Framework for Component-Based Structural Engineering Software

Naveed Anwar[1], Worsak Kanok-Nukulchai[2]

KEYWORDS

Data Communication, Structural Engineering, Structural Models, Information Management, Computer Application

ABSTRACT

Component-based software development has gained wide acceptance within the computer science discipline and the software development industry. Apparently, application of this concept has not been attempted for the development of software for structural engineering applications. This paper presents the overall framework for component-based software development for integrated structural engineering applications using the principles of information transformation. An information-oriented review of the overall structural design process is carried out to develop the information content and transformations involved in this process. This information transformation concept is then used to propose a framework for the development of software for integrated structural design applications including the identification of key packages, patterns, components, class hierarchies and object models. Use of XML is proposed as the main information description and communication standard. The advantages of using the proposed framework are discussed along with application and implementation scenarios.

INTRODUCTION

Component Based Software Development (CBSD) is a logical extension of object-oriented concepts and the corresponding object technologies, which have pervaded in the software industry since the early 1990s. The CBSD has also made it possible for parts of the same software to be developed by different organizations, and to assemble new applications from existing, specialized as well as generalized components. In such a scenario, the component-based architecture can significantly formalize, streamline, expedite, and enhance the software development process for structural design applications as well.

INFORMATION VIEW OF STRUCTURAL DESIGN PROCESS

Traditionally, the structural design process has been described and viewed in terms of distinct design steps, such as conception, modeling, analysis, design, detailing, drafting, and costing, (Kanok-Nukulchai 1986) and (Anwar 1994). Consequently, computer programs and software were also traditionally developed to address one or two design steps; for example, analysis programs, design programs, and detailing programs. However, much research is currently focused on the development of integrated systems capable of handling several of the design steps in the overall design process using various forms of database and data flow models.

[1] Associate Director, Asian Center for Engineering Computations and Software (ACECOMS), Asian Institute of Technology (AIT), Bangkok, Thailand
[2] Dean, School of Engineering and Technology, Asian Institute of Technology (AIT), Bangkok, Thailand

Structural Design Information Space

The structural design process is part of the overall planning and design process that leads to construction, use and maintenance of a construction projects. Information used in the structural design process is therefore part of the overall construction information space. Therefore, the information needs to be shared across discipline boundaries. Several attempts (Grubbs, Leach, and Law 1988), (Przybylo and Mokrzycki 1989), (Nguyen and Oloufa 2001) and (Pena-Mora and Choudary 2001) have been made to represent civil engineering and building project databases in a universal format that can be used across several disciplines, with varying levels of success and acceptance by the industry.

Information Processing Packages and Information Flow

Data processing has progressively given way to information processing, which is now giving way to knowledge processing, and ultimately to the processing of concepts and ideas. These new developments in the information technologies can be readily applied to the development of an information model for the structural design process.

The information space and information flow described in the proceeding section is used here as the starting point for developing the architecture of a typical integrated structural engineering application. Each information-processing block, or design process step in this case can be viewed as a "package". The following main information processing packages have been identified:

a) The Conceptual and Preliminary Design

b) The Structural Modeling and Analysis

c) The Structural Design

d) The Structural Detailing and Drafting

It is proposed here, that rather than a sequential information flow, the concept of information bus be adopted for this model (Figure 1). In this concept, each information block or package, accesses and updates the overall information space as needed.

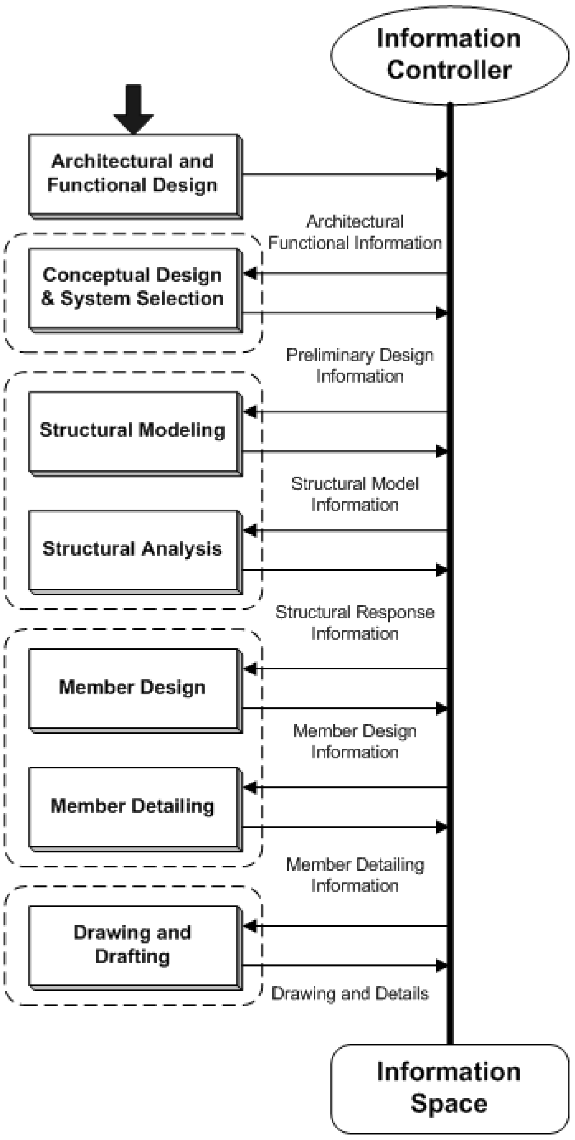
Information Controller
Architectural and Functional Design
Architectural Functional Information
Conceptual Design & System Selection
Preliminary Design Information
Structural Modeling
Structural Model Information
Structural Analysis
Structural Response Information
Member Design
Member Design Information
Member Detailing
Member Detailing Information
Drawing and Drafting
Drawing and Details
Information Space

Figure 1: Information flow model for structural design process.

The information flow is therefore considered to occur on an information bus passing outside of the packages so that it becomes independent of the individual package processes or any predetermined information flow sequence. Each package then takes the information it needs from the information space through this bus and returns the processed information in the same way. In this architecture, the entire structural design information space is assumed to exist, irrespective of the existence of information packages or their functionality or operation.

INFORMATION ORIENTED COMPONENT BASED FRAMEWORK

By definition, Frameworks "… provide a skeletal design that can be built upon to create an organized system where many packages or components work together" (Stevens and Pooley 2000). The basis for such a framework in the development of structural design applications is laid out in Figure 1, where the overall information space of a construction project and the role of structural design are defined. A conceptual information package architecture is shown in Figure 2, with the proposed information flow mechanism.

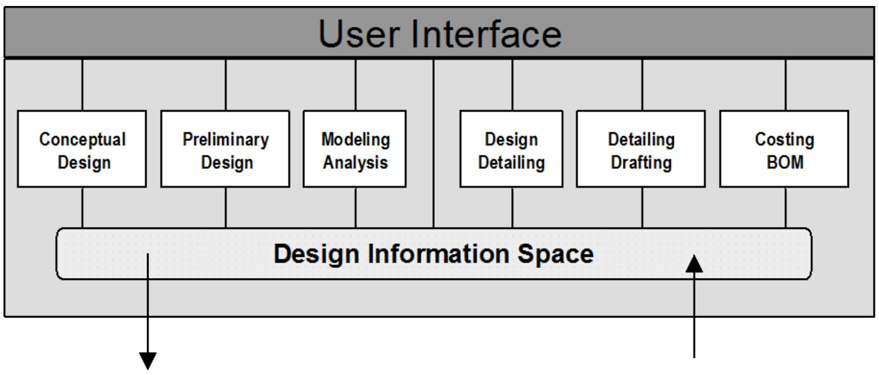

Figure 2: Conceptual structural design information package architecture

Overall Application Architecture and Frameworks

The overall software architecture for an integrated structural design based on the information transformation concepts can be described in terms of three basic layers:

- Layer-1: The application made up of one or more packages. Each package handles one major information processing task in the overall design process. This layer also handles the activation and marshaling of packages and acts as the main controller;

- Layer-2: Packages made up of one or more integrated components. Each component provides specific services or handles dedicated tasks within the package; and

- Layer-3: The components made up of one or more objects. Each object provides specific services or handles specific information processing tasks.

The frameworks required to govern the design and development of actual applications conforming to the above architecture will now be defined. These include:

a) A general framework for the whole software that defines the purpose of the software; the tasks to be assigned to each package; and the way in which they will be located and used;

b) A package framework for each package that defines the tasks to be assigned to each component and the way in which they will be located and used;

c) Component frameworks, which are at a lower level and govern the design of the components themselves, define the behavior of the components as well their overall design in terms of objects, classes, services, messages, etc.

Application Framework

In general, the top level role of the application is to define the bounds of structural design information space; establish the extent of information processing to be performed by the application; establish ways and means of obtaining the information needed from other information sources, identify and integrate packages to complete the contents of the required information space; and finally, update and communicate with the outgoing information sources. The overall application framework pattern is shown in Figure 2. Two types of basic frameworks are proposed:

1. A Loosely Integrated Application: In this framework, each package is completely self-contained and all required components are compiled into the package.

2. Tightly Integrated Application (Cohesive Framework): In this framework, all general-purpose components needed by all packages are physically contained within the main application.

Package Frameworks

The specific design and functionality of the individual packages identified earlier will vary significantly depending on the type of structure and level of integration of the packages. However, the common framework and pattern for these packages can be defined in terms of the basic role they play in the overall application. Each package is basically an information processor. Upon activation, it takes the required information from the information space using the information bus, converts it to appropriate context, processes this information using its components and produces new information or updates the initial information, converts it back to the global context, and returns it to the main information space. This framework pattern is shown in Figure 3. Specific frameworks and patterns for some selected package are developed next, identifying main components and their usage within the package.

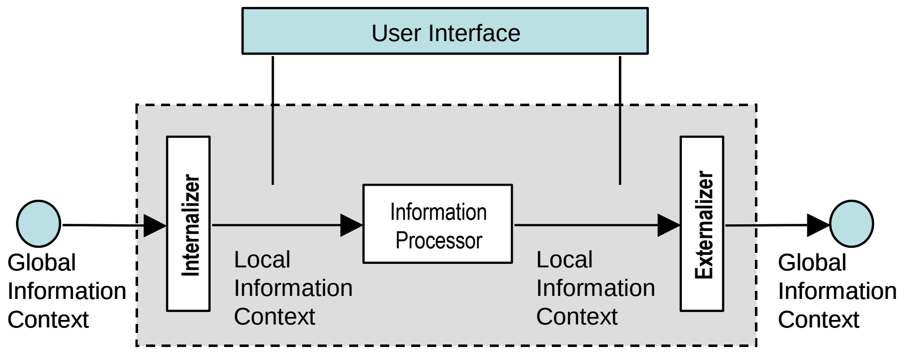
User Interface
Global Information Context
Internalizer
Local Information Context
Information Processor
Local Information Context
Externalizer
Global Information Context

Figure 3: Typical package framework

The Conceptual and Preliminary Design Package

The conceptual and preliminary design package is primarily concerned with the development of conceptual design, selection of appropriate structural system, and carrying out of preliminary sizing and location of the structural components. This package should be capable of creating the basic structural model in terms of defining key structural systems, elements, their location, and their proportions. How this package works is not of concern at this stage of software system definition. The important decisions to be taken are related to what information will be needed (or consumed) by this package and what information will be generated. The information generated by the package will be a fairly well defined structural model of the structural system along with some basic material specification.

The Modeling and Analysis Package

The modeling and analysis package is probably the most important package in the overall application. The main function of this package is to convert the structural model, which may be defined by the user directly or generated by the preliminary design package, and convert it into an analysis or mathematical model. The package will then analyze the model and produce the structural response information. The analysis model is most likely a Finite Element Model, although other models may also be generated. Most structural responses can ultimately be defined in terms of nodal displacements, element actions, and element deformations. The internal capabilities of this package will vary considerably from one application to the next. However, the basic function of the package is to convert the structural, geometric, and load information to structural response information. The minimum component framework needed to accomplish this task is shown in Figure 4. A modular framework for implementing the finite element method has been presented (Yu and Kumar 2001). This framework can be used to generalize and expand the modeling and analysis package framework described above.

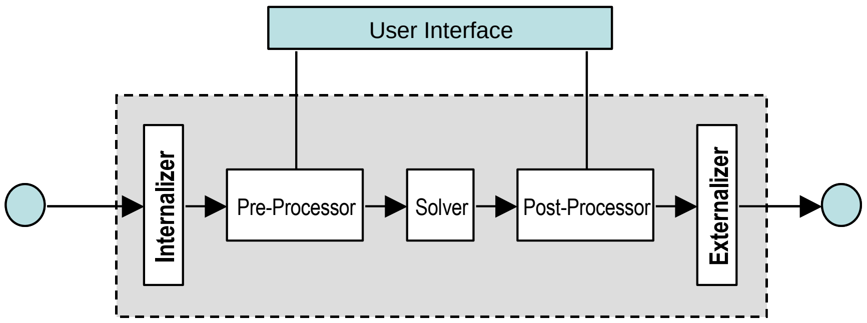

Figure 4: Modeling and analysis package

The Designing Package

The designing package is the second most important and complex package in the overall application. The main function of this package is to convert the structural response information to element/component design and detailing information. There is a certain amount of overlap between the design, detailing, and drafting activities. A fair amount of detailing information is generated during the member design process; also, a considerable amount of drafting information is generated during the detailing process. There is typically more overlap in the design and detailing of reinforced concrete structures than in that of steel structures. For these reasons, developing a general framework for a designing package is more difficult and complex than for an analysis package; however, the minimum component framework needed to accomplish this task is shown in Figure 5.

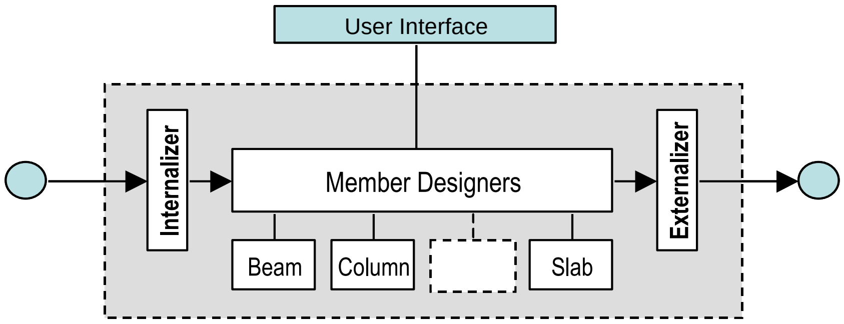

Figure 5: Designing package

Detailing and Drafting Package

The main functionality of the detailing-drafting package is to "elaborate" on the design-detailing information provided by the design-detailing package or by the user. This depends to a large extent on the type of structure, level of detailing needed, local practices and engineers' preferences. The functionality of this package will also depend on the level of detailing carried out by the design package. The output from this package is the detailed drawing, which is produced directly by the package and/or generated in a form readable by external Computer Aided Design (CAD) applications. The overall framework of the detailing package is shown in Figure 6.

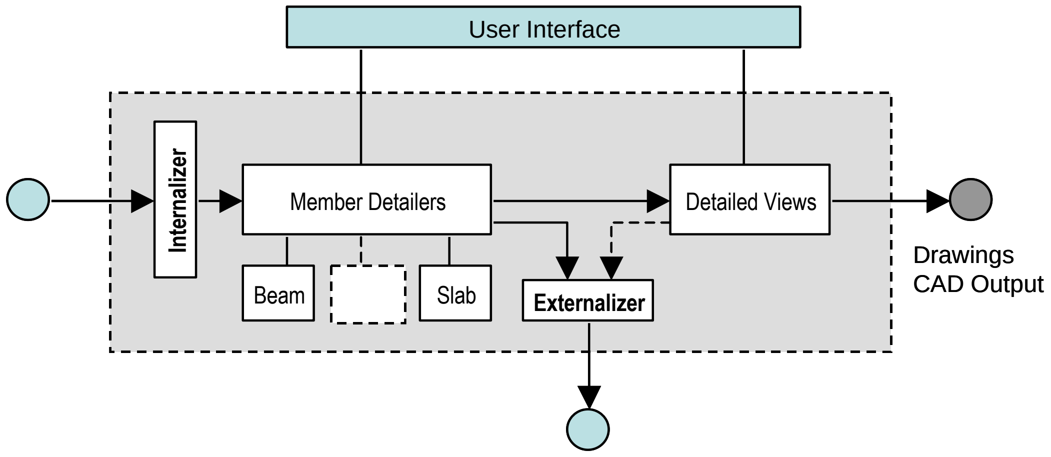

Figure 6: Detailing and drafting package

Communication Between Objects Components and Packages

For the component-based software to work, communication must be established at various levels amongst packages, components, objects, and external information sources. In fact, one of the key issues in the development and use of these component-based systems is finding a means of communicating and transferring information between various parts.

Figure 9 shows a graphical interpretation of this information communication process. All the information transfer or sharing outside the application can be done using XML or Databases by either the application itself or by the packages of the application, while components and objects share and transfer information between themselves, the packages and the application. The most generic and extensible information transfer standard being used these days in the computing industry is XML, whose advantages have been well documented and accepted. It is therefore proposed that XML be used as the standard backbone information communication channel as it can handle structured definition and communicate a large amount of structural model, analysis, design, and detailing information between packages and between components. This requires the definition of standard XML schemas to describe the information content and its structure, and the use of XML files based on these schemas containing actual information about specific projects.

SAMPLE IMPLEMENTATIONS

The concepts discussed in the proposed framework have been applied to develop software for integrated modeling analysis and design of concrete buildings, called BATS (Building Analysis of Three-Dimensional Systems) (BATS 2002). At the early stages of the development process of this software it was decided that existing software and programs within the author's Institute and Center would be used as much as possible and further development would be limited to functionalities not available in those programs. The main requirement of the new software was the ability to quickly generate building structural models using parametric templates and fully interactive graphics manipulation; analyze the model quickly for gravity and lateral loads; provide a visual interpretation of the analysis results, and; design concrete beams, columns and footings based on the results of analysis.

A review of the programs available within our organization indicated two candidates, described below, that could be used as components and packages in the overall framework:

1. XETAB, for the analysis of building models using the rigid diaphragm concept (Worsak 1988). This program was written in FORTRAN and was running under DOS. The source code of the program was not to be modified or included in the new software so this program had to be wrapped and used as a complete component.

2. GEAR, for the design of concrete members such as beams, columns, slabs, and footings (GEAR 2001). This program had no structural modeling or analysis capability, but fairly comprehensive design, and some detailing functionality. Each member's design was carried out by a separate component.

The final BATS application consists of two packages. The first package is for the modeling and analysis and the second for design and detailing (Figure 12). The modeling and analysis package was developed completely new; but, the XSolve was used as a component wrapper for XETABS which had all the analysis capability required of the new program. The design and detailing package was developed using object and component based concepts and written for the windows environment with good interactive and graphics capabilities. The data storage for this program was based on the 'tag-value' concept - similar to HTML and XML. No modification or redevelopment was required in this package and was integrated into the new software in its entirety.

This example shows that the component based software development framework was used efficiently to develop a new application quickly, partly by using new core functionality and partly by using existing components and packages.

CONCLUSIONS

An architectural Framework has been presented for the development of integrated structural engineering applications using component based software development with special emphasis to the information-processing context.

The proposed framework views the integrated design process in terms of various standard information processing packages. The packages themselves are assembled from information processing components. The components are developed using design patterns, encapsulating and exhibiting the generalized functionality. The concept of information bus is used to establish communication between packages and components. The use of XML is proposed as the generic information representation standard. Sample Schemas have been developed to represent structural design information in various information models. Three sample applications are presented that have been developed using the proposed framework, to demonstrate some of the implementation advantages.

It is envisaged that use of component based software development for structural engineering applications will pave the way for collaborative software development with greater specialization and re-use of standard problem solutions. Frameworks for addressing specific analysis and design aspects are already being developed and reported in literature. The overall framework proposed here will provide the ability to integrate such local frameworks into a larger integrated application. The development of XML based structural model information will provide a platform and application independent information communication standard. This aspect, which is part of the present research, is presented in a separate paper.

REFERENCES

Anwar, N. and Kanok-Nukulchai, W. (2005), "Component Based, Information Oriented Structural Engineering Applications" ASCE-Journal of Computing in Civil Engineering, pp. 45-57, January 2005.

Anwar, N. and Kanok-Nukulchai, W., (2001), "Paradigms, Tools and Techniques for Structural Engineering Software", The Eighth East Asia-Pacific Conference on Structural Engineering and Construction 5-7 December 2001, Nanyang Technological University, Singapore.

Anwar, N. and Kanok-Nukulchai, W.,(1996), "Application of Object Oriented Techniques in Structural Engineering Software", International Conference on Urban Engineering in Asian Cities in 21st Century, Proceedings Volume 1, School of Civil Engineering, Asian Institute of Technology, Bangkok, Thailand

Anwar, N., (1994). "Integrated Analysis, Design, Detailing and Costing of Concrete Buildings", Proceedings of 3rd Regional Conference on Computer Applications in Civil Engineering, Kuala Lumpur, Malaysia

Asian Center for Engineering Computations and Software, (2001), "ACECOMS GEAR User's Manual and Technical Reference" Asian Institute of Technology, Bangkok, Thailand

Asian Center for Engineering Computations and Software, (2002), "BATS 2001 Three-Dimensional Analysis and Design of Buildings User's Manual", Asian Institute of Technology, Bangkok, Thailand

Aster, M.,Bergmeister K., Rio O., Schonach A., Sparowitz L., (1998), "Iterative Object-Oriented Modeling For Structural Engineering". COMPUTATIONAL MECHANICS New Trends and Applications CIMNE, Barcelona, Spain

Crnkovic, I, (2002), "Component-based Software Engineering: Building Systems for Software Components" Proceedings-IEEE Computer Society's International Computer Software and Applications Conference, 2002.

Grubbs, J.H., Leach, L.M. and Law ,K.H. (1988), "Data Exchange for Collaborating Structural Design Programs", ASCE

H. Adeli and G. Yu, (1995), "An Integrated Computing Environment for Solution of Complex Engineering Problems Using the Object-Oriented Programming Paradigm and A Blackboard Architecture", Computers & Structures Vol., 54, No.2, pp. 255-265

Kanok-Nukulchai, W., (1986), "On a Microcomputer Integrated System for Structural Engineering Practices", Computers & Structures, Vol.23, No.1., pp. 33-37

Krisnamoorty, C.S., Venkatesh, P. P, Sudarshan, R., (2002), "Object-Oriented Framework for Genetic Algorithms with Application to Space Truss Optimization" ASCE-Journal of Computing in Civil Engineering, Vol 16 No. 1, January 2002.

Nguyen, T.H. and Oloufa, A.A., (2001), " Computer-Generated Building Data: Topological Information", ASCE Journal of Computing in Civil Engineering, October 2001, Volume 15, Issue 4pp. 268-274

Pena-Mora, F and Choudary K. K., (2001), "Web-Centric Framework for Secure and Legally Binding Electronic Transactions in Large-Scale", A/E/C Projects, ASCE Journal of Computing in Civil Engineering, October 2001, Volume 15, Issue 4 pp. 248-258

Pena-Mora, F., Vadhavakar, S., Dirisala, S.K., (2001), "Component-based Software Development for Integrated Construction Management Software Applications", AIEDAM, Vol 15, No. 2 April 2001.

Pena-Mora, F. et al, (1999), "Information Technology Planning Framework for Large Scale Projects", ASCE-Journal of Computing in Civil Engineering, Vol 13 No. 4, December, 1999.

Przybylo, W. and Mokrzycki, T., (1989), "Microcomputer Integration of AUTOCAD FEM System and 3D Graphics Using FEMA Database", Proceedings of the Second East Asia-Pacific Conference on Structural Engineering & Construction, pp. 291-296, Division of Structural Engg., Asian Institute of Technology Bangkok, Thailand

Raphael, B., Krishnamoorthy, C. S., (1993), "Automating Finite Element Development Using Object-Oriented Techniques", Engineering Computations Vol. 10, pp 267-278

Tailibayew, S.I., (1999), "Development Strategies Leading To A General – Purpose Finite Element Analysis Program Using Object-Oriented Paradigm" M. Eng. Thesis, No. ST-99-5, Asian Institute of Technology, Bangkok, Thailand