CIFE CENTER FOR INTEGRATED FACILITY ENGINEERING

CIFE Data Inventory: A Report on CIFE Data-Intensive Research Projects

by

D. H. Douglas Phan
Jamal A. Abdalla
H. Craig Howard

TECHNICAL REPORT
Number 57

October, 1991

Stanford University

If you would like to contact the authors please write to:

c/o CIFE, Civil Engineering,
Stanford University,
Terman Engineering Center
Mail Code: 4020
Stanford, CA 95305-4020

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS AND ACRONYMS

AEC	Architecture, Engineering, Construction
CAD	Computer-Aided Design
CIFE	Center for Integrated Facility Engineering
DBMS	Data Base Management System
PDES	Product Data Exchange using STEP
STEP	STandard for the Exchange of Product model data

GLOSSARY

ATTRIBUTE.

A formal property of the object class. The attribute definition in the object class specifies a static property true for all instances of that class, while the *attribute value* of a particular instance describes a dynamic property in the current state of that instance. The attribute value can be of a simple data type (e.g., integer, real, character, string) or a more complex data type.

DATUM.

A unit of information that describes a real life phenomenon or an abstract idea that people formulate and record [TSIC82].

DATA.

Specific instances of recorded information. Data include much detail, change rapidly over time, and are voluminous and stored permanently in database systems [WIED86].

DATABASE.

A formal collection of related data organized according to a priori defined schema [TSCI82]. Databases often serve the needs and cooperative uses of a large community of users. This function imposes a number of requirements in the development of the database: (1) an *external view (or subschema) level* that supports distinct classes of users or processes sharing the data, (2) a *logical schema level* that integrates these different view models, and (3) an *internal physical level* for the persistent data storage [BROD84].

DATABASE SCHEMA.

A logical set of data structures, their properties, constraints, and relationships that is well defined for a certain application using a particular data model [TSCI82]. It should be distinguished from a *data model*, which is the set of conceptual modelling tools used to define the database schema for a particular application.

DATABASE MANAGEMENT SYSTEM.

A system that provides facilities and tools for defining, manipulating, and controlling information in the database, including a protection mechanism for data and users [CARD90]. Aspects of the database control include semantic integrity maintenance, security in terms of user authorization, concurrency for multiple users, and recovery in the event of failure of some type.

DATA MODEL.

Collection of conceptual modelling tools for describing data, data relationships, data semantics and constraints among data items [TSCI82]. A data model consists of three major parts: *static properties* (data structures, constraints, and relationships), *integrity rules* (defined over both data and operations in order to maintain data accuracy), and *dynamic properties* (operations on data) [BROD84].

DATA STRUCTURE.

Basic building block of a data model used to organize data. A data structure can be a record in the network or hierarchical data models, a relation and its tuples in the relational data model, or an object in an object-oriented data model.

DATA TYPE.

A definition of data items in a particular domain, together with a well-defined structure, specific constraints, and operations defined on them. A data type can be of simple type (e.g., integer, real, character, string, etc.) or of more complex user-defined type.

ENTITY.

Representation of a distinguishable real-world object (whether it is concrete or abstract).

instance RELATIONSHIP.

Relationship that creates a particular occurrence of a generic entity with specific attribute values. The inverse relationship is the instance-of relationship.

is-a RELATIONSHIP.

Relationship to relate an entity to its parent entity. Object-oriented programming languages use this relationship to relate a subclass to its superclass. It enables the subclass to inherit all the properties and behavior from the superclass. The inverse relationship is the subclass relationship.

KNOWLEDGE.

Generalized statements that are used to process data and to support decision making. Knowledge is formalized from education and experience. Knowledge tends to be less time-variant and more complex, compared to data that correspond to specific instances of recorded information [WIED86].

OBJECT CLASS (or CLASS).

A set of similar objects that exhibit some common properties and behavior. The formal definition of a class specifies the descriptive features that are shared by the members (or

instances) of the class. Properties of a class are defined in terms of *attributes*, while its behavior is defined in terms of *methods*.

part-of RELATIONSHIP.

Relationship that relates an entity of a lower abstraction level to another entity of a higher abstraction level. This relationship is based on the abstraction method of aggregation. The inverse relationship is the subpart relationship.

RELATIONSHIP.

An association among entities in the data model; or formally a tuple of entities [e_1, e_2,..., e_n] from a mathematical relation R_i among n entities, each of which belongs to an entity set E_i [CHEN76].

REASONING.

Deliberation on representational structures in a system in order to answer question or to deduce new information in problem solving [LEVE85].

REPRESENTATION.

Selection of appropriate structures to represent data or knowledge and appropriate reasoning mechanisms to answer questions and to deduce new information, in accordance to the truth theory of some underlying representation language [LEVE85].

ABSTRACT

As CIFE researchers are rapidly developing intelligent systems to help integrating the facility engineering process, the potential for sharing and reusing data among these systems naturally instigates a need to study these data. This led to the CIFE data inventory task in the Fall of 1990. Its primary objective is two-fold: (1) to better understand the data modelling needs and requirements from the CIFE system developers' perspective and (2) to provide feedback to our ongoing data modelling effort.

Eight CIFE research projects were selected for this inventory task on the basis of their intensive data representation effort or of the large amounts of data used in the project. Data in these projects relate to the architecture, structural engineering, and construction aspects of facility engineering and show a strong potential to be shared and reused among the projects. The inventory of data in each project includes three steps. First, the *data collection* began with interviewing the primary investigator of the project and followed by collecting documentation of the project. Second, the *data analysis* involves studying the interview protocol and the collected documentation. Last, the *data summary* identifies the input data, output data, and potential integration points of the system with other systems studied in the inventory. This final step results in a written summary table of the data used in the project. Both the data analysis and the data summary follow a *three-tier framework* that reveals increasing levels of information about the project. The first level presents the general information about the research project such as research objectives, software and hardware implementation tools, etc. The second level describes the representation and reasoning aspects of the system developed in the project. The third level includes a detailed inventory of the data and knowledge used in that system.

As a result of this inventory task, data summary tables of the eight research projects are created using the three-tier framework mentioned above. In addition, other task results are also presented. First, common problem areas in representing facility engineering data in these projects are identified and discussed. Second, as a short-term solution to the data integration problem, key data integration points among the systems are defined and recommended for future CIFE research projects. These integration points are defined along the dimensions of the principal views of facility engineering data (i.e., architectural, structural and construction views) and along the key aspects of form, function, and behavior of structural engineering objects. Form, function, and behavior are emphasized here as the essential building blocks in representing facility engineering data. Third, the issue of software system documentation is brought to attention from observing the system documentation in these projects. For a more effective documentation of system and system data in future work, the same framework used in this inventory task to analyze and summarize data is recommended to system developers.

1. Introduction

Data are basic units of information that describe real life phenomena or abstract ideas that people formulate and record. Since data relate to specific instances of recorded information, they include much detail, change rapidly over time, and are voluminous in quantity [WIED86]. Data are stored in database systems, generated in computer applications, or used in problem-solving tasks of knowledge base systems. As the principal substance for information processing and problem solving, data play a significant role in all these computer systems. More importantly, suitable data representation and organization in order to serve the needs and uses of data in these systems are also of upmost importance.

Compared to data in traditional business applications (e.g., payroll, accounting, inventory control), data in facility engineering are more complicated and thus are more difficult to model. In addition, facility engineering data are used by and shared among several project participants, design phases, and computer applications. In a cooperative engineering environment of this nature, data must be closely "integrated" to support different data needs and uses. In a broader sense, *integration in facility engineering* means coordination of the planning, design, and construction phases of a single project through the cooperative use of information (including data) from computer database and knowledge base systems [TATU90]. Since the design and construction of a facility (e.g., building, bridge, transmission pole, space station) involve a complex engineering process, advanced computer tools and means of computer-based automation have been studied in order to achieve higher degree of integration [HOWA89a, COLL90]. At the Center for Integrated Facility Engineering (CIFE), many research projects are exploring new frontiers of facility engineering integration.

As CIFE researchers are rapidly developing intelligent systems to help integrating the facility engineering process, the potential for sharing and reusing data among these systems within CIFE naturally instigates a need to study these data. This led to the CIFE data inventory task that took place in the Fall of 1990.

This report describes the CIFE data inventory task. Section 2 presents the background information, the objective of the task, the research projects included in the inventory, and the framework within which the task was carried out. Section 3 summarizes the task results and discusses a number of lessons learned from this data inventory experience. Section 4 concludes the report by presenting some potential solutions to the data integration problem. Section 5 acknowledges the support that we received from CIFE researchers in carrying out this task. Section 6 lists the reference materials used in this report.

2. CIFE Data Inventory Task

2.1 Background

The CIFE data inventory task was part of the data modelling effort that supports the CIFE Flagship project on "Linking Design Data with Knowledge-Based Construction System" [HOWA89b]. The goal of this project is to develop and to demonstrate a prototype environment that supports flexible and dynamic linkage of facility design data with knowledge-based construction systems. The CIFE data modelling effort supports one of the key objectives of the Flagship project: "define a unified view of data spanning both design and construction" [HOWA89b]. It is an integral component of the Flagship project.

The early data modelling effort included the development of an initial model for the domain of structural steel framing, namely the Structural Steel Framing Data Model (SSFDM) [LAVA89]. This model was later evaluated using a real life structure as the modelling prototype [PHAN90]. These initial activities provided an insight into the area of data modelling, a better understanding of the PDES/STEP data exchange standard, and valuable experience in developing and testing a data model. The long-term goal aims at developing a data model for structural engineering that supports data integration in facility engineering and at applying this data model to different structure types.

2.2 Objective

The objective of the CIFE data inventory task is two-fold:

- ***To better understand the data modelling needs and requirements from the CIFE system developers' perspective.*** First, this task enables us to gain a more in-depth understanding of different CIFE research projects in terms of their data representation and use. By "use," we mean the manipulation and reasoning of data in the application to perform problem-solving tasks. As a result, we can identify the common data among different CIFE applications and also can better understand the overall needs and requirements for data modelling from the developers' viewpoint.

- ***To provide feedback to our ongoing data modelling effort in terms of the data use in different CIFE research projects.*** Second, the user perspective gained from this experience will help us to develop an integrated data model that can support the consortium of CIFE applications. This objective fits into our long-term goal to define a unified view of design and construction data for the Flagship project .

Indirectly, we viewed this task as an opportunity to study the issues as well as the potential of data integration among the different CIFE research projects.

2.3 Task Description

2.3.1 Selection of CIFE Research Projects

SELECTION CRITERIA This task involves the data inventory of a number of selected CIFE research projects that suit the above task objective and that conform to the following criteria:

- research projects in the three CIFE thrust areas (i.e., Artificial Intelligence/Knowledge Base Systems, Data Base, and Field Automation/Robotics), with strong relevance to the design and construction aspects of facility engineering

- projects that are further advanced in their development time table

- projects that involve a large quantity of data† or an intensive data representation effort in a framework that we can follow and comprehend.

† Although we use the term "data" more often in this report, some of the selected research projects involve the development of a knowledge base system where it is more appropriate to mention both knowledge and data.

SELECTED RESEARCH PROJECTS The research projects selected for this data inventory task include:

1. *Knowledge-Based Layout Generation System*, Paul Chinowsky with Professor P. Teicholz
2. *Logic-based Integrated Structural Design of Steel Office Buildings*, Deepak Jain with Professor H. Krawinkler
3. *Constructibility Improvement for the Preliminary Design of Reinforced Concrete Buildings*, Martin Fisher with Professor C. B. Tatum
4. *Development of CIFECAD System (Version 2.0)*, M. N. Kolountzakis and Martin Fischer
5. An Object Model for Integrated Project Management, Thomas Froese with Professor B. Paulson
6. *Data Base Schema for Building Design*, Lixin Chen with Professor K. H. Law (Part of the PENGUIN project [LAW89])
7. *Applying Design-Dependent Knowledge in Structural Engineering Design*, Jenmu Wang with Professor H. C. Howard
8. *OARPLAN, an Object-Action-Resource PLANning system*, Adnan Darwich and Richard McGrath with Professor R. Levitt.

2.3.2 Steps in Data Inventory

The inventory of data in each of the above projects consists of the following three steps:

1. ***Data Collection:*** We first conducted an interview with the primary investigator of the project using the questionnaire form shown in Appendix A. Before the interview session, the interviewee had a chance to review the questionnaire. After the interview, we collected relevant documentation about the research project, about the system[¥] in development, and about the detailed data description for later examination.

2. ***Data Analysis:*** In this step, we examined the interview protocol as well as the documentation collected from the previous step. We carefully studied the data representation and data use in the project using a methodical framework that will be described in the next section. For every project, we then conducted a follow-up interview in order to fully comprehend the details and fine points of the research. In some projects, we managed to see a demonstration of the system given by the system developer.

3. ***Data Summary:*** For every system, we identified the input data, output data, and potential integration points of the system with other systems (studied in the inventory) and prepare a

[¥] We use the word "system" here in a general sense; however, some of the projects involve the development of a model or a data base schema.

written "data summary" of the research project. After the data summary was done, we also spent much time and effort with the researcher to confirm the accuracy of its content.

2.3.3 Framework for Data Analysis and Summary

The data summary from Step 3 above follows a three-tier framework that is intended to provide the reader with increasing levels of information about the research project. The same framework was used to analyze data in Step 2. The three levels of the framework are:

1. ***About the Research:*** This level captures the general information about the research project: project title, name of primary investigators (students, visiting fellows, sponsoring faculty members), research objectives, salient aspects of the research (or contributions), reference literature, name of the system in development, and software and hardware implementation tools.

2. ***About the System/Model:*** This level includes more specific information about the system (or model) developed in the research project. In the data analysis step (Step 2 above), we studied both the representational and reasoning aspects of the system. In particular, we distinguished the *knowledge level* from the *symbol level* of the system. On one hand, the knowledge level focuses on the identification of the types of knowledge included in the system to solve the problem at hand (namely what is said about the real world in the system), rather than the symbolic structures used to represent those knowledge types. On the other hand, the symbol level deals specifically with the symbol structures and the representation of the embedded concepts as well as the reasoning and search methods used in the system [STEF90].

3. ***About the Data/Knowledge:*** This level is much more detailed than the previous two levels because it includes a detailed inventory of the data and knowledge used in the system. The analysis done at this level requires us to examine the detailed project documentation (even the program code in some cases) and to closely interact with the system developer.

Appendix B shows the tabular forms that reflect the above analytical levels. These tables are used both in the data analysis and data summary steps.

3. Task Results

Appendix C contains the data summaries of the projects studied in our inventory. This appendix is organized according to the three-tier framework discussed in the previous section. In addition, there are a number of lessons learned that provide valuable feedback to our ongoing data modelling research. These lessons relate to the following areas:

1. ***Facility Engineering Data Representation:*** Compared to data in traditional business applications (e.g., payroll, accounting, inventory control), structural engineering data are more complicated and thus are more difficult to model. First, structural engineering design artifacts are complex and involve many levels of details. Second, the associated data have highly nested data structures and intricate relationships among themselves. Third, like other engineering data, structural engineering data involves more types than instances, thus requiring larger and more involved database schemata. Last, they describe physical, functional, and behavioral properties of the design artifact. In fact, they can be classified into three categories: *form*, *function*, and *behavior* [LUTH90]. Form data describe the physical properties of the design object such as coordinates, square footage area, member span length, shape, material, etc. Function data

relates to the description of the intended role or purpose of the design object in its constructed environment. They include a variety of data that describe architectural and structural engineering functions and design requirements such as daylighting requirements, privacy level, acoustic level, structural load resisting, load transferring, member supporting, etc. Behavior data manifests how the function of the object is carried out under the influence of the environment. In structural engineering, the behavior of a structural component under the influence of loading is measured in terms of internal forces and stresses, deflection, deformation, vibration, etc. Form, function, and behavior should be distinct, yet cohesive, elements in representing facility engineering data.

2. ***Data Integration within CIFE:*** In facility engineering, a computer application that supports a particular design phase needs the data generated from applications of the preceding design phases. For example, the structural floor generation application uses the data (describing the location and dimensions of the architectural spaces) from an architectural space layout application. The needs for data sharing and reuse among applications in a cooperative design environment such as facility engineering require that data be readily exchanged among these systems. However, the difficulty in integrating these applications is that they might differ in terms of programming paradigm, representation language, and system model. This is referred to as the "impedance mismatch" problem. Moreover, data used in these applications might be different in terms of their syntax and semantics. Data integration provides a bottom-up approach to handle the integration problem from the grass-roots level of data and data models. It ensures compatibility of data and database schemata through standards and facilitates data exchange among computer applications. As a result, it reduces the degrees of impedance mismatch among applications and increases the potential for sharing and using data among them. In short, data integration should be an important and integral part of the overall facility engineering integration effort.

   In particular, although the actual data representation may differ from one application to the others, there is a strong potential for sharing and reusing data and database schemata among applications developed at CIFE. The short-term need is to identify key data integration points along which applications can share data. The long-term goal is to develop a data model that directly supports the data integration goal and that provides a unified representation base for both design and construction data. The requirements for a data model of this nature are that (1) it should be flexible to allow users to customize their own views about the data, (2) it should be extensible to allow adding new data types to the database schema, (3) it should facilitate the data exchange among different applications, and (4) it should incorporate the aforementioned elements of form, function, and behavior in the data representation.

3. ***Data and System Documentation:*** As the number of CIFE software systems proliferates, the amount of information about these systems quickly accumulates. The success of data integration depends not only on how well the data were represented or how well the system was developed, but also on how well the data and the system were documented for later reuse. Although it may seem trivial, this is an important pragmatic aspect of data reusability and therefore of data integration.

In the subsequent sections, we discuss the above areas in greater detail.

3.1 Problem Areas in Facility Engineering Data Representation

In this section, we will present the problem areas in facility engineering data representation that were observed from this inventory task. Generally speaking, system developers are dedicated to developing the end product for certain applications. The preoccupation with developing a working

system and with reaching the end goal might unfavorably compromise the benefits of good system representation or model design. While there might many possible instances of poor representation, the following problem areas are frequently encountered in the data inventory:

1. ***Semantic Overloading of Object Attributes:*** This refers to the situation where the same attribute represents different things in various object classes. This might occur in two cases: (1) when an attribute is inherited in different subclasses from a common superclass, and (2) when the same attribute is used in several disjoint object classes. For example, an attribute SIZE is used in four different object classes: "Beams," "Columns," "Slabs," and "Doors." Clearly, it can take upon different meanings in these object classes: beam depth in "Beams," diameter in (round) "Columns," wall thickness in "Walls," and door width in "Doors." This plethora of meanings from the same attribute in different object classes is a potential source of confusion for both the code reader and the developer.

2. ***Misuse of Is-A Relationship:*** Tools to build generalization class hierarchies are common to commercial object-oriented software (i.e., programming languages, expert system shells, database management systems). The ubiquitous is-a relationship relates a subclass to its superclass and is used to construct these class hierarchies. The convenience of using this abstraction tool leads to a common trap in which the semantic accuracy of the representation is completely ill-suited. As illustrated in Figure 1, a single class hierarchy consists of the "Buildings" class as the root node and a subclass "Structural_Members," which in turn has subclasses "Beams," "Columns," "Slabs," "Doors," and "Windows." This class hierarchy is ill-defined and semantically inaccurate (because the "Structural_Members" class should be related to the "Buildings" class by the part-of relationship). As strongly suggested by research studies on database schema integration [DAYA84, NAVA86], there is a need to define the semantics and proper use of the is-a relationship. Zdonik and Maier, for instance, suggested three different kinds of is-a class hierarchies for cleaner semantics: *classification hierarchy* for collecting and categorizing object classes, *specification hierarchy* that honors substitutability of subclasses in their superclasses, and *implementation hierarchy* for code sharing with less emphasis on the semantic validity [ZDON90].

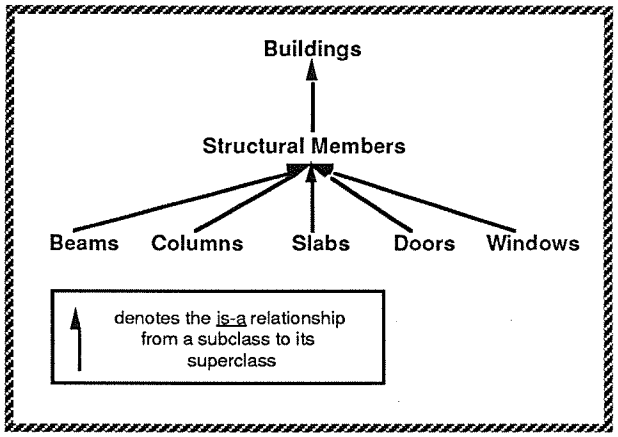

FIGURE 1: Improper is-a Class Hierarchy

3. ***Non-homogeneous Class Hierarchy:*** This phenomenon occurs when many different criteria are used to define subclasses at different levels of the class hierarchy, as shown in Figure 2. This is an undesirable feature for two reasons. First, since different views and semantics are mixed in an indiscriminate manner in the construction of the class hierarchy, this leads to a poor representation with minimal separation between the different aspects (i.e., form, function, and behavior) of the object class description. Second, the use of such non-homogeneous hierarchies requires the modeler to anticipate all possible combinations of subclasses. This can lead to what is called the "cross product phenomenon": when the number of criteria used in a single hierarchy increases, the resulting model grows exponentially larger, and the modeler must supply the expertise to eliminate subclasses that represent invalid combinations (and possibly some valid combinations as well) [HOWA91].

4. ***Large Object Clusters:*** The convenience of hustling many attributes into one object class presents a common trap. The end results are "large object clusters." By "large", we mean that multiple aspects of description are combined into one object class definition, resulting in a large number of attributes and relationships and a highly nested data structure. For instance, consider an object class "Columns" as shown in Figure 3. These large object clusters create a number of problems:

- First, it becomes increasingly awkward to create or modify instances of these large object classes, and to manage their versions. From the designer's point of view, instantiating such an object class demands attribute values that are not all defined at the same time. Update and version management of such instantiated objects are difficult since each instance involves a large cluster of attributes that are not all modified.

- Second, in terms of problem solving, as Minsky put it:

"... in such a complex problem, one can never cope with many details at once. At each moment one must work within a reasonably simple framework... any problem that a person can solve at all is worked out at each moment in a small context and that the key operations in problem-solving are concerned with finding or constructing these working environments." [MINS75]

- Third, a schema with only a few large objects is too rigid and cannot accommodate future evolution of the schema in the design process. It is also difficult to reuse such a schema in other applications.

- Fourth, exchanging large objects among applications requires parsing out the needed data, which can be highly inefficient and computationally costly.

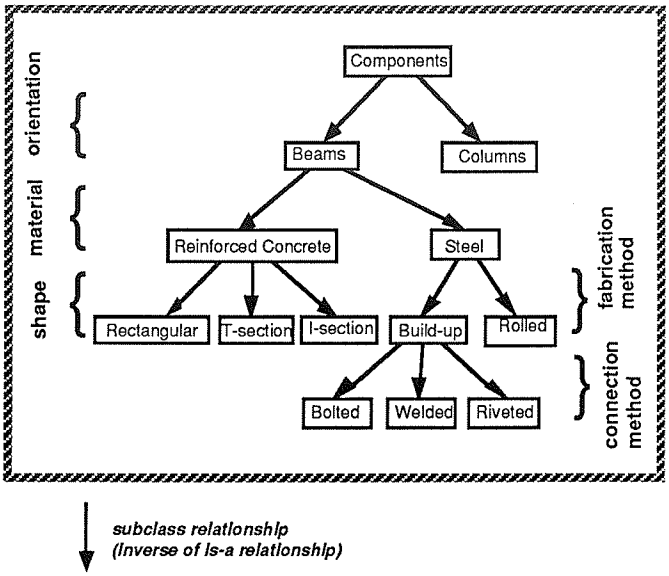

FIGURE 2: ***A Sample Non Homogeneous Class Hierarchy.***

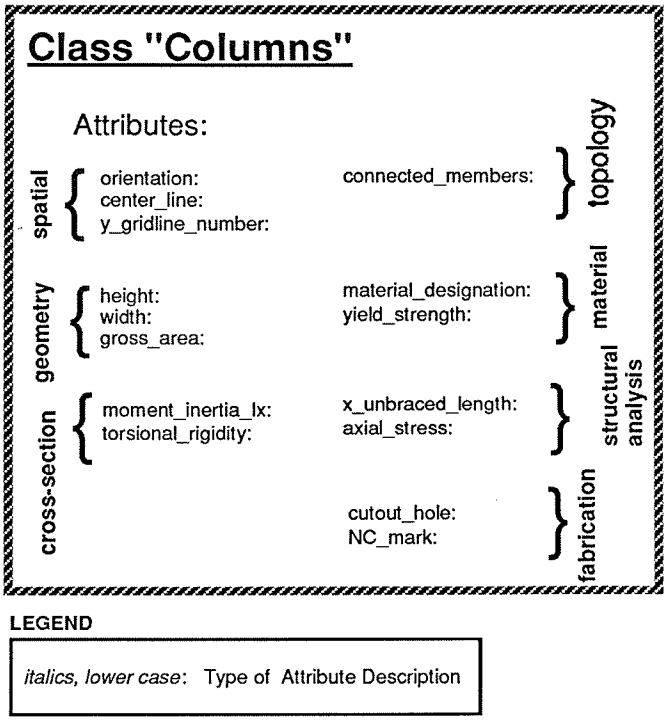

FIGURE 3: ***A Sample Large Object Cluster.***

- Last, the semantic composition of such large objects becomes incoherent and, as a consequence, poorly defined. In some cases, it remains unclear whether certain attributes truly belong to the object, or instead describe some properties that are beyond the abstract boundaries of the intended object definition.

5. ***The part-of Fallacy:*** Aggregation is a common abstraction method for constructing complex design objects from their components. The inverse *decomposition* method breaks down complex design objects into their component objects. The part-of relationship and its inverse subpart relationship are used for these abstraction methods respectively. Aggregation is commonly used in engineering data modelling since engineering data tends to be hierarchical in nature [KETA86]. First, although the importance of aggregation/decomposition cannot be overlooked, their misuse can be counterproductive. For example, part-of relates two objects "structural-member-23" and "structural-element-23" whose relationship is not aggregational nor hierarchical; in actuality, these two objects represent different aspects (i.e., functional and analytical) of the same object "beam23." Instead, their relationship should be an association. Second, the semantics of aggregational relationships such as part-of were not fully addressed. [KUCZ90] pointed out the semantic subtleties of the part-of relationship in statements like "the piston is part of the engine," "the ship is part of the fleet," "steel is part of the car," "paying is part of shopping", etc.

The above key points underline the importance of good representation in developing a computer system (e.g., computer program, design application, knowledge base system, or a database schema) for facility engineering. Although the representational aspect of the system is traditionally motivated by its reasoning mechanisms—this is well understood and even strongly promoted in Model-Based Reasoning [KUNZ88]—representation should also be done in such a way that the chosen representational structures can be reused in other systems, and their data content can be easily exchanged among different applications. The bottom line is that shareability and reusability of the system model can enhance the prospect of integrating that system with other systems in the computing environment.

3.2 Issue of Facility Engineering Data Integration

Although there were dissimilarities in data representation among the research projects studied in this inventory, we also observed a strong potential for integrating these projects at the data level. Due to the large amounts and heterogeneity of data in these projects, it is difficult within the limits of this report to define integration points at the detailed level of object attributes. Instead, a more sensible approach here is to identify "macro" (or high level) integration points along the dimensions of the principal views of facility engineering data (i.e., architectural, structural and construction) and along the key aspects of form, function, and behavior of structural engineering design objects. This approach is consistent with the way facility engineers reason about their design artifacts and with the framework within which computer applications can be developed to simulate this type of design behavior.

ARCHITECTURAL VIEW In the architectural view, we define the following macro integration points:

- *Architectural Space Form:* Physical data that describe the geometric properties of a three-dimensional architectural space in a floor layout plan (e.g., square footage area, dimensions, dimension ratios).

- *Architectural Space Layout:* Data about the location, orientation, and arrangement of architectural spaces in a floor layout plan. This also includes the interactive relationships among architectural spaces in the layout as well as any hierarchical organization of those spaces.

- *Architectural Space Utilization Function:* Designated utilization function of an architectural space such as core, lobby, corridor, elevator-bank, staircase, and office.

- *Architectural Space Design Attributes:* Design attributes of architectural spaces such as daylighting, privacy, security, acoustic, view requirements, which are used as requirements or guidelines in the floor layout process. These attributes are also part of the functional description of the architectural space.

STRUCTURAL VIEW In the structural engineering view, we distinguish form, function, and behavior. In addition, we further subdivide "form" into different types of form that are commonly used in the systems we examined in the inventory. These types of form include:

- *Spatial Form:* This form describes where and how a physical object is located, oriented, and realized in three-dimensional space. Such a description includes the spatial envelope of the object and its location and orientation with respect to certain global reference datum or relative to other objects in its environment. The spatial envelope of a physical object can be defined in terms of a local coordinate system and the dimensions (length, width, and height) of its spatial enclosure.

- *Geometric Form:* This form defines the geometric shape (and dimensioning) of the object in terms of geometric elements such as points, lines, curves, surfaces, etc. Physical objects are three-dimensional, but their shapes can be represented in many ways by different geometric forms.

- *Cross-sectional Properties Form:* This form includes the cross-sectional properties of the object such as gross cross-sectional area, effective load resting area, moments of inertia Ix and Iy, torsional rigidity, etc. These properties are derived from the above geometric forms of the object.

- *Topological Form:* This form defines the connectivity of the object in the constructed environment in terms of topological elements such as vertices, edges, faces, volumes, etc. In structural engineering, a wire frame model of the structure is commonly constructed in order to define the topology of the structure. Such a wire frame model is analogous to a finite element model used for structural analysis purpose.

- *Material Form:* This form describes the material type and material properties of the object.

- *Hierarchical Aggregation Form:* This form is used to describe the hierarchical framework of structural design systems and their component objects. Structural members such as beam, column, girder, etc. are not designed independently of one another, but are designed from some preconceived load resisting systems to which they belong. Common entities of this form include "Systems," "Assemblies," "Arrangements," "Members," "Parts," and "Segments."

- *(Part) Fabrication Form:* This form includes fabrication features of an engineering part prescribed by the designer. There is a large set of standard fabrication features such as taper, bend, thread, cut out hole, edge clipping, edge preparation, NC mark, etc.

In addition, we also consider *structural loading* as another integration point. Structural loading data are considered and used in both the analysis and the design of the structure. Two important aspects of structural loading are the loads and the loading conditions. While the former directly determine the functions and design requirements of the structural members, the latter influence the designer's criteria concerning the external loading effects for which the entire structure was analyzed and designed.

CONSTRUCTION VIEW In the construction view, we identified the following two broad integration points:

- *Activity-Related Objects:* These objects describe activities in the construction project. They include attributes describing the activity, the resources utilized, and project scheduling parameters (e.g., activity duration, start and end dates, and precedences) for the execution and completion of the activity.

- *Cost-Related Objects:* These objects are used to estimate and to control the cost of the construction project in terms of material, labor, equipments, etc.

Table I shows the key integration points among the research projects studied in this inventory.

TABLE I: IDENTIFIED MACRO INTEGRATION POINTS AMONG RESEARCH PROJECTS IN THE DATA INVENTORY. These integration points are defined along the dimensions of the principal views of facility engineering data (i.e., architectural, structural and construction) and along the key aspects of form, function, and behavior of structural engineering design objects. The systems developed in these research projects are:

(1) CAADIE — Knowledge-Based Layout Generation System -
(2) FFG — Logic-based Integrated Structural Design of Steel Office Buildings
(3) COKE — Constructibility Improvement for the Preliminary Design of Reinforced Concrete Buildings
(4) CIFECAD (Version 2.0) — Development of CIFECAD System -
(5) An Object Model for Integrated Project Management
(6) Data Base Schema for Building Design
(7) DDIS — Applying Design-Dependent Knowledge in Structural Engineering Design
(8) OARPLAN — an Object-Action-Resource PLANning system.

Note: The functions of structural engineering design objects in these systems, except for FFG, were designated implicitly by the class type and class name of the object such as beams, columns, slabs, etc.

Notations: +: Strong integration points (A lot of data are shared, object definitions can be directly reused.)

√: Weaker integration points (Some data are reused.)

RESEARCH PROJECTS / INTEGRATION POINTS	CAADIE	FFG	COKE	CIFECAD	OBJECT MODEL	BUILDING SCHEMA	DDIS	OARPLAN
ARCHITECTURAL VIEW								
Architectural Space Form	+	+						√
Architectural Space Layout	+	√						√
Architectural Space Utilization Function	+	√						√
Architectural Space Design Attributes	+							
STRUCTURAL VIEW								
Spatial Form		+	+	+		+	+	√
Geometric Form		+	+	+	√	+	+	√
Topological Form		+	+	+	√	+	+	+
Cross-Sectional Properties		+				+	+	
Material Form		+	+	√	√	+	+	√
Hierarchical Aggregation Form		+	√	√		+		+
(Part) Fabrication Form								
Functions		+	√	√	√	√	√	√
Behaviors		+					+	
Structural Loading		+					+	
CONSTRUCTION VIEW								
Activity-Related Objects					+			+
Cost-Related Objects		√			+			

Figure 4 shows how the four different CIFE applications (i.e., CADDIE, FFG, COKE, and OARPLAN) that automate various tasks in architectural design, structural engineering, and construction project planning can share data along the line of the above key integration points.

3.3 Issue of Software System Documentation

In this data inventory, we found that the software system documentation is proprietary to the developers of the system and can be dispersed in different sources. Moreover, the complete software system documentation generally occurs toward the end of the project. We strongly believe that the availability of adequate and up-to-date system documentation during the project development would greatly benefit the project participants and other outside users, particularly when reviewing certain features (e.g., the system representation of certain concept or the system implementation of a particular reasoning task) or software modules of the system in order to reuse them in other projects. We also feel that the software system documentation should be done early in the project and preferably during and throughout the design and implementation of the system.

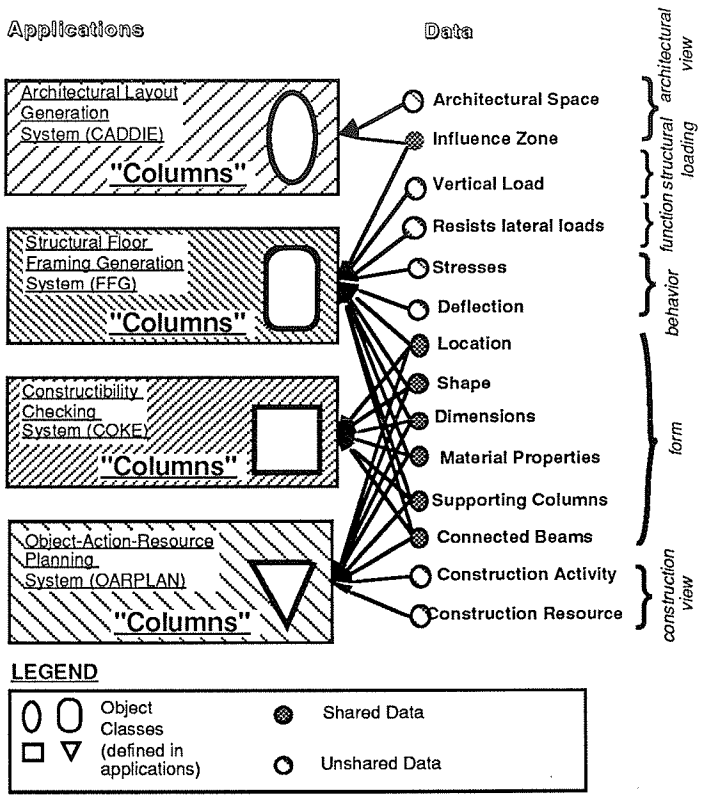

FIGURE 4: ***Example of Data Sharing Among Applications.***

The complete system documentation should describe in detail the user interface, all required input data, the architecture and functionalities of all software modules, all representational structures, all reasoning tasks and their implementation, all output data, and any testing and validation of the system that has been performed. In addition, data used in the system should also be documented for the purpose of data exchange and reuse.

The complete documentation of individual data elements should include the data name, the data unit, the data type (e.g., character, integer, string, real, abstract, pointer, etc.), the domain (i.e., the set of permissible data values as in the case of enumerated data type), the data source (i.e., the abstract data type or object class where the data element is defined), the data characteristics (i.e., input, optional, derived, output), the data use (i.e., reasoning tasks in which the data element is used), and any data integration points (i.e., any other systems or domain tasks that potentially need and reuse the data elements) that have been identified.

For future reference, the system developer may use the framework for data analysis/summary as shown in the appendices to document the system and system data.

4. Conclusions

In this report, we described the data inventory of a selected number of data-intensive research projects within CIFE. These projects involve large amounts of data that relate to the various aspects of facility engineering (i.e., architecture, structural engineering, and construction) and that also show a strong potential to be shared and reused among the projects. We pointed out the common problem areas encountered in facility engineering data representation. We also emphasized that form, function, and behavior should be the important building blocks in representing facility engineering data. As a short-term solution to the data integration problem, we identified some key data integration points among the systems studied in the data inventory and suggest that these integration points will be considered and followed in future research projects.

As a long-term solution to the data modelling and data integration problems, we identified a need for an integrated data model that can provide the conceptual modelling tools to represent facility engineering data, that can support multiple user views and design schema evolution, and that can facilitate data exchange among applications that make use of the data model. We are currently working in this direction, with the development of the Primitive-Composite data model [ABDA91]. This data modelling work will offer many benefits to future research development at CIFE. First, the Primitive-Composite data model will be applied to different types of structures in order to develop their database schemata of object classes. These object classes can be reused in future CIFE research projects and therefore can expedite their data representation effort. Second, the database schemata in turn will be used to create integrated databases of prototype structures that will provide real life sample data to different CIFE applications. The use of a unified database schemata and a common data sample will greatly enhance the potential to integrate these applications. Third, this development effort will establish a formal framework for standardizing facility engineering data. Finally, it will also provide a data integration approach that can be dynamically implemented using KADBASE [HOWA86, HOWA89b] to support real-time data queries between heterogeneous application systems.

5. Acknowledgments

The experience we gained from this data inventory task is valuable to our ongoing data modelling research. We were motivated by the spirited level of cooperation and support from our CIFE research colleagues. Without it, it would be impossible to carry out and complete this task, especially given the time constraints. We express our appreciation to all of them, for their time, effort and patience in getting us well acquainted and informed about the fine points of their research. In particular, we extend our special thanks to Ajay Lavakare and Dr. Martin Fischer for their input and review of this report.

6. References

[ABDA91] Abdalla, G. A., Phan, D. H., and Howard, C., *Formalization of the Primitive-Composite Structural Data Model,* Upcoming CIFE Technical Report, Center for Integrated Facility Engineering, Stanford University, Stanford, CA, 1991.

[BROD84] Brodie, M. L., "On the Development of Data Models," *On Conceptual Modelling: Perspectives from Artificial Intelligence, Databases, and Programming Languages,* pp. 19-47, edited by M. L. Brodie, J. Mylopoulos and J. W. Schmidt, Springer-Verlag, NY, 1984.

[CARD90] Cardenas, A. F. and McLeod, D., "An Overview of Object-Oriented and Semantic Database Systems," *Research Foundations in Object-Oriented and Semantic Database Systems*, pp. xvii-xxv, Prentice Hall, 1990.

[CHEN76] Chen, P. P. S., "The Entity-Relationship Model - Toward a Unified View of Data," *ACM Transactions on Database Systems*, Vol. 1, No. 1, pp. 9-36, March 1976.

[CHEN90] Chen, Lixin, *Database Scheme for Building Design,* Unpublished Project Update Report, Department of Civil Engineering, Stanford University, Stanford, CA, November 1990.

[CHIN90a] Chinowsky, P. and Teicholz, P., *A Knowledge-Based Layout Generation System,* CIFE Working Paper No. 7, Center for Integrated Facility Engineering, Stanford University, Stanford, CA, February 1990.

[CHIN90b] Chinowsky, P., *A Model for the Representation of Design Knowledge,* Unpublished Paper, Center for Integrated Facility Engineering, Stanford University, Stanford, CA, November 1990.

[CIFE90] Center for Integrated Facility Engineering (CIFE), *CIFE Symposium Proceedings*, CIFE Technical Report No. 24, Stanford University, Stanford, CA, March 1990.

[COLL90] Collins, R. J., Levitt, R. E. and Tatum, C. B., "Organizational and Institutional Impacts of Computer Technologies in Facility Engineering: a Background Review," Center for Integrated Facility Engineering (CIFE), Stanford University, Stanford, CA, 1990.

[DARW89] Darwiche, A., Levitt, R. E. and Roth, B. H., *OARPLAN: "Generating Project Plans in a Blackboard System by Reasoning about Objects, Actions and Resources,"* CIFE Technical Report No. 2, Center for Integrated Facility Engineering, Stanford University, Stanford, CA, February 1990.

[DATE83] DATE, C. J., *An Introduction to Database Systems,* Vol. II, The System Programming Series, Addison-Wesley Publishing Co., Massachusetts,1983.

[DAYA84] Dayal, U., and Hwang, H. Y., "View Definition and Generalization for Database Integration in the Multidatabase System," IEEE Transactions on Software Engineering SE-10, Vol. 6, pp. 628-44, 1984.

[FISC89] Fischer, M. and Tatum, C. B., *Partially Automating The Design-Construction Interface: Constructibility Design Rules for Reinforced Concrete Structures,* CIFE Working Paper No. 4, Center for Integrated Facility Engineering, Stanford University, Stanford, CA, June 1989.

[FISC90] Fischer, M., *Constructibility Improvement for the Preliminary Design of Reinforced Concrete Buildings,* Unpublished Ph. D. Thesis, Construction Engineering & Management Program, Department of Civil Engineering, Stanford University, Stanford, CA, June 1989.

[FROE90a] Froese, T. M., *An Object Model for Integrated Project Management Software,* Ph.D. Research Proposal, Construction Engineering & Management Program, Department of Civil Engineering, Stanford University, Stanford, CA, April 1990.

[FROE90b] Froese, T. M., "Object-oriented Programming for Project Management Software", *7th International Symposium on Automation and Robotics in Construction*, pp. 513-521, Bristol, England, June 1990.

[FROE90c] Froese, T. M., *Object Model for Project Management - Conceptual Design,* Unpublished Paper, Construction Engineering & Management Program, Department of Civil Engineering, Stanford University, Stanford, CA, November 1990.

[GARR89] Garrett, James H., "An Object-Oriented Representation of Design Standards", *Computing in Civil Engineering*, edited by T. O. Barnwell, Sixth Conference on Computing in Civil Engineering, Atlanta, Georgia, 1989, American Society of Civil Engineers, pp. 267-274, 1989.

[HAYE84] Hayes-Roth, B., *BB1: An architecture for blackboard systems that control, explain, and learn about their own behavior*, Heuristic Programming Project Report HP-84-16, Stanford University, Stanford, CA, December 1984.

[HOWA86] Howard, H. C., and Rehak, D. R., "Interfacing Databases and Knowledge Based Systems for Structural Engineering Applications, " *Technical Report EDRC-12-06-86*, Engineering Design Research Center, Carnegie-Mellon University, Pittsburgh, PA, November 1986.

[HOWA89a] Howard, H. C., Levitt, R. E., Paulson, B. C. Jr., Pohl, J. G. and Tatum, C. B., "Computer Integration: Reducing Fragmentation in the AEC Industry," *Journal of Computing in Civil Engineering,* Vol. 3, No. 1, pp. 18-32, 1989.

[HOWA89b] Howard, H. C. and Levitt, R. E., "Linking Design Data with Knowledge-Based Construction Systems," *A Flaship Project Proposal to CIFE*, Center for Integrated Facility Engineering, Stanford University, Stanford, CA, Stanford, CA, October 1989.

[HOWA91] Howard, H. C., and Abdalla, J. A., and Phan, D. H., "A Primitive-Composite Approach for Structural Data Modelling," To be published in the *Journal of Computing in Civil Engineering*, Department of Civil Engineering, Stanford University, Stanford, CA 1990.

[JAIN89] Jain, D., Krawinkler, H., and Law, K. H., *Knowledge Representation with Logic,* CIFE Technical Report No. 013, Center for Integrated Facility Engineering, Stanford University, Stanford, CA, July 1989.

[JAIN90] Jain, D., Luth, G. P., Krawinkler, H. and Law, K., *A Formal Approach to Automating Conceptual Structural Design,* CIFE Technical Report No. 31, Center for Integrated Facility Engineering, Stanford University, Stanford, CA, August 1990.

[KETA86] Ketabchi,M. and Berzins,V., "Component Aggregation: A Mechanism for Organizing Efficient Engineering Databases," *IEEE*, pp. 104-112, Los Angeles, February 1986.

[KOLO90] Kolountzakis, M. N. and Fischer, M. A., *CIFECAD: Description and User's Manual*, Center for Integrated Facility Engineering, Stanford University, Stanford, CA, October 1990.

[KUCZ90] Kuczora, P. W., "Representing Parts Hierarchies for Expert Systems in Engineering Project Management," *Computing & Control Division of the Institution of Electrical Engineers*, Conference Publication No. 322, First International Conference on Expert Systems, June 1990.

[KUNZ88] Kunz, J. C., "Model Based Reasoning in CIM," *Intelligent Manufacturing: Expert Systems and the Leading Edge in Production Planning and Control*, Addison Wesley, 1988.

[LAVA89] Lavakare, A. and Howard, H. C., *Structural Steel Framing Data Model,* Technical Report No. 012, Center for Integrated Facility Engineering (CIFE), Stanford University, Stanford, CA, June 1989.

[LAW89] Law, K. H., Barsalow, T. and Wiederhold, G., *Management of Complex Structural Engineering Objects in a Relational Framework*, Technical Report No. 019, Center for Integrated Facility Engineering (CIFE), Stanford University, Stanford, CA, September 1989.

[LAW86] Law, K. H. and Jouaneh, M. K., "Data Modelling for Building Design," *Computing in Civil Engineering, Proceedings of the Fourth Conference*, October 1986.

[LEVE85] Levesque, H. J. and Brachman, R. J., "A Fundamental Tradeoff in Knowledge Representation and Reasoning (Revised version)," *Readings in Knowledge Representation,* pp. 41-70, edited by R. J. Brachman and H. J. Levesque, Morgan Kaufmann Publishers, Inc., CA, 1985.

[LUTH90] G.P. Luth, Representation and Reasoning for Integrated Structural Design of High-Rise Commercial Office Buildings. Ph.D. Thesis (under preparation), Department of Civil Engineering, Stanford University, Stanford, CA, 1990.

[MARS87] Marshall, G., Barber,T. J. and Boardman, J. T., "Methodology for Modelling a Project Management Control Environment," *IEE Proceedings*, Vol. 134, No. 4, pp. 287-300, July 1987.

[MINS75] Minsky, M. L., "A Framework for Representing Knowledge," *The Psychology of Computer Vision,* pp. 211-277, edited by P. Winston, McGraw-Hill, NY, 1975.

[NAVA86] Navathe, S., Elmasri, R., and Larson, J., "Integrating User Views in Database Design," IEEE Computer, Vol. 19, No. 1, pp. 50-62, 1986.

[NEDZ90] Nedzel, J. L., *SAGA - Spatial and Geometric Analyzer*, Ph.D. Research Proposal, Department of Civil Engineering, Stanford University, Stanford, CA, March 1990.

[PHAN90] Phan, D. H. and Howard, H. C., *Evaluation of the Structural Steel Framing Data Model,* Technical Report No. 41, Center for Integrated Facility Engineering (CIFE), Stanford University, Stanford, CA, November 1990.

[SMIT82] Smith, B. C., "Reflection and Semantics in a Procedural Language," Ph. D. thesis and Technical Report MIT/LCS/TR-272, M.I.T., Cambridge, MA, 1982.

[STEF90] Stefik, M., Introduction to Knowledge Systems, Book Draft, Xerox Palo Alto Research Center, Xerox Corporation, Summer 1990.

[TATU90] Tatum, C. B., "Managing Integration Technology for Engineering and Construction," *Management of Technology Research Thrust in CIFE,* Center for Integrated Facility Engineering (CIFE), Stanford University, Stanford, CA, March 1990.

[TSIC82] Tsichritzis, D. C. and Lochovsky, F. H., *Data Models*, Prentice-Hall, Englewood Cliffs, NJ, 1982.

[WANG88] Wang, J., *Design-Dependent Knowledge For Structural Engineering Design*, Ph. D. Thesis Proposal, Department of Civil Engineering, Stanford University, Stanford, CA, October 1988.

[WANG90] Wang, J., *Applying Design-Dependent Knowledge in Structural Engineering Design*, Draft of Progress Report, Department of Civil Engineering, Stanford University, Stanford, CA, November 1990.

[WIED80] Wiederhold, G., "The Structural Model for Data Base Design," *Proceedings International Conference on the Entity-Relationship Approach to System Analysis and Design*, 1980.

[WIED86] Wiederhold, G., "Knowledge versus Data," *On Knowledge Base Management Systems: Integrating Artificial Intelligence and Database Technologies,* pp. 77-82, edited by M. L. Brodie, and J. Mylopoulos, Springer-Verlag, NY, 1986.

[ZDON90k] Zdonik, S. B. and Maier, D., "Fundamental of Object-Oriented Databases," *Readings in Object-Oriented DataBase Systems*, The Morgan Kaufmann Series in Data Management Systems, (series) edited by Jim Gray, Morgan Kaufmann Publishers, Inc., 1990.

APPENDIX A:

CIFE Data Inventory Questionnaire Form

CIFE DATA INVENTORY - QUESTIONNAIRE

RESEARCH / SYSTEM:

INVESTIGATORS & SPONSORS:

CIFE TRUST AREA:

I. General Research/System Description

What is the **specific purposes** of the research being conducted or of the system being built ?

Who are the **users** ?(area of specialization, degree of technical expertise, level of computing skills)

What **stages of the project** does the research/system support? (preliminary design, conceptual ...)

List any other CIFE system or application that **interacts** with this system (e.g., CAD, data base).

Does it exchange data with these systems/applications? How?

II. Data, Knowledge and Data Model

II.A Data

[Data: A unit of information that has a factual basis and records about a phenomenon]

What is the **domain** of the data/knowledge used ? (architectural design, structural steel design...)

What are the **input data** (before any computation)? Are they distinctive ? atomic ? measurable ?

What are the data **sources** (origin and owner) ?

What **stages of the project** do they pertain to? (preliminary design, conceptual design ...)

What are their **types** ? (numeric, characters, graphical, symbolic)

What are the **derived or output data** (that are calculated and/or used) ?

Where is the data **stored** (e.g. application files, spreadsheet, relational data base ...)

II.B Knowledge

[Knowledge: Informational resources that are used in processing and using the above data]

Is there any **general knowledge** ? (physics, chemistry, social sciences, computing sciences ...)

Is there any **domain knowledge** ? (definition, domain constraint, heuristic, first principle)

Is there any **design requirement** on the purpose, function, performance of the design object ?

Is the **function** of physical things described in the system ? How is it represented ?

Is the **behavior** of physical things described in the system ? How is it represented ?

II.C Use of Data / Knowledge

What is the **methodology** used in the system ?

What are the **general functions, control, and heuristics** used in solving the problem ?

II.D The Model

What is the type of model ? (theory model, device model, model for implementing the system)

Does your model fit into one of these categories: hierarchical, network, relational, object-oriented? Or is there any data modelling methodology used in building the model ?

What are the **objects or entities** of the model ?

What are the **primary object/entity relationships** used in the model ?

What are the evidences that **the model work well** to support the reasoning in the application ?

III. Inputs to the CIFE Data Inventory

Do you agree that there is a need and utility in developing an integrated CIFE data model ?

What input do you have to the format and content of this questionnaire ?

APPENDIX B:

Data Analysis/Summary Forms

(With Documentation)

ABOUT THE RESEARCH ...

RESEARCH []: References	INVESTIGATORS { *: Student; (): Advisor; **: CIFE Fellow }	OBJECTIVES	SALIENT ASPECTS	SYSTEM BUILT	IMPLEMENTATION PLATFORM [Software & Hardware]
TITLE OF RESEARCH PROJECT [ANY REFERENCE MATERIAL]	Name of the primary investigators of the research projects: *: Student (): Sponsoring Faculty Member **: CIFE Fellow	Objectives of the research	Salient aspects about the research which uniquely distinguish this research from other works in the same area. For instance, unique contributions of the research can be listed in this column.	Name of the software system built in the research project.	Software: Software implementation tools used in building the system (if applicable). Hardware: Available hardware platforms which the built system runs on (if applicable).

ABOUT THE SYSTEM / MODEL : REPRESENTATION

RESEARCH	SYSTEM / MODEL DESCRIPTION	DOMAIN	TYPES OF KNOWLEDGE (Knowledge Level)	Representation (Symbol Level)		
				STRUCTURES		FUNCTIONS
				Objects / Entities	Relationships	(operations, methods, etc.)
TITLE OF RESEARCH PROJECT [ANY REFERENCE MATERIAL]	Description of the nature, general purpose and functionality of system or model developed.	Domain of application of the system built. The scope of the research project can be stated in this column.	Types of knowledge which are represented in the system and needed in order to solve the problem. This relates to the *knowledge level* of the system whose focus is upon what is said about the real world that the system represents, rather than what symbolic structures are used in representing knowledge. The *symbol level* is described in the next three columns.	Objects / entities which are represented in the system.	Relationships among objects / entities represented in the system.	Functions included in the representation of the systems. Procedures, rule sets, objects' methods can be listed in this column.

NOTE:

- We use the word "system" in general in this writing, but some of the projects involve the development of some model or data base schema.
- Comments or general categories of items in the table are shown in italics .

ABOUT THE SYSTEM / MODEL: REASONING

RESEARCH	REASONING		
	Paradigm / Principles	**Tasks**	**Control / Heuristic Knowledge**
TITLE OF RESEARCH PROJECT [ANY REFERENCE MATERIAL]	Reasoning Paradigm or principles used in solving the problem For example: generate-test-modify, blackboard problem-solving paradigm, hierarchical planning, analysis-synthesis-evaluate, etc.	Tasks to be performed in solving the problem	Control and/or heuristic knowledge used in the reasoning process for solving the problem. Control knowledge: strategic knowledge used in guiding the problem solving process. Heuristic knowledge: experiential knowledge which comprises of rules of thumbs used in solving a particular problem.

ABOUT THE DATA / KNOWLEGE ...

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
TITLE OF RESEARCH PROJECT [ANY REFERENCE MATERIAL]	Name of data (or knowledge) item. What is listed in this column depends on the the representational scheme of the system: • Frame-based: Name of a frame or its slots • Object-oriented: Name of an object class and its attributes • Logic-based: Name of a constant, variable, predicate or function • Relational (data base): Name of a relation and its attributes. *Notation*: • Italics for comments or general categories of data items • All lower case for attributes of object classes, frames, relations. • Upper-lower case for object classes, frames, relations, or any object term in logic-based representation. • Bold face for discriminatingly important data items • Idented object names are used for subclasses in contrast to their superclass	Structure that is used in the organization of the data item and that contains the item. What is listed in this column also depends on the the representational scheme of the system: • Frame-based: Frame or slot of a frame • Object-oriented: object class or attribute of the object class • Logic-based: Constant, variable, predicate or function • Relational (data base): Relation or attribute of a relation.	Type of the data item. It can be simple data types or more complex user-defined data types. The two simple data types of general nature that we commonly use is Numeric or Symbolic. In frame-based or object-oriented systems, a slot/attribute is a place holder for a pointer to another frame or object.	• Use of the data item. By "use", we mean the manipulation or reasoning about the data item in order to perform problem-solving tasks for the particular application. When the use of the data item is rather obvious or implicit to the reader, we may state in this column some additional explanation or general comments about the data item. In such cases, we denote the explanation with the { } symbol. Input or output data are also designated in this column.

NOTE:

- Although we mention primarily about data in this form, the scope of our inventory effort also includes knowledge in the system.
- **This is only a summary!** It is not intended to be the complete list of all data or knowledge in the system.

APPENDIX C:

Data Summaries of Selected CIFE Data-Intensive Reseach Projects

About the Research ...

Suggestion: *Before reviewing the information in this Appendix, please consult Appendix B which explains what is included in each column of the enclosed summary tables.*

Disclaimer: *The information presented in this Appendix was verified and is correct up to the time when we complete this report (January 1991). Some of these systems reported herein will continue to evolve in the later development or enhancement stages.*

ABOUT THE RESEARCH ...

RESEARCH []: References	INVESTIGATORS { *: Student; (): Advisor; **: CIFE Fellow }	OBJECTIVES	SALIENT ASPECTS	SYSTEM BUILT	IMPLEMENTATION PLATFORM [Software & Hardware]
KNOWLEDGE-BASED LAYOUT GENERATION SYSTEM [CHIN90a,b]	Paul Chinowsky * (P. Teicholz)	• Development of a knowledge model for design synthesis • Identification of layout heuristic knowledge used by architectural designer • Development of a design synthesis approach which best emulates the way the architectural designer generates layout	• Knowledge-based approach to architectural layout • Knowledge model for design synthesis • Designer oriented system development approach rather than computing tool oriented approach	Computer-Aided Architectural DesIgn Expert (CAADIE)	• Software: KEE • Hardware: SUN Workstation
LOGIC-BASED INTEGRATED STRUCTURAL DESIGN OF STEEL OFFICE BUILDINGS [JAIN90]	Deepak Jain * (H. Krawinkler)	• Formal methodology for conceptual structural design of steel structures. This research objective is also complemented from the research on Reasoning and Representation and for Integrated Structural Design by [LUTH90]. • Demonstration of the use of logic-based symbolic representation for conceptual structural design	• Methodical approach for automated floor plan generation • Cost/Value ratio as utility measure of design alternatives • Explicit representation of form, function and behavior • Incorporation of exogenous constraints in conceptual structural design	Floor Framing Generator (FFG) *Note*: This system is being currently enhanced by Dr. Jain and will be named *Galileo*. It incorporates the design of lateral support systems as well as gravity support systems of the structure.	• Software: EPIKIT (User interface developed using Allegro Common Lisp on the MacIntosh) • Hardware: MacIntosh

RESEARCH []: References	INVESTIGATORS { *: Student; (): Advisor; **: CIFE Fellow }	OBJECTIVES	SALIENT ASPECTS	SYSTEM BUILT	IMPLEMENTATION PLATFORM [Software & Hardware]
CONSTRUCTIBILITY IMPROVEMENT FOR THE PRELIMINARY DESIGN OF REINFORCED CONCRETE BUILDINGS [FISC89,90]	Martin Fischer * (C. B. Tatum)	• Construction knowledge base system to provide constructibility input to structural design of reinforced concrete structures • Increase design-construction integration and reduce inefficiency due to facility engineering project fragmentations	• Identify the representation and reasoning capabilities needed for automating constructibility input to structural design • Identify data needed from CAD database for constructibility evaluation	COnstruction Knowledge Expert (COKE)	• Software: KAPPA • Hardware: P-C
DEVELOPMENT OF CIFECAD SYSTEM (VERSION 2.0) [KOLO90]	M. N. Kolountzakis * Martin Fischer *	• Development of a high level CAD system built on top of AUTOCAD for building structural design	• High level commands to draw building structural design objects (e.g., columns, walls, beams) as opposed to lower level objects (e.g., line, rectangle, circle) • Facility to convert/write design object data into an ASCII text file for other applications such as CoKE [FISC89].	CIFECAD *Note*: There are two versions of CIFECAD: the original version developed by Kenji Ito and the revised version described herein.	• Software: AUTOCAD • Hardware: (available on) MacIntosh, Sun or P-C

RESEARCH []: References	INVESTIGATORS { *: Student; (): Advisor; **: CIFE Fellow }	OBJECTIVES	SALIENT ASPECTS	SYSTEM BUILT	IMPLEMENTATION PLATFORM [Software & Hardware]
AN OBJECT MODEL FOR INTEGRATED PROJECT MANAGEMENT [FRO90a,b,c]	Thomas M. Froese * (B. C. Paulson, Jr.)	• *Primary objective*: Develop an object model that provides a unifying modeling framework for integrated object-oriented construction project management systems • Generalize results from the object modeling for object-oriented system design in construction project management • Create a computer-based prototype system	• Identification of the basic elements of the practice of project management • Library of object-oriented software components for project management • An object model suitable for integrated, flexible and intelligent object-oriented project management software systems	A computer-based library of construction project management objects	• Software: Objective-C • Hardware: NEXT
DATA BASE SCHEMA FOR BUILDING DESIGN [CHEN90] *(Part of the PENGUIN project [LAW89])*	Lixin Chen * (K. H. Law)	• Design of a data base schema for building design in support of the PENGUIN project development • Demonstrate the concept of using the Structural Model [WIED80] to model complex engineering design data and support multiple engineering views	*Note*: • The data base schema is still in development, as of 11/90. • Data pertains to the preliminary design phase of reinforced concrete structures.	An underlying relational data base to be used in the PENGUIN project development	• Software: RDB-VMS • Hardware: VAX workstation

RESEARCH []: References	INVESTIGATORS { *: Student; (): Advisor; **: CIFE Fellow }	OBJECTIVES	SALIENT ASPECTS	SYSTEM BUILT	IMPLEMENTATION PLATFORM [Software & Hardware]
APPLYING DESIGN-DEPENDENT KNOWLEDGE IN STRUCTURAL ENGINEERING DESIGN [WANG88, WANG90]	Jenmu Wang * (H. C. Howard)	• Identify design-dependent knowledge for structural engineering design • Demonstrate the feasibility of using design-dependent knowledge in knowledge-based structural design system • Develop a prototype knowledge-based design system	New approach for developing knowledge-based structural design systems that: • Integrate both design-dependent and design-independent knowledge • Combine both case-based reasoning and traditional rule-based heuristic reasoning in problem solving.	Design Dependent and Independent System (DDIS)	• Software: Blackboard problem-solving environment built after BB1 [HAYE84] on top of KEE software • Hardware: TI Explorer

RESEARCH []: References	INVESTIGATORS { *: Student; (): Advisor; **: CIFE Fellow }	OBJECTIVES	SALIENT ASPECTS	SYSTEM BUILT	IMPLEMENTATION PLATFORM [Software & Hardware]
OBJECT-ACTION-RESOURCE PLANNING SYSTEM (OARPLAN) [DARW89, HOWA89b] *(Part of the research on Knowledge-based planning of facility projects [CIFE90])*	Adnan Darwiche * Dick McGrath * (R. E. Levitt and B. H. Roth)	*Current objectives:* • Develop an intelligent planning knowledge base system for construction projects that: • provides a natural and powerful constraint language for construction planning knowledge and • generates a construction plan by satisfying constraints expressed in that language. *Long-term goal of OARPLAN:* • Develop a planning system that can interpret descriptions of the facility at several stages of refinement of the project and provide the designer immediate feedback on construction planning and scheduling throughout the evolvement of the facility design process.	• OARPLAN automatically generates a project plan by reasoning about facility data uploaded from a CAD system. • Activity elaboration is performed in order to derive precedence relationships among project activities, as opposed to hardwire these relationships into the activity descriptions. • Deep Causal Reasoning about topological and spatial data of the facility is based on first principle knowledge (e.g., gravity support and topological enclosure) in order to derive these activity precedence relationships. • An activity is uniquely represented as a tuple <action> <object> <resources>, according to PIPPA [MARS87].	Object-Action-Resource PLANning system (OARPLAN)	• Software: The current version is developed using Allegro Common Lisp in conjunction with the Parmenidis frame-based system and FRules rule- based system. • Hardware: MacIntosh and DEC workstation. *Note*: There is an early version implemented on the TI Explorer.

APPENDIX C:

Data Summaries of Selected CIFE Data-Intensive Reseach Projects

About the System / Model: Representation

Suggestion: *Before reviewing the information in this Appendix, please consult Appendix B which explains what is included in each column of the enclosed summary tables.*

Disclaimer: *The information presented in this Appendix was verified and is correct up to the time when we complete this report (January 1991). Some of these systems reported herein will continue to evolve in the later development or enhancement stages.*

ABOUT THE SYSTEM / MODEL : REPRESENTATION

RESEARCH	SYSTEM / MODEL DESCRIPTION	DOMAIN	TYPES OF KNOWLEDGE (Knowledge Level)	Representation (Symbol Level)		
				STRUCTURES		FUNCTIONS
				Objects / Entities	Relationships	(operations, methods, etc.)
KNOWLEDGE-BASED LAYOUT GENERATION SYSTEM [CHIN90a,b]	A system to automatically generate an architectural layout based on the design requirements specified by the designer. This research also contributes a model of knowledge used by designer for architectural layout	Architectural layout of education buildings	• Topological Attributes (knowledge about physical dimensions of design objects) • Design Attributes (knowledge for the layout generation such as daylighting, space planning, etc.) • Spatial Ordering Concepts (knowledge of spatial ordering concepts such as linear or clustered) • Designer Expertise (in dealing with a particular building type) • Knowledge Selection Heuristics (used as control knowledge)	*Organized into frame hierarchies:* • Topological hierarchy: Design Objects, etc. • Design Attribute hierarchy: Design Attributes, etc. • Spatial Ordering Concepts hierarchy: Linear concept, Space requirements concept	• Space-to-space internal orientation relationships: North, South, East, West • Space-to-space adjacency requirement relationships: Required, Desired, Negative, Strongly negative • Space-to-site external orientation relationships: North, South, East, West	• Layout generation methods and rules • Attribute evaluation methods • Layout evaluation methods • Requirements generation methods of design objects such as labs, lecture rooms, etc.

RESEARCH	SYSTEM / MODEL DESCRIPTION	DOMAIN	TYPES OF KNOWLEDGE (Knowledge Level)	Representation (Symbol Level)		
				STRUCTURES		FUNCTIONS
				Objects / Entities	Relationships	(operations, methods, etc.)
LOGIC-BASED INTEGRATED STRUCTURAL DESIGN OF STEEL OFFICE BUILDINGS [JAIN90]	Automated generator of structural floor layout	Conceptual structural design of steel office buildings	• Structural Elements and Systems Knowledge • Behavior Knowledge (fundamental principles of structural engineering) • Performance Knowledge (knowledge about performance criteria imposed on the structure) • Product Knowledge (knowledge about specific products for construction) • Strategic Knowledge (or control knowledge) • Architectural and Structural Concepts (e.g. influence zone, column lines, etc.)	*Using logic-based symbolic representation [JAIN89]:* • Truth Symbols: True and False • Constants: symbols whose value never changes (e.g., Frame101, Seismic_Zone_0). • Variables: symbols whose value changes (e.g., x1, y2). • Logical Connectives: and, or, not, implies, equivalent • Quantifiers: universal (for all), existential (there exists)	Adjacency of structural members in term of logical sentences	*Using logic-based symbolic representation [JAIN89]:* • Functions: Functions or operators that operate on a fixed number of arguments and evaluate to a value (e.g., Sin, Log). • Predicates or Relations: Relation that hold among a fixed number of arguments (e.g., Even, NeigborOf). Terms: arguments of functions and relations. • Functional Expressions: this consists of a function following by a list of terms.

RESEARCH	SYSTEM / MODEL DESCRIPTION	DOMAIN	TYPES OF KNOWLEDGE (Knowledge Level)	Representation (Symbol Level)		
				STRUCTURES		FUNCTIONS
				Objects / Entities	Relationships	(operations, methods, etc.)
CONSTRUCTIBILITY IMPROVEMENT FOR THE PRELIMINARY DESIGN OF REINFORCED CONCRETE BUILDINGS [FISC89,90]	• Knowledge base system to provide constructibility inputs (without cost consideration) to structural design in a "advice when needed" mode • The system also provides an interface with CIFECAD (see next page) where designer can design and draw a structure.	Constructibility reinforced concrete structures during preliminary design	*Five design-relevant constructibility knowledge types:* • Application Heuristics (to determine the applicability of a construction method) • Layout Knowledge (to constraint the layout of structural elements) • Dimensioning Knowledge (to influence the dimensions of structural elements) • Detailing Knowledge (to affect the detailed design of the structure) • Exogenous Knowledge (about important exogeneous factors)	• *"Topological" model of the structure which is replicated from CIFECAD objects:* columns, walls, beams, slabs, etc. (See CIFECAD) • *Objects for construction methods such as:* • Forming • Shoring • Placing of concrete • Concrete Reinforcement	*Similar to CIFECAD object relationships*	• Functions to interpret data from CIFECAD • Functions which map the input CIFECAD data to constructibility knowledge in the knowledge base • Functions to perform reasoning about: • Object attributes • Relationships between object attributes • Spatial data

RESEARCH	SYSTEM / MODEL DESCRIPTION	DOMAIN	TYPES OF KNOWLEDGE (Knowledge Level)	Representation (Symbol Level)		
				STRUCTURES		FUNCTIONS
				Objects / Entities	Relationships	(operations, methods, etc.)
DEVELOPMENT OF CIFECAD SYSTEM (VERSION 2.0) [KOLO90]	• Small high-level CAD system that provides commands to draw building structural design objects (e.g., beams, columns, wall, etc.) and to extract design object data into an ASCII text file for other applications such as CoKE [FISC89]. • This is the second version of CIFECAD developed by Kolountzakis.	Limited to a number of common reinforced concrete structural design objects		*Common steel and reinforced concrete structural design objects:* • Columns (rectangular or round) • Walls (rectangular, one-thickness) • Beams (rectangular and V-shaped) • Slabs (uniform thickness, any polygon shape, no opening, no camber) • Drop Caps (rectangular, one-depth) • Doors (rectangular) • Windows (rectangular) • Foundations (rectangular, one depth): single footings, strips and mats	• Spatial relationships among physical objects: left, right, below, above, etc. • Supported-by relationship among structural objects • Connected-to relationship among structural objects	• General system functions • Functions to draw and manipulate these objects • Functions to extract design object data into an ASCII text file for other applications • Functions (written in C) to add back pointers for the relationship "above" in generating the ASCII text file

RESEARCH	SYSTEM / MODEL DESCRIPTION	DOMAIN	TYPES OF KNOWLEDGE (Knowledge Level)	Representation (Symbol Level)		
				STRUCTURES		FUNCTIONS
				Objects / Entities	Relationships	(operations, methods, etc.)
AN OBJECT MODEL FOR INTEGRATED PROJECT MANAGEMENT [FRO90a,b,c]	An object model for a large integrated project management system that: • Defines the objects' name, purpose and basic information processing capabilities (in terms of object methods) • Provides organization to the objects • Defines the major interactions among the objects	For construction project management application systems *The scope of the model is defined in three levels:* • Survey of all project management aspects • Detailed object design of a smaller subset of the previous • Full implementation of an even smaller subset of the previous.	*Three general categories of object definitions:* • General system objects (common generic object types that reside in general system libraries and can be used in different projects) • Project primitive objects (primitive objects representing basic elements of a particular project) • Typical application objects (objects that generate, refine or utilize the above project primitives objects for a particular application in the project management such as plan generation, cost/schedule control, accounting, etc.)	*Three levels of object definitions:* • An object class as a general "templates" for objects • a type (class) as a group of objects with similar characteristics • an individual object instance *Corresponding class hierarchies in the three categories of object definitions. Object classes represent:* • Resources • Components • Actions (or construction operations) • (construction work) Methods	*Common object classes relationships:* • Generalization / Specialization (subclass / superclass) • Aggregation (part-of / subpart) *A number of associative relationships with particular relationship names such as resource types, material types, etc. The latter are names of the objects that are pointed to.*	Construction operations and work methods are represented as object classes that are related to the physical component objects.

RESEARCH	SYSTEM / MODEL DESCRIPTION	DOMAIN	TYPES OF KNOWLEDGE (Knowledge Level)	Representation (Symbol Level)		
				STRUCTURES		FUNCTIONS
				Objects / Entities	Relationships	(operations, methods, etc.)
DATA BASE SCHEME FOR BUILDING DESIGN [CHEN90] *(Part of the PENGUIN project [LAW89])*	A relational data base schema for reinforced-concrete structures based on the Structural Model [WIED80]	Preliminary design of reinforced concrete structures		*Data are structured into relations. For the purpose of this summary, we identify the following general categories of relationsd:* • Buildings • *Structural Systems:* Super-structures, Storeys, Frames, Slabs, etc. • *General structural members:* Columns, Girders, Walls, Secondary Beams, etc. • *Reinforced-concrete structural members*: RC_Columns, RC_Girders, etc. • *Shape of structural members:* Wall_Shapes, Slab_Shapes	Relationships defined in the Structural Model (See Reference [WIED80]): • Ownership • Reference • Subset	

RESEARCH	SYSTEM / MODEL DESCRIPTION	DOMAIN	TYPES OF KNOWLEDGE (Knowledge Level)	Representation (Symbol Level)		
				STRUCTURES		FUNCTIONS
				Objects / Entities	Relationships	(operations, methods, etc.)
DATA BASE SCHEME FOR BUILDING DESIGN [CHEN90] *(Part of the PENGUIN project [LAW89])*				• *Cross Section Properties of structural members:* RC_Column_ Sections, RC_Column_ Designations, etc. • *Material*: R_C_Materials • *Supports of structural members:* Girder_Supports, Slab_Supports, etc. • *Connections of structural members*: Wall_Column_Conn ections, etc. • *Architectural Entities:* Floor Plans, Wall Openings, etc. • *Other entities:* Coordinate_System, Gridlines, Nodes, etc.		

RESEARCH	SYSTEM / MODEL DESCRIPTION	DOMAIN	TYPES OF KNOWLEDGE (Knowledge Level)	Representation (Symbol Level)		
				STRUCTURES		FUNCTIONS
				Objects / Entities	Relationships	(operations, methods, etc.)
APPLYING DESIGN-DEPENDENT KNOWLEDGE IN STRUCTURAL ENGINEERING DESIGN [WANG88, WANG90]	Integrated knowledge-based structural engineering design system that incorporates both design-dependent and design-independent knowledge	An example domain that has been used for early system prototyping is for structural steel beam-column design.	• Design-dependent knowledge which consists of memories of previous good (and bad) designs and design strategies. There are plan, goal, and solution memories. • Design-independent knowledge which include, in this case, abstract reasoning rules that are generalized and independent of specific designs.	• *The objects of the DDIS Design Model are represented in the KEE frame-based paradigm.* • Data Items (e.g., function, rule set, condition, rule, user query, mapping) • *Design Solution Objects:* • *Physical Objects* (e.g., Beam, Column, etc.) • Variable Objects • Constraint Objects • *Design Control Objects:* • Plan • Goal • Retrieved Design • Design Action *Objects:*	• Common generalization/specialization relationships among classes in the same hierarchy • Named slots contains pointers to other reference objects.	Objects have implemented methods: • Constraint objects have a method to evaluate the current condition of their intrinsic constraint. • Variable objects have a method to evaluate themselves and return the value. • Design Action Objects have a method to execute their intrinsic action which modifies the content of control and design information blackboards.

RESEARCH	SYSTEM / MODEL DESCRIPTION	DOMAIN	TYPES OF KNOWLEDGE (Knowledge Level)	Representation (Symbol Level)		
				STRUCTURES		FUNCTIONS
				Objects / Entities	Relationships	(operations, methods, etc.)
APPLYING DESIGN-DEPENDENT KNOWLEDGE IN STRUCTURAL ENGINEERING DESIGN [WANG88, WANG90]				• Knowledge Source • Knowledge Source Activation Record (KSAR) • Design Case *Objects* • *Knowledge sources can be classified into domain and control knowledge sources. These two types can be further categorized design-dependent and design-independent knowledge sources. These sources are organized in four modules:* • Design-independent Reasoner • Case Memory Knowledge Base • Design-dependent Reasoner • Case Recorder.		• Goal object class has a method to evaluate Knowledge Source Activation Records (KSAR) posted on the triggered agenda.

RESEARCH	SYSTEM / MODEL DESCRIPTION	DOMAIN	TYPES OF KNOWLEDGE (Knowledge Level)	Representation (Symbol Level)		
				STRUCTURES		FUNCTIONS
				Objects / Entities	Relationships	(operations, methods, etc.)
OBJECT-ACTION-RESOURCE PLANNING SYSTEM (OARPLAN) [DARW89, HOWA89b] *(Part of the research on Knowledge-based planning of facility projects [CIFE90])*	• OARPLAN is a planning system that automatically generates construction project plans (list of needed activities and their precedence constraints) from the input topological and spatial data of the facility uploaded from a CAD system. In this case, the latter is CIFECAD. • OARPLAN reasons about objects, actions and resources associated with the construction activities in the project in order to produce the construction plan.	Construction project planning of reinforced concrete and steel structures	• Knowledge about the architectural, structural and topological aspects of design objects of the facility to be built. • Knowledge about project planning that centers around actions in construction. • Knowledge about resources (not incorporated in the current prototype system) • Knowledge about action elaboration and activity dependency for planning purpose.	• ***Object:*** *These are modeled after the Data Model for Building Design [LAW86]. Building components of the facility are organized in three aggregational hierarchies: architectural, structural and topological. These hierarchies provide the corresponding views of the root node object* Building*:* Building General_Description Representational_ Description View Architectural_ View Structural_ View Topological_ View	*Relationships in OARPLAN are derived from geometric and topological data of building components.* *As reported in [DARW89], there are fundamentally four types of relationships:* • Spatial: adjacent-to, in-same-floor, enclosed-by (not implemented) • Topological: supported-by, etc. • Sequential activity dependency: before, after. These are temporal relationships.	

RESEARCH	SYSTEM / MODEL DESCRIPTION	DOMAIN	TYPES OF KNOWLEDGE (Knowledge Level)	Representation (Symbol Level)		
				STRUCTURES		FUNCTIONS
				Objects / Entities	Relationships	(operations, methods, etc.)
OBJECT-ACTION-RESOURCE PLANNING SYSTEM (OARPLAN) [DARW89, HOWA89b] *(Part of the research on Knowledge-based planning of facility projects [CIFE90])*	*Note*: • The current prototype system does not allocate project resources nor compute activities' duration for scheduling purpose. • Future extensions of the prototype system will include these capabilities, as well as incorporate other features such as time-cost tradeoff simulation, project risk identification, etc. In the long term goal, OARPLAN will include facilities to provide real-time feedback to the designer in terms of design change impacts on construction project planning and scheduling.			*Architectural Objects:* Floor, Floor-Space, Room, Corridor, etc. *Structural Objects:* Column, Beam, Girder, Slab, etc. *Topological Objects:* Space-3D (not implemented) *Other objects*: Location, Wall-Specification, Door-Specification, etc. • ***Action***: An action can be simple (e.g., installing <Bolt>) or compound (construct <Floor-1>). • ***Resource***: Not implemented in the current prototype system.	Other relationships not implemented in the current version are requires, causes and lags. • Aggregational relationships among activities. These relationships falls into two categories: • subpart: This relationship is used in the decomposition of an activity into a plan of subactivities. • contain: This relationship is used in reducing the scale of high level abstract activity into constituent activities of smaller scale along different dimensions (e.g., for time/cost estimation control, for overall project duration)..	

RESEARCH	SYSTEM / MODEL DESCRIPTION	DOMAIN	TYPES OF KNOWLEDGE (Knowledge Level)	Representation (Symbol Level)		
				STRUCTURES		FUNCTIONS
				Objects / Entities	Relationships	(operations, methods, etc.)
OBJECT-ACTION-RESOURCE PLANNING SYSTEM (OARPLAN) [DARW89] *(Part of the research on Knowledge-based planning of facility projects [CIFE90])*				• ***Activity***: This object is represented as a tuple: <action> <object> <resources>.	In the current version of OARPLAN, relationships are implemented as object classes. The following relationship classes exist: • has-parts and inverse part-of • has-view and inverse view-of • has-location • other relationships and their inverse: rep(resentational)-description, gen(eral)-description, has-specs, has-bounding-member and has-bounding-wall	

APPENDIX C:

Data Summaries of Selected CIFE Data-Intensive Reseach Projects

About the System / Model: Reasoning

Suggestion: *Before reviewing the information in this Appendix, please consult Appendix B which explains what is included in each column of the enclosed summary tables.*

Disclaimer: *The information presented in this Appendix was verified and is correct up to the time when we complete this report (January 1991). Some of these systems reported herein will continue to evolve in the later development or enhancement stages.*

ABOUT THE SYSTEM / MODEL: REASONING

RESEARCH	REASONING		
	Paradigm / Principles	**Tasks**	**Control / Heuristic Knowledge**
KNOWLEDGE-BASED LAYOUT GENERATION SYSTEM [CHIN90a,b]	Analysis-Synthesis-Evaluation	• Requirements Generation from the design objects of spatial layout • Space Selection • Placement Selection • Design Attributes Checking • Placement Refinement • Placement Update • Layout Evaluation There is also the generation of different layout design alternatives. However, this task must be initiated by the user.	Mainly heuristic knowledge: • Design Attributes heuristics • Spatial Ordering heuristics • Knowledge Selection heuristics • Design Expertise This knowledge is in the form of rule sets.

RESEARCH	REASONING		
	Paradigm / Principles	**Tasks**	**Control / Heuristic Knowledge**
LOGIC-BASED INTEGRATED STRUCTURAL DESIGN OF STEEL OFFICE BUILDINGS [JAIN90]	*Reasoning with structural and exogenous constraints:* • Constraints Formulation • Constraints Propagation • Constraints Satisfaction	• Generation of column locations through consideration of areas outside and inside the core • Configuration of the floor system: floor type selection, generation of beam and girder locations • Design of members: selection of steel grade and approximate sizing of beams, girders and columns • Cost estimation (also account for lateral systems connections) • Evaluation of design alternatives based on Cost/Value ratio and feedback Also generation of different floor plan design alternatives	Strategic knowledge relates to the meta-level knowledge about how to use other knowledge in the system and guide the design process. General examples of strategic knowledge in this system include: • Knowledge about decomposing the problem • Knowledge about when to formulate what constraints • Knowledge about how to utilize certain concepts. A more specific example is the minimum weight strategy used in selecting steel sections for the structural elements in the floor system. Most of the strategic knowledge in the system is predominantly based on heuristics.

RESEARCH	REASONING		
	Paradigm / Principles	**Tasks**	**Control / Heuristic Knowledge**
CONSTRUCTIBILITY IMPROVEMENT FOR THE PRELIMINARY DESIGN OF REINFORCED CONCRETE BUILDINGS [FISC89,90]	*Three types of reasoning:* • Reasoning about attributes of objects • Reasoning about relationships among attributes of objects • Spatial reasoning	• Construct the "topological" model of the structure from transferred CIFECAD data in ASCII file format • Map the input CIFECAD data to the constructibility knowledge in the knowledge base in order to select construction methods • Apply application heuristics for the selected construction methods • Update the list of possible construction methods • Apply layout and dimensioning knowledge to the updated list of construction methods • Generate constructibility feedback on the screen	Constructibility Knowledge in COKE is predominantly heuristics gathered from: • Literature • Four types of experts: designers, general contractors, subcontractors and suppliers of construction materials and equipments. There is also meta-level knowledge about which method of construction to be used for a particular structural design object.

RESEARCH	REASONING		
	Paradigm / Principles	**Tasks**	**Control / Heuristic Knowledge**
APPLYING DESIGN-DEPENDENT KNOWLEDGE IN STRUCTURAL ENGINEERING DESIGN [WANG88, WANG90]	• Hierarchical planning: Design planning follows a top-down decomposition approach. • The generate-test-modify problem solving paradigm is implemented via a blackboard architecture. • The blackboard architecture enables opportunistic reasoning using both design-dependent and design-independent knowledge sources. There are two blackboards: • Control blackboard which has three levels of control objects: plan, goal and retrieved design. • Design information (or solution) blackboard which dynamically stores information generated by various design knowledge sources such as action and evolving solution objects. • Truth Maintenance System (TMS) in DDIS: it keeps track with the design variable dependencies and is responsible for updating the current design at different stages in the problem-solving process.	The problem-solving process follows execution cycles, each of which consists of the following tasks: • Updating control objects on the control blackboard • Applying knowledge sources to the updated blackboard situation. This task consists of three steps directed by the global control object: • Triggering appropriate knowledge sources • Creating knowledge source activation records (KSAR) for every applicable content of each triggered knowledge source. KSAR are posted on the triggered agenda. • Verifying the preconditions of the KSARs for executability by the triggered knowledge sources. Executable KSARs are moved to the executable agenda. • Rating the KSARs by the scheduler and executing them by the executor according to the rating. Posting the execution results on the blackboard. Repeat the cycle until a successful solution is found.	• Control knowledge is stored in control knowledge sources and control objects such as plan, goal and retrieved design. Control objects are posted on the control blackboard, control knowledge sources are applied to post, modified and remove these control objects. • Design-independent heuristic knowledge is in the form of abstract design rules in the Design-independent Reasoner knowledge module.

RESEARCH	REASONING		
	Paradigm / Principles	**Tasks**	**Control / Heuristic Knowledge**
OBJECT-ACTION-RESOURCE PLANNING SYSTEM (OARPLAN) [DARW89, HOWA89b] *(Part of the research on Knowledge-based planning of facility projects [CIFE90])*	• Product-oriented construction planning: the final goal is to produce some product that meets the design requirements. • Hierarchical planning in term of top-down elaboration of the project activities • Deep Causal Reasoning to derive activity dependency relationship. It is based on first principle knowledge such as gravity support, topological enclosure, etc. There are two ways: utilizing dependencies inherited from activity subplanning; or inferring dependencies by applying dependency KS. *Note*: The blackboard problem-solving paradigm was only implemented in the earlier version on the TI Explorer.	• Activity Elaboration: There are two forms of activity elaboration: activity subplans and activity scale reduction. The former involves the decomposition of an activity into its subplan of activities; the latter performs scale reduction of high level abstract activity into smaller scale constituents along different dimensions (e.g., for time/cost estimation control, for overall project duration). • Execution Cycle: The execution cycle in the blackboard problem-solving paradigm is done in the following manner: • Triggering appropriate knowledge sources • Posting recommendations from knowledge sources whose triggered conditions are satisfied. Recommendations are in the form of knowledge source activation records (KSAR) posted in the triggered agenda. • Rate KSARs and execute the one with the highest score. Repeat the cycle until a successful solution is found.	• Elaboration Knowledge Sources (KS) which contain activity elaboration rules. There are two types: • Specific KS applies to activities which remain constant across the project • Generic KS applies to activities which vary across the project • Dependency Knowledge Sources which contain rules for determining dependencies among activities • In the early version of OARPLAN which utilizes the blackboard architecture, heuristic knowledge is used in rating the knowledge source activation records (KSAR) which will be transposed to the executable agenda.

APPENDIX C:

Data Summaries of Selected CIFE Data-Intensive Reseach Projects

About Data / Knowledge ...

Suggestion: *Before reviewing the information in this Appendix, please consult Appendix B which explains what is included in each column of the enclosed summary tables.*

Disclaimer: *The information presented in this Appendix was verified and is correct up to the time when we complete this report (January 1991). Some of these systems reported herein will continue to evolve in the later development or enhancement stages.*

ABOUT THE DATA / KNOWLEGE ...

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
KNOWLEDGE-BASED LAYOUT GENERATION SYSTEM [CHIN90a,b]	*In the topological attributes frame hierarchy:* **Design Objects** Floors Sites Spaces Special-purpose Spaces according to space type (classrooms, common areas, offices, labs) *Physical Data:* square footage area size ratio length width dimensioning variance percentage	Root object class of the topological attributes frame hierarchy - Floors, Site and Spaces are subclasses of Design Objects - - Subclasses of Spaces which differ from one another by the default values of their attributes - Common attributes of the above object classes	Numeric Numeric Numeric Numeric Numeric	* The output of the system is the layout of the architectural spaces. { Site has primary and secondary views that provides view references to spaces. } - • Used as physical data requirements in spatial layout • Used in determining the relationships (e.g., adjacency) between spaces. • To check for space overlapping, view blocking * These are Input data. -

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
KNOWLEDGE-BASED LAYOUT GENERATION SYSTEM [CHIN90a,b]	*On- Site Location Data:*	- Attributes of objects in the topological attributes frame hierarchy		
	coordinates		Numeric	
	distances between spaces	-	Numeric	
	In the architectural design attributes frame hierarchy:			
	Design Attributes	Root object class of the architectural attributes frame hierarchy		• Used as requirements and guidelines in the architectural layout
	Daylighting	Subclass of Design Attributes		
	type	- Attributes of Daylighting	Symbolic (e.g., Direct, non-direct, etc.)	* Input
	orientation	-	Symbolic (e.g., North, South, East, West)	* Input
	Human Factors	Subclass of Design Attributes		
	Privacy Level	- Subclasses of Human Factors object	The value of its attribute is symbolic (e.g., Public, Semi-Public, Private)	- * Attributes of these classes require Input data.
	Security Level		- The value of its attribute is symbolic (e.g., High, Low, Medium)	
	Acoustic Level		-	
	View Requirements	-	The value of its attribute is symbolic (e.g., Primary, Secondary)	-

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
KNOWLEDGE-BASED LAYOUT GENERATION SYSTEM [CHIN90a,b]	Space Planning Adjacency (Positive, Desired, Negative, Strongly negative) *Spatial Ordering Concepts:* Linear Concept Requirement-oriented ordering Concept	Subclass of Design Attributes class Subclasses of Space Planning These concepts are partially included in objects in the Solution concepts frame hierarchy, but mainly in the form of heuristic rules about spatial ordering.	These are actually adjacency relationships but are implemented as object classes.	• Used as architectural space organization paradigms in layout generation

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
LOGIC-BASED INTEGRATED STRUCTURAL DESIGN OF STEEL OFFICE BUILDINGS	***Building:***	*In logic-based representation, the following data are either represented as constants, variables, functions, predicates or function expressions.*	*In this column, I designate the value of the constants, variables or the value returned by the functions and functions expressions.*	* All variables shown in this table required Input data.
[JAIN90]	Number of Stories	Variable	Numeric	• Used in computing the magnitude and distribution of lateral loads (wind & seismic); estimating material saving between using lightweight or normal weight concrete; computing total building cost; and deciding column designs for different stories.
	Floor-to-Ceiling Height	Variable	Numeric	• Used in computing the effective length of columns for sizing them; determining the overall height of the building in order to estimate lateral loads; estimating steel weight of columns which effect material and transportation cost.
	Site Exposure Category	Variable	Constant (e.g., B, C)	• Used in computing wind loading
	Seismic Zone Factor Z	Variable	Numeric (e.g., 0 through 4)	- Used in computing seismic loading
	Site Soil Coefficient	Variable	Constant (e.g., S1 through S4)	-
	Importance Factor for wind loading	Variable	Numeric	Used in computing wind loading
	for seismic loading	Variable	Numeric	Used in computing seismic loading

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
LOGIC-BASED INTEGRATED STRUCTURAL DESIGN OF STEEL OFFICE BUILDINGS [JAIN90]	***Architectural Floor Plan:***			
	Spaces			
	Coordinates	Predicate	Numeric	* Input
	(Architectural) Function	Constant	Symbolic	* Input { e.g., staircase, lobby, perimeter, core, misc., elevator-bank, shaft }
	Opening	Predicate	Truth Symbol	
	No Column Zone	Predicate	Truth Symbol	
	Structural Systems: *Floor Plate, Diaphragm, Moment-Resisting Frame*			{ This is one form of knowledge in the system: structural elements and systems knowledge. It includes global static information about structural elements and systems. The value of the attributes will always be true. }
	* *Form:*			
	Dimensionality (1D, 2D, 3D)	Predicate	Constant	
	Orientation (Vertical, Horiz.)	Predicate	Constant	
	Coordinates Data	Predicate	Numeric	{ Only applicable to moment-resisting frame }
	System Material Designation	Predicate	Constant (e.g. A-36 steel, A-50 steel, light weight concrete, normal weight concrete)	* Input
	* *Function:*	Predicate		{ e.g., "To transfer applied loading in a vertical direction" }
	* *Behavior:*			
	Primary Behavior	Predicate	Constant	- { e.g., Axial Tension, Compression, Flexure, Shear, Shear Lag }
	Other Behaviors	Predicate	Constant	-
	* ***External Loading:***			
	Loads			

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
LOGIC-BASED INTEGRATED STRUCTURAL DESIGN OF STEEL OFFICE BUILDINGS [JAIN90]	Load Type	Variable	Constant	- • These attributes are used in computing external loads applied on the structure in different load combinations. The external load types considered in FFG include dead loads, live loads, wind loads, and seismic loads. -
	Load Magnitude	Predicate	Numeric	
	Load Direction	Variable	Constant (vertical, lateral-xx, lateral-yy)	
	Load Location (in terms of coordinates)	Variable	Numeric	
	Load Factor	Variable	Numeric	
	Load Combinations	Function	Numeric	• To represent combinations of loading applied on the structure (e.g., as per ANSI, 1.4D, 1.2D + 1.6L + 0.5 (Lr or S or R)).
	Structural Members: Column, Beam, Girder, Wall, Shear Stud, Hanger			
	** Form:*			
	General Location	Variable	Constant (Exterior, Interior)	- • Used in determining the required strength of the member (also referred to as Demands)
	Coordinates Data	Variable	Numeric	* Ouput
	Supported Members	Variable	Constant (in term of the names of the supported members)	-
	Member Span	Variable	Numeric	- • Used in determining the available strength of the member (also referred to as Capacities).
	Member Shape	Variable	Constant	
	Member Depth	Variable	Numeric	
	Story number	Variable	Numeric	* These are Output data. Beam and girder depth have max and min values as Input data.
	Member X-Section Type	Variable	Constant	-
	** Behavior:*			- • Used in structural member design.
	Demand vs. Capacity	Function	Numeric	
	Deflection	Function	Numeric	-

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
LOGIC-BASED INTEGRATED STRUCTURAL DESIGN OF STEEL OFFICE BUILDINGS [JAIN90]	***Cost:***			- Used in cost estimation.
	Material Cost	- Functions	Numeric	* Material cost data are Input. There are default cost values.
	Fabrication Cost		Numeric	
	Erection Cost		Numeric	{ FFG has material costs for light and normal weight concrete, A-36 and A-50 steel, wire fabric, decks and studs. }
	Auxiliairy Cost	-	Numeric	{ Auxiliary Cost includes transportation, shoring, cambering, and fire proofing } -
	Total Project Cost		Numeric	* Output
	Product Knowledge:			
	* *Material Properties: (Young modulus, etc.)*	Functions	Numeric	• Used in member sizing
	Steel			
	Concrete			
	* *AISC hot-rolled steel member shape properties*	In a separate data base of predicates representing these standard cross-section properties	Numeric	• Used in selecting sections for beams, girders, and columns. The current restriction in FFG member design is to use standard wide flange shapes.
	* *Vendor-dependent product data: Properties of composite metal decks from manufacturers' catalog (Vulcraft Mfr.)*	In a separate data base of predicates		- • Used in selecting decks for floor plate
	Deck spanning capability	- Predicates	Numeric	
	Self weight of composite deck		Numeric	-
	Unit volume of concrete		Numeric	
	Recommended wire fabric	-	Constant	

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
LOGIC-BASED INTEGRATED STRUCTURAL DESIGN OF STEEL OFFICE BUILDINGS [JAIN90]	** Pricing information about material / labor cost:*	In a separate data base of predicates	Numeric	• Used in cost estimation
	Behavior / Performance Knowledge:			
	** First principle knowledge:*			
	Stress	- Functions	Numeric	e.g., $f = Mc / I$
	End Reactions		Numeric	e.g., $M = wl^2/8$
	Deflection		Numeric	e.g., $\partial = 5wl^4/384EI$
	Principle of superposition	-		
	** Other:*			
	Strength Reduction Factors	Constants		{ Different reduction factors are used for different behaviorial modes (e.g. tension yielding, tension fracture, compression, bending, shear) }
	Architectural / Structural / Other Concepts:			
	Characteristic Dimension (CD)	Predicate	Numeric	• Used for describing geometric patterns
	Influence Zone	Predicate	Numeric (in term of its coordinate data)	
	Column lines	Predicate	Numeric (in term of its coordinate data)	• Used as a design constraint in laying out the columns' location.
	Design Constraints:			
	**** Structural***			• Constraints used in the design
	(Load-carrying) Function Constraints	Variable	Numeric	{ Due to external loads }

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
LOGIC-BASED INTEGRATED STRUCTURAL DESIGN OF STEEL OFFICE BUILDINGS [JAIN90]	Min and Max Economic Spans	Variable	Numeric	• Governs the span of structural members
	Min and Max Economic Spacings	Variable	Numeric	• Governs the spacing of the floor members and thus the spanning capabilities of the overlying steel deck and the incident loading.
	Minimum and Maximum Depths of Beams and Girders	Constanst	Numeric	• To accomodate the design of mechanical ducts through the structural system
	Fire Resistance			• Influences the concrete thickness on top of the metal deck. Or requires spay fireproofing.
	Performance Constraints:			{ Beside safety and serviceability constraints, there are other performance constraints which are derived from Performance Knowledge (e.g., effective flange width, live load reduction, beam-column interaction equation). }
	Safety Constraints	Predicate		{ Demand <= Capacity }
	Serviceability Constraints	Predicate		e.g., maximum permissible deflections

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
LOGIC-BASED INTEGRATED STRUCTURAL DESIGN OF STEEL OFFICE BUILDINGS [JAIN90]	***Product Knowledge Constraints***			{ There are a number of constraints generated from the product knowledge (See Product Knowledge in previous pages) such as concrete type (normal or light weight concrete), concrete strength, grade of steel, etc. }
	Constructibility Constraints			{ There are a number of constructibility constraints considered in FFG: • know-how in construction methods (e.g., flexural member construction: shoring, cambering, composite) • frequency of splicing column • material availability constraints (grade of steel, type and strength of concrete, type of shear studs); etc. }

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
CONSTRUCTIBILITY IMPROVEMENT FOR THE PRELIMINARY DESIGN OF REINFORCED CONCRETE BUILDINGS [FISC89,90]	***Project and General Building Data:***			
	General Data	Object Class		• To encapsulate both project and general building data
	(project) name	-	Symbolic	- * These are Input data.
	client	Attributes of Project	Symbolic	Most data items about project, building, structural systems and members shown here are input data to the system (COKE); the only type of output from COKE is constructibility feedback in the form of text displayed in windows during the run session.
	location		Symbolic	
	use		Symbolic	
	designer		Symbolic	
	(project) duration		Numeric	
	(project) budget		Numeric	
	total floor area		Numeric	
	total (building) height		Numeric	
	type of vertical struct. syst.		Symbolic	
	type of horizont. struct.syst.		Symbolic	-
	number of levels		Numeric	
	number of columns		Numeric	
	number of round columns		Numeric	
	number of rectangular columns		Numeric	
	number of beams		Numeric	
	number of walls		Numeric	
	number of slabs		Numeric	
	number of drop caps		Numeric	
	number of windows	-	Numeric	

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
CONSTRUCTIBILITY IMPROVEMENT FOR THE PRELIMINARY DESIGN OF REINFORCED CONCRETE BUILDINGS [FISC89,90]	***Topological Model of the Building Structure:***			
	Level	Object class		
	height	- Attributes of Level	Numeric	* Input
	elevation	-	Numeric	* Input
	Structural Elements:	Object class. Superclass of Columns, Walls, Beams, Slabs, Drop Caps, Doors, Windows, Foundations. The latter are object classes that correspond to the ones defined in CIFECAD. Their attributes are presented here according to the knowledge types identified in the system.		• Used to map into constructibility knowledge in the knowledge base in order to define applicable construction methods and to determine constructibility
	General Data:			
	ID (identifier)	- Attributes of the object classes for structural element	Numeric	
	Layout Data:			* These layout data are input data.
	coordinates		Numeric	
	orientation		Numeric	
	(floor) level		Numeric	{ defined in terms of a PHI angle} Used for columns to determine exterior or interior location
	connected members		Pointer to other objects	
	span		Numeric	- { Only applicable to beams }
	clear span	-	Numeric	-

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
CONSTRUCTIBILITY IMPROVEMENT FOR THE PRELIMINARY DESIGN OF REINFORCED CONCRETE BUILDINGS [FISC89,90]	*Dimensioning Data:* member dimensions (length, width, height, thickness) *Detail Data:* concrete strength rebar percentage ***Constructibility Knowledge:*** Application Heuristics Layout Knowledge Dimensioning Knowledge Detailing Knowledge Exogeneous Knowledge	Attributes of the object classes for structural members - This knowledge is incorporated in object classes which represent construction methods of Beams, Columns, Slabs, Steel Forms, Slip Forms, Flying Forms, STEELdomes, FIBERGLASSdomes, etc. For more detail, refer to Appendix 3 of [FISC90]	Numeric Numeric Numeric	* Input * Input * Input • Used in determining the constructibility of the input structural design objects drawn from CIFECAD

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
DEVELOPMENT OF CIFECAD SYSTEM (VERSION 2.0) [KOLO90]	***General Project Data:***	In general, CIFECAD object attributes are specified and controlled by the user. CIFECAD offers default values which can be changed by the user.		* All data shown here is input data provided by the user. CIFECAD is a small high-level CAD system that provides commands to draw building structural design objects. As output, CIFECAD provides a facility to convert/write these data into an ASCII text file for use in other applications such as CoKE [FISC89].
	project name		Symbolic	
	project client		Symbolic	
	project use		Symbolic	
	project designer		Symbolic	
	project duration		Symbolic	
	project budget		Symbolic	
	General Building Data:			{ General data about the building structure drawn in CIFECAD }
	structural systems description		Symbolic	
	number of floors		Numeric	
	Floors:			
	height		Numeric	
	elevation		Numeric	
	floor number		Numeric	
	Structural Elements:			
	Common Attributes of structural element objects:			
	total number of elements		Numeric	
	element ID number		Numeric	
	coordinates (X, Y, Z)		Numeric	
	material designation		Numeric	

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
DEVELOPMENT OF CIFECAD SYSTEM (VERSION 2.0) [KOLO90]	*Common Attributes of Columns, Beams, Walls, Slabs:*			
	reinforcement designation		Numeric	
	elevation		Numeric	
	floor level		Numeric	
	orientation PHI angle		Numeric	{ Not applicable to Slabs }
	Columns	Object implemented as a block in AUTOCAD		
	length (or DX)	- Attributes of Columns	Numeric	
	width (or DY)		Numeric	
	height		Numeric	
	column-below designation		Numeric	
	round shape	-	Boolean	
	Beams	Object implemented as a block in AUTOCAD		
	length		Numeric	
	top width	- Attributes of Beams	Numeric	
	bottom width		Numeric	
	height		Numeric	
	left column	-	Numeric	
	Walls	Object implemented as a block in AUTOCAD		
	length	- Attributes of Walls	Numeric	
	width		Numeric	
	height	-	Numeric	

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
DEVELOPMENT OF CIFECAD SYSTEM (VERSION 2.0) [KOLO90]	left object		- Pointer to another object	
	right object		-	
	Slabs	Object implemented as a block in AUTOCAD		
	area	- Attributes of Slabs	Numeric	
	uniform thickness		Numeric	
	boundary columns		- Pointer(s) to other objects	
	supporting columns	-	-	
	Drop Caps	Object implemented as a block in AUTOCAD		
	floor level	- Attributes of Drop Caps	Numeric	
	thickness		Numeric	
	type designation		Numeric	
	length		Numeric	
	width	-	Numeric	
	Doors	Object implemented as a block in AUTOCAD		
	floor level	- Attributes of Doors	Numeric	
	width		Numeric	
	height		Numeric	
	wall designation	-	Numeric	

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
DEVELOPMENT OF CIFECAD SYSTEM (VERSION 2.0) [KOLO90]	Windows	Object implemented as a block in AUTOCAD		
	same attributes as Doors	- Attributes of Windows -		
	elevation		Numeric	
	Foundations:			
	Common Attributes of different foundation types:			
	length		Numeric	
	width		Numeric	
	thickness		Numeric	
	Single Footings	Object implemented as a block in AUTOCAD		
	column designation	- Attribute of Single Footings -	Numeric	
	Strips	Object implemented as a block in AUTOCAD		
	offsets (left, right, front, rear)	- Attributes of Strips	Numeric	
	support objects	-	Pointer(s) to other objects	
	Mats	Object implemented as a block in AUTOCAD		
	vertices	- Attributes of Mats -	Numeric	
	support objects		Pointer(s) to other objects	

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
AN OBJECT MODEL FOR INTEGRATED PROJECT MANAGEMENT [FRO90c]	***General System Libraries:***			* This is a model. Identification of input and output is not appropriate in this case.
	Resource Types	Object class		
	Labor Types	- Subclasses of Resource Types		• Objects in these libraries encapsulate common attributes of general project management based on the notion of "types" (similar to typical entities). The different types are for labor, equipment, material, crew and components.
	Equipment Types			
	Material Types			
	Crew Types	-		
	laborers	Attributes of Crew Types	List of Labor Types and number of laborers	• The uses of these general object libraries include work pricing, cost forecasting, job accounting, resource leveling, etc.
	equipment		List of Equipment Types and number of equipments	
	size parameters	-	Numeric	
	Component Types	Object class		• To represent types of physical project component.
	parts	-	List of subpart Component Types	
	part of	Attributes of Component Types	Slot for another Component Types	
	size parameters		Numeric	
	materials		Slot for Material Types and material quantities	
	function		Symbolic	
	work types		List of Work Types	{ Type of function such as column, beam, etc. }
	co-requirements	-	Slots for other objects	For identifying other components that must co-exist with these components

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
AN OBJECT MODEL FOR INTEGRATED PROJECT MANAGEMENT [FRO90c]	Actions	Object class		• To represent construction operations performed on a component (e.g., install, form))
	Methods	Object class		• To represent work methods
	component types	- Attributes of Methods	List of Component Types	
	actions		List of Actions	
	crew type		List of Crew Types	
	material types		List of Material Types	
	productivity	- 	Numeric	{ Productivity rating. Function of the component and crew size parameters}
	Work Types			
	component type	Object class	Slot for Component Types	• To represent standard types of work which identify the action, component and method pertinent to the work.
	action		Slot for Actions	
	method		Slot for Methods	
	Project Elements:			
	Components			• To represent specific project components or component groups
	component type	Object class	Slot for Component Types	
	quantity	- Attributes of Components	Numeric Numeric	
	location		Slot for Work Items	{ Location of the component in terms of coordinate data }
	work items	-		

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
AN OBJECT MODEL FOR INTEGRATED PROJECT MANAGEMENT [FRO90c]	Resources	Object class		• To represent specific resources used on a project
	resource type	-	Slot for Resource Types	
		Attributes of Resources		
	uses	-	Slot for Resource Uses	
	Work Items	Object class		• To represent items of work on a project
	work type	-	Slot for Work Types	
	components	Attributes of Work Items	Slot for Components	
	resource uses	-	Slot for Resource Use	
	Resource Uses	Object class		• To represent the use of specific resources for specific work items
	work item	-	Slot for Work Items	
	resource type	Attributes of Resource Uses		
	resource	-		
	For Applications:	*Include objects that are directly relevant to different project management applications. The latter include cost/ schedule control, financial forecast, accounting, etc.*		
	**** Pricing Database:***			
	Work Type Unit Prices	Object class		• To encapsulate cost / productivity data and methods associated with a specific work type
	work type	-	Slot for Work Types	
	historic work item prices	Attributes of Work Type Unit Prices -	List of Work Item Prices	

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
AN OBJECT MODEL FOR INTEGRATED PROJECT MANAGEMENT [FRO90c]	Work Item •Prices	Object class		• To encapsulate cost / productivity data and methods associated with a specific work item
	work item	Attribute	Slot for Work Items	
	Resource Type Unit Prices	Object class		• To encapsulate cost data and methods associated with a specific resource type
	resource type	- Attributes of Resource Type Unit Prices	Slot for Resource Types	
	historic resource item prices	-	List of Resource Item Prices	
	Resource Item Prices	Object class		• To encapsulate cost data and methods associated with a specific resource item
	resource item	Attribute	Slot for Resource Items	
	**** Estimating:***			
	Estimates	Object class		• To represent calculations of the total expect cost of the project
	estimate items	Attribute	Slot for Estimate Items	
	Estimate Items	Object class		• To represent calculations of the expected cost of an individual work item
	work item	-	Slot for Work Items	
	work item price	Attributes of Estimate Items	Slot for Work Item Prices	
	unit prices		Numeric	
	total cost	-	Numeric	

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
AN OBJECT MODEL FOR INTEGRATED PROJECT MANAGEMENT [FRO90c]	* ***Schedule:***			
	Schedules	Object class		• To represent calculations of time frames expected for a project
	activities	- Attributes of Schedules	Slot for Activities	
	calendar	-	Slot for Calendar	
	Activities	Object class		• To represent segments of work in an expected time frame
	predecessor constraints	-	List of logic constraints	
	successor constraints	Attributes of Activities	List of logic constraints	
	calendar		Slot for Calendar	
	work item	-	Slot for Work Items	

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
DATA BASE SCHEME FOR BUILDING DESIGN [CHEN90] *(Part of the PENGUIN project [LAW89])*	**Buildings**	Relation	As of 11/90, data format and integrity constraints in the subsequent relations are not defined.	• Relation for a building * This is a data base schema. Identification of input and output is not appropriate in this case.
	building_id	- Attributes of the relation Buildings		
	building_name			
	building_type			
	building_function			
	building_system			
	building_location	-		
	Structural Systems:			
	Super_Structures	Relation		• Relation for a "super structure" or major structural system in a building ?
	building_id	- Attributes of the relation Super_Structures		
	ss_id			{ Super-structure of the story }
	cs_name			{ Coordinate system of this superstructure }
	storey_num			
	basement_storey_num			
	penthouse_storey_num			
	height			
	structure_type	-		
	Storeys	Relation		• Relation for a story which consists of vertical and horizontal elements
	building_id	- Attributes of the relation Storeys		
	ss_id			{ Super-structure of the story }
	storey_no			
	storey_type	-		

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
DATA BASE SCHEME FOR BUILDING DESIGN [CHEN90] *(Part of the PENGUIN project [LAW89])*	Frames frame_id bay_num storey_num *+ There are other relations for structural systems such as* Floor_Systems, Floors, Slabs, etc.	Relation - Attributes of the relation Frames -	As of 11/90, the data format in the subsequent relations are not defined.	• Relation for a frame { Number of bays } { Maximum number of stories }

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
DATA BASE SCHEME FOR BUILDING DESIGN [CHEN90] *(Part of the PENGUIN project [LAW89])*	***Structural Members:***		As of 11/90, the data format in the subsequent relations are not defined.	
	Columns	Relation		• Relation for a structural column member
	building_id	-		
	ss_id	Attributes of the relation Columns		{ Super-structure of the column }
	storey_no			
	column_no			
	x_gridline#			{ Gridline in the x direction }
	y_gridline#			{ Gridline in the y direction }
	node1#			{ Node 1 of the column }
	node2#			{ Node 2 of the column }
	x_frame#			{ Frame in the x direction }
	y_frame#			{ Frame in the y direction }
	x_size			{ Width in the x direction }
	y_size			{ Width in the y direction }
	x_offset			{ Offset to the gridline in the x direction }
	y_offset			{ Offset to the gridline in the x direction }
	col_height	-		

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
DATA BASE SCHEME FOR BUILDING DESIGN [CHEN90] *(Part of the PENGUIN project [LAW89])*	*+ There are other relations for structural members such as* Girders, Secondary_Beams, Walls, ShearWalls, Boundary_ShearWalls, Plain_ShearWalls, etc.			
	Reinforced concrete structural members:			
	RC_Columns	Relation		• Relation for a reinforced concrete column member
	building_id	-		
	ss_id	Attributes of relation RC_Columns		{ Super structure of the column }
	storey_no			
	column_no			
	RC_column_list_name	-		{ Set of similar columns from bottom to top ? }
	+ There are other relations for reinforced concrete structural members such as RC_Girders, RC_Secondary_Beams, etc.			
	Shape of structural Members:			
	Wall_Shapes	Relation		• Relation for the shape of a wall
	shape_id	- Attributes of relation Wall_Shapes		
	vertex_node_no	-		{ Vertex node number of the wall shape ? }

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
DATA BASE SCHEME FOR BUILDING DESIGN [CHEN90] *(Part of the PENGUIN project [LAW89])*	*+ There are other relations for structural member shapes such as* Slab_Shapes ***Cross-section Properties of structural members:***		As of 11/90, data format and integrity constraints in the subsequent relations are not defined.	
	RC_Column_Sections	Relation		• Relation for the cross section of a reinforced concrete column
	list_name	-		{ ? }
	story_no	Attributes of relation RC_Column_Sections		
	list_id			{ ? }
	section_designation	-		{ Designation of the cross section properties }
	RC_Column_Section_Designations	Relation		• Relation for the cross section properties of a reinforced concrete column
	section_designation	-		
	Dx	Attributes of relation RC_Column_Section_Designations		{? }
	Dy			{? }
	gross_area			
	shear_area			
	lx	-		{ Unbraced length in the x direction }
	ly			{ Unbraced length in the y direction }

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
DATA BASE SCHEME FOR BUILDING DESIGN [CHEN90] *(Part of the PENGUIN project [LAW89])*	flexure_steel_area shear_steel_area x_rebar_number x_rebar_dia y_rebar_num y_rebar_dia shear_rebar_dia shear_spacing *+ There are other relations for structural member cross sections such as* RC_Beams_Sections, ShearWall_Sections, SHW_Designations etc.	Attributes of relation RC_Column_Section_Designations -	As of 11/90, data format and integrity constraints in the subsequent relations are not defined.	
	Material: RC_Materials material_designation concrete_grade_no rebar1_grade rebar2_grade rebar1_dia rebar2_dia	Relation - Attributes of relation RC_Materials -		• Relation for reinforced concrete material

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
DATA BASE SCHEME FOR BUILDING DESIGN [CHEN90] *(Part of the PENGUIN project [LAW89])*	***Supports of structural members:***		As of 11/90, data format and integrity constraints in the subsequent relations are not defined.	
	Girder_Supports	Relation		• Relation for the members that support a girder
	building_id	-		
	ss_id	Attributes of relation Girder_Supports		
	storey_no			
	girder_id			{ Girder being supported ? }
	support_member_id	-		{ Members that support the girder ? }
	+ There are other relations for structural member supports such as SLAB_SUPPORTS, SLAB_SB_ SUPPORTS, etc.			
	Connections of structural members:			
	Wall_Column_Connections	Relation		• Relation for the connection between a wall and column
	building_id	-		
	ss_id	Attributes of relation WALL_COLUMN_CONNECTIONS		
	storey_no			
	wall_id			
	column_id	-		

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
DATA BASE SCHEME FOR BUILDING DESIGN [CHEN90] *(Part of the PENGUIN project [LAW89])*	*+ There are other relations for support and connections such as* Wall_Wall_Connections, etc. *Architectural Entities:* Floor_Plans floor_plan_id floor_plan_name scale designer *+ There are other relations for architectural entities such as* Wall_Openings, Doors, Windows, etc. *+ There are other relations for entities such as* Coordinate_Systems, Gridlines, Nodes, etc.	 Relation - Attributes of relation Floor_Plans -	As of 11/90, data format and integrity constraints in the subsequent relations are not defined.	 • Relation for an architectural floor plan

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
APPLYING DESIGN-DEPENDENT KNOWLEDGE IN STRUCTURAL ENGINEERING DESIGN [WANG88, WANG90]	**Data Items** Data Items has several subclasses: Condition, Rule, Mapping, User.Query, Function and Rule.Set. *Each subclass of Data Items represents a self-contained data item type. It has a value, type information and methods for determining its value.*	Object class *This implementation follows Garrett's object-oriented design standard model [GARR89].*		• To represent all logical expressions and variables.
	Design Solution Objects: *Physical Objects:* Column Beam etc.	*These objects are based on the Structural Steel Framing Data Model [LAVA89].*		• To represent the physical design objects. Beam, Column and Beam-Column are the objects used in the structural steel beam-column design in the early prototype. * For the current application domain of beam-column design, the inputs to the system are the length L of the member, unbraced length factor K, loading (P and M at both end) and material designation. The output is the designed member section.
	Variable Objects	Instances of object class Data Items		• To represent variables used in design calculation and constraint checking
	Constraint Objects	Instances of object class Condition		• To represent design requirements such as strength constraints, geometric constraints, etc.

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
APPLYING DESIGN-DEPENDENT KNOWLEDGE IN STRUCTURAL ENGINEERING DESIGN [WANG88, WANG90]	***Design Control Objects:***			
	Plan	Object Class		• To represent sequences of design goals to be followed and the intention of the plan
	goal.list	-	List of Goal objects	
	intention	Attributes of Plan	LISP expression	
	status		Symbolic	
	weight		Numeric [1,10]	
	originated. case		Pointer to a design Case object	
	active.goal		Pointer to a Goal object	
	remaining.goal.list	-	List of Goal objects	
	Goal	Object Class		• To represent parts of a design plan or individual design considerations
	function	-	LISP expression	
	intention	Attributes of Goal	LISP expression	
	status		Symbolic	
	weight		Numeric [1,10]	
	include.in	-	Pointer to a Plan object	
	Retrieved Design	Object Class		• To represent objects which provide a previous design solution or plan that is retrieved for reusability based on a similarity rating
	ase.Name	-	Pointer to a Case object	
	Solution.Similarity	Attributes of Retrieved Design	Numeric [1,100]	
	Plan.Similarity	-	Numeric [1,100]	

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
APPLYING DESIGN-DEPENDENT KNOWLEDGE IN STRUCTURAL ENGINEERING DESIGN [WANG88, WANG90]	***Design Action Objects:***			
	Design.Action	Object Class with no attribute		• To represent a design action.
	Knowledge Source	Subclass of Design.Action		• To represent domain or control knowledge sources. A knowledge source is considered here as an action object which contains knowledge about when it is applicable, how its bindings are set, what action to perform, and other declarative information.
	trigger.condition	- Attributes of Knowledge Source	LISP predicate	
	context.variables		Nested list of pairs. Each pair contains a variable and an expression that returns a value.	
	name.generator		List to be generated into name of KSAR	
	precondition		LISP predicate	
	action.code		LISP lambda expression	
	description		Symbolic	
	priority		Numeric or LISP expression that returns a number	
	originated.case		Pointer to a Retrieved Design object	
	status	-	Symbolic	
	Knowledge Source Activation Record (KSAR)	Subclass of Design.Action		• To represent specified action objects that perform blackboard modification tasks. Each object is created from a triggered knowledge source and has all context variables bound.
	Attributes carried over from the originating knowledge source:			
	precondition	- Attributes of KSAR	LISP predicate	

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
APPLYING DESIGN-DEPENDENT KNOWLEDGE IN STRUCTURAL ENGINEERING DESIGN [WANG88, WANG90]	action.code	Attributes of Knowledge Source Activation Record	LISP lambda expression	
	description		Symbolic	
	priority		Numeric or LISP expression that returns a number	
	originated.case		Pointer to a Retrieved Design object	
	New attributes:			
	variable.bindings		Nested List of pairs, similar to the attribute Context.Variables in class Knowledge Source. The difference is that each pair consists of a variable and a returned value.	
	rating		Numeric	
	KS		Pointer to a Knowledge Source object	
		-		
	Design Case *This object class includes a number of attributes such as* Case.Name, Description, Problem.Input, Design.Plan, Redesign.Plan, Solution, *etc. These attributes describe information about a previous design case in Case Memory.*	Object class		• To represent a previous design case stored in the Case Memory. It encapsulates data about the design case's name, description, problem input, design plan, redesign plan, solution, etc.

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
OBJECT-ACTION-RESOURCE PLANNING SYSTEM (OARPLAN) [DARW89, HOWA89b] *(Part of the research on Knowledge-based planning of facility projects [CIFE90])*	***Objects***:			
	Object	Object class		• Used as the root node of the generalization hierarchy of object classes
	Building	Subclass of Object		• To represent buildings
	rep-descrip	- Attributes of Building	Pointer to Representational-Description	
	gen-descrip	-	Pointer to Gen-Description	
	Representational- Description	Subclass of Object		• Used to reference all objects that describe or compose the building [LAW86]
	rep-descrip-of	- Attributes of Representational-Description	Pointer to Building	
	has-view	-	Pointer(s) to View	
	Gen-Description	Subclass of Object		• Used to describe general information about the building [LAW86]
	gen-desription-of	-	Pointer to Building	
	owner	Attributes of Gen-Description	Symbolic	* Input
	location		Symbolic	* Input
	name	-	Symbolic	* Input
	View	Subclass of Object		• To represent particular views of building data
	view-of	- Attributes of View	Pointer to Representational-Description	
	has-part	-	Pointer(s) to other objects	
	Structural-View	Subclass of View	The attribute has-part is the slot for pointers to structural objects.	• To represent the structural view of building data

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
OBJECT-ACTION-RESOURCE PLANNING SYSTEM (OARPLAN) [DARW89, HOWA89b] *(Part of the research on Knowledge-based planning of facility projects [CIFE90])*	Architectural-View	Subclass of View	The attribute has-part is the slot for pointers to architectural and topological objects respectively.	• To represent the architectural view of building data
	Topological-View	Subclass of View		• To represent the topological view of building data
	Architectural Objects:			* The attribute values in these objects require input data.
	Floor	Subclass of Object		• To represent floors in the building
	element-object	-	Boolean (Yes, No)	
	ceiling-height	Attributes of Floor	Numeric	
	floor-height		Numeric	
	has-part	-	Pointer(s) to floor space objects (e.g., Room, Wall)	
	Space-Floor	Subclass of Object		• To represent spaces occupied by floors in the building ?
	part-of	- Attributes of Space-Floor	Pointer to Floor	
	element-object	-	Boolean (Yes, No)	
	Space-Ceiling	Subclass of Object		• To represent spaces (space functions) within the floors of the building such as rooms, corridors, stairways, elevators,etc.
	part-of	- Attributes of Space-Ceiling	Pointer to Floor	
	element-object	-	Boolean (Yes, No)	
	Floor-Space	Subclass of Object		• To represent spaces occupied by ceilings in the building ?
	has-bounding-wall	- Attributes of Floor-Space	Pointer to Wall	
	element-object		Boolean (Yes, No)	
	zones	-	Pointer to Floor	
	Room, Corridor	- Subclasses of Floor-Space		

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
OBJECT-ACTION-RESOURCE PLANNING SYSTEM (OARPLAN) [DARW89, HOWA89b] *(Part of the research on Knowledge-based planning of facility projects [CIFE90])*	Elevator	Subclass of Floor-Space		
	Stairway	Subclass of Floor-Space -		
	flight-width	Attributes of Stairway	Numeric	
	no-steps		Numeric	
	stair-depth	-	Numeric	
	floor-distance		Numeric	
	Wall	Subclass of Object		
	part-of	-	Pointer to Space-Floor ?	
	bounding-wall-of	Attributes of Wall	Pointer to Floor-Space or its subclasses	
	adjacent-to		Pointer(s) to Slab	
	supported-by		Pointer(s) to Column	
	supports		Pointer to Floor ?	
	zones		Pointer to Floor	
	element-object	-	Boolean (Yes, No)	
	Exterior-Wall	- Subclasses of Wall		
	Interior-Wall	-		
	Opening	Subclass of Object		
	element object	- Attribute of Opening -	Boolean (Yes, No)	

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
OBJECT-ACTION-RESOURCE PLANNING SYSTEM (OARPLAN) [DARW89, HOWA89b] *(Part of the research on Knowledge-based planning of facility projects [CIFE90])*	Door	Subclass of Opening		
	part-of	- Attribute of Door -	Pointer to Wall ?	
	Window	Subclass of Opening		
	part-of	- Attribute of Window -	Pointer to Wall ?	
	Structural Objects:			* The attribute values in these objects require input data.
	Structural-Member	Subclass of Object		
	part-of	- Attributes of Structural-Member	Pointer to Structural-View	
	supported-by		Pointer to other structural member objects	
	supports		Pointer to other structural member objects	
	adjacent-to		Pointer to other structural member objects	
	element-object		Boolean (Yes, No)	
	zones	-	Pointer to Floor	
	Column	Subclass of Structural-Member		
	Rectangular-Column	- Subclasses of Columns		
	Circular-Column	-		

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
OBJECT-ACTION-RESOURCE PLANNING SYSTEM (OARPLAN) [DARW89, HOWA89b] *(Part of the research on Knowledge-based planning of facility projects [CIFE90])*	Beam	Subclass of Structural-Member -		
	bounding- member-of	Attribute of Beam -	Pointer to Slab	
	Rectangular-Beam	- Subclasses of Beam		
	T-Beam	-		
	Girder	Subclass of Beam		
	Rectangular-Girder	- Subclasses of Girder		
	T-Girder	-		
	Bracing-Member	- Subclasses of Structural-Member		
	Foundation	-		
	Slab	Subclass of Structural-Member		
	has-bounding-member	- Attribute of Slab -	Pointer(s) to Beam	
	One-Way-Slab	- Subclasses of Slab		
	Two-Way-Slab	-		
	Topological Objects:	Not implemented in the current version		* The attribute values in these objects require input data.
	3D-Space			

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
OBJECT-ACTION-RESOURCE PLANNING SYSTEM (OARPLAN) [DARW89, HOWA89b] *(Part of the research on Knowledge-based planning of facility projects [CIFE90])*	*Other Objects:*			
	Specs	Subclass of Object		• To encapsulate data about material, dimensions and other attributes of design objects (e.g. Wall, Door, Window)
	specs-of		Pointer to objects that own this specification (e.g. Wall, Door, Window)	
	Wall-Specification	Subclass of Specs		• To encapsulate data about material, dimensions and other attributes of walls
	finish		Symbolic	
	material		Symbolic	
	thickness		Numeric	
	Door-Specification	Subclass of Specs		• To encapsulate data about material and dimensions of doors
	material		Symbolic	
	thickness		Numeric	
	width		Numeric	
	length		Numeric	
	Window Specification	Subclass of Specs		• To encapsulate data about material and dimensions of windows
	thickness		Numeric	
	width		Numeric	
	length		Numeric	
	Resource	Not implemented in the current version		

DATA SOURCE	DATA ITEM	DATA STRUCTURE	DATA TYPE	USE
OBJECT-ACTION-RESOURCE PLANNING SYSTEM (OARPLAN) [DARW89, HOWA89b] *(Part of the research on Knowledge-based planning of facility projects [CIFE90])*	**Action**	Subclass of Object		* These objects are output of the system. • To represent a construction action such as <construct>, <weld>, <paint>, etc.
	superaction	-	- Pointers to other Action objects	
	subaction	Attributes of Action		
	after			
	before	-	-	
	Activity	Subclass of Object		* These objects are output of the system. • To represent a construction activity such as <weld> <pipe> <welder>. An activity consists of a tuple of <action> <object> <resources>. Resources are not implemented in the current version.
	involved-action	- Attributes of Action	Pointer to Action	
	involved-object	-	Pointer to Object	