are the properties that must be maintained Atomicity requires that either all the results of a procedure call are seen or none are seen If the system successfully completes an atomic operation, it moves the system from a consistent state to another consistent state If it fails, the system remains in the prior consistent state just as if no operation had been performed Serializability requires that the effects of interleaving several operations concurrently must be the same as if each is executing in some sequential order Atomicity and serializability in the EIS framework have to deal with much higher granularity than those in data processing environments

6. Architectures of EIS

The environment of an EIS is naturally distributed Therefore, all the concurrency and consistency control of distributed databases is present Furthermore, an EIS has additional distribution requirements that are dictated by the presence of tools interacting with it

In a typical EIS environment, most of the interactions will originate from a workstation or a tool tailored to some specific task It is highly unlikely that all the functionality of special purpose processors and tools that currently exist in the market or those that they will be developed in the near future can be provided by a single EIS Instead, special purpose processing will

continue to be done at tailored tools which will interact with an EIS This requires an EIS-tool architecture that is partially distributed between the two but tightly controlled by a special tool interface that explicitly models and manages their interactions

Data and processing is typically downloaded to the tool and processed independently To guarantee database consistency, updates that affect the EIS database must be transmitted to the EIS where they are validated for consistency Therefore, what is needed is not just an interface standard between EIS and tools, but a whole lot more EIS will keep track of how information from the EIS disseminates to the various tools, how updates done after the dissemination affect the processing at the tools, and what updates must be propagated to them Differential file techniques will have to be used to minimize data transmission

7. Conclusions

The purpose of this panel is to focus the attention of government, academia and industry to the importance of the concept of the EIS framework and nourish research and development There are long term and short term goals to be considered In the short term, the wealth of existing engineering design tools should be brought together in an EIS environment In the long term, the EIS must provide a strong framework for the continued evolution of tools and the engineering process

Current practice limitations, the lack of engineering tool integration with DBMSs, and strategies for achieving the above short and long term goals will be discussed