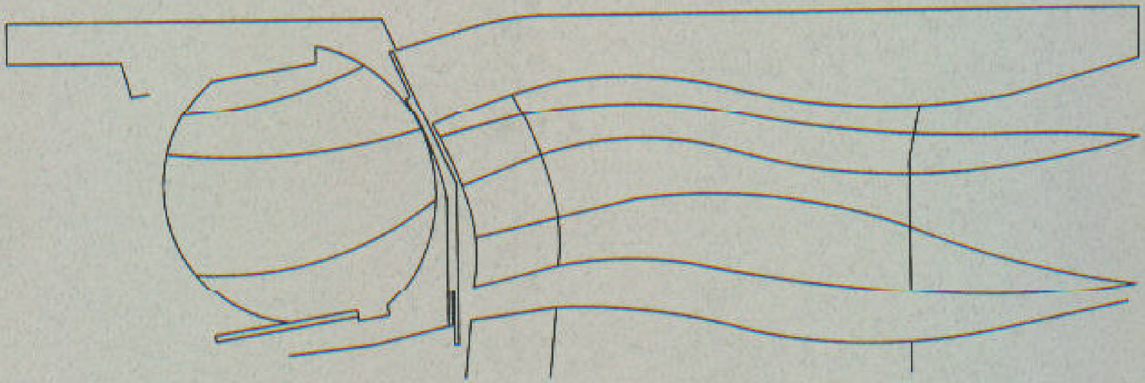
GIVING MEANING TO DESIGN DATA USING XML



By

Cosmas Tzavelis

The Louis and Jeanette Brooks Engineering Design Center

Gateway Engineering Education Coalition
Albert Nerken School of Engineering
The Cooper Union for the Advancement of Science and Art

Studies in the Design Process
and Design Education
Series A, Number 3

Foreword

This report, prepared under the auspices of the NSF Gateway Coalition by Professor Tzavelis, addresses the problems faced by participants of multidisciplinary teams working on design and construction sites not necessarily located in the same geographical area. Developing, documenting, and sharing information over a network of computers becomes very important under such circumstances.

This study makes use of a new emerging standard, Extensible Markup Language (XML), to mark up design data transferred and display it on the web. As our experience with the Gateway Concurrent Engineering project (directed at Cooper Union by Professor Wei and involving teams of student designers at other universities) has shown, this is not an abstract problem.

As pointed out by Professor Tzavelis, this work is applicable to many courses at Cooper. It is particularly well-suited to multidisciplinary projects that consider different functions of the same design object.

This work is therefore important in our effort to encourage interdisciplinary projects, and its author should be commended for proposing a comprehensive, pertinent and fruitful design tool.

Jean Le Mée
Director, Curriculum Development and Innovation

ABSTRACT 3
INTRODUCTION 4
WHAT IS XML 4
XML FOR ENGINEERING AND CONSTRUCTION DATA 5
INTEGRATIVE AXIOMATIC DESIGN 7
AN AEC PRODUCT MODEL BASED ON AXIOMATIC DESIGN 10
IMPLEMENTATION USING AN OBJECT-ORIENTED DATA MODEL 15
INTELLIGENT DATA 16
DEMONSTRATION USING SPREADSHEETS 16
THE AUDIENCE THIS REPORT ADDRESSES 19
CLASSES THAT CAN USE THESE CONCEPTS 19
DELIVERABLES 19
FUTURE WORK 19
CONCLUSION 20
APPENDIX 1. SAMPLE OF A DXF FILE FORMAT 21
APPENDIX 2. HOUSE.XLS 23
APPENDIX 3. FRAME.XLS 25
APPENDIX 4. BEAMS.XLS 27
APPENDIX 5. COLUMNS.XLS 28
APPENDIX 6. CROSSSECTIONS.XLS 29
APPENDIX 7. FOOTINGS.XLS 30
APPENDIX 8. ACADPOINTS.XLS 31
APPENDIX 9. CONNECTIONS 31
APPENDIX 10. MATERIALS 32
APPENDIX 11. LOADS.XLS 32

Abstract

There is a long-standing need to structure and give meaning to design data.

For years the only way for Designers and Constructors, to express their design ideas is through drawings and specifications. But, drawings and specifications, whether in paper or in electronic form, simply document the geometry or form of the designed objects and are rarely able to capture the function of the data or the indent of the designer. With the wide spread use of the web and the ever growing use of information technology and computers, new ways are needed to express design data in a more meaningful way that take advantage of these technologies.

In the present study, the new web based XML standard and the old Axiomatic Design theory have been combined to demonstrate a more meaningful way to communicate design intent and information. In Axiomatic Design, the design process is formally defined as "the creation of a synthesized solution in the form of products, processes or systems that satisfy perceived needs through the mapping between the functional requirements (FRs) in the functional domain and the design parameters (DPs) of the physical domain, through the proper selection of DPs that satisfy FRs" [Suh90][1].

Taking advantage of the hierarchical structure of the XML documents, with the proposed nesting of design data and design parameters (DPs) inside tags that describe their function (FRs), one may give meaning to design data. This is achieved by requiring the designer to tag form describing data or DPs, using meaningful functional specifications (FRs) or performance requirements.

To demonstrate the usefulness of this approach, spreadsheets have been used to model individual design objects expressed in XML. In addition, hyperlinks and linked formulas have been employed to model object relationships.

Introduction

Designers and constructors need to be able to express, document and communicate their ideas, in order to enable themselves as well as others, to understand their designed or constructed products. They all use, design or construct products, and they process them and improve them in order to achieve their intended function. There is also a need to capture the designer's intent and give meaning to design data so that others can understand and process them as well.

Design and construction is a multidisciplinary process that involves numerous participants not necessary located in the same geographical area. Therefore developing, documenting and sharing this information over a network of computers is very important.

For years the use of drawings and specifications in paper, and more recently in electronic format, was the only means to describe these products. But now with the ever increasing use of computers and information technology (IT), the creators and users of design and construction information need to have better means of expressing and sharing this information.

The present study, has tried to use a new emerging standard called Extensible Markup Language or XML, to mark up design data transferred and displayed in the web.

It will be shown how this standard in conjunction with design theories such as Axiomatic Design[2], can be used to document, share, transfer and reuse design and construction information, while helping to capture the designer's intent.

What is XML

XML is a way to structure data and create hierarchies using tags.

Extensible Markup Language is a text-based format that lets developers describe, deliver and exchange structured data between a range of software applications to clients or users for local display and manipulation. XML also facilitates the transfer of structured data between users and developers themselves. Vast stores of design and construction information exist today, distributed across disparate, incompatible software and databases. XML allows the identification, exchange and processing of this data in a manner that is mutually understood using custom formats for particular applications if needed.

XML resembles and complements HTML used in the web to display all kinds of information. XML describes data, such as beam name, cross-section, moment of inertia, and HTML defines tags that describe how the data should be displayed, such as with a bulleted list or a table. XML, however, allows developers to define an unlimited set of tags, bringing great flexibility to authors, who can decide which data to use and determine its appropriate standard or custom tags.

XML is a text-based format, designed especially to *store and transmit data.* An XML source is made up of XML elements, each of which consists of a "start tag" (such as <AcadLine>), an "end tag" (such as </AcadLine>), and the information between the two tags (referred to as the contents). Like HTML, an XML document holds text annotated by tags. However, unlike HTML, XML allows an unlimited set of tags, each indicating not how something should look, but what something means. For example, an XML element might be tagged as a beam, a cross-section, or a cost. It is up to each document's author to determine what kind of data to use and which tag names best fit his needs.

With both valid and well-formed XML, XML encoded data is self-describing. The open and flexible format used by XML allows it to be employed anywhere a need exists for the exchange and transfer of information. This makes it *extremely* powerful.

Here is an example of an AutoCAD line with all of its properties presented in terms of XML format.

```
<AcadLine>
<EntityType>      19      </EntityType>
<Color>           256     </Color>
<startPoint>      1233.800192  770.0163555  96      </startPoint>
<endPoint>        1233.116884  867.0666024  96      </endPoint>
<Handle>          8E2D   </Handle>
<Layer>           STRUCTURAL       </Layer>
<Linetype>        BYLAYER    </Linetype>
<LinetypeScale>        1     </LinetypeScale>
<Normal>          0     0     1     </Normal>
<Thickness>       0     </Thickness>
<Visible>         TRUE  </Visible>
</AcadLine>
```

As one may see from the above XML document, it is easy to understand the meaning of the data representing the properties of the AcadLine. To make this more obvious one may compare this format with the corresponding format of the AutoCAD dxf file for the same data (See Appendix 1). The well known and widely used .dxf file as well as any other exportable file from existing software, is practically unreadable with no sense of the meaning of their data. On the contrary with the proposed XML format and using carefully named and meaningful tags, one can easily understand what these data represent.

This is the reason it is proposed to use XML to structure Engineering Design Data.

XML for engineering and construction data

Lets take a look at the XML representation of a rectangular cross section.

```
<cross_section name="rct1" material="steel">

        <Area units="sq in"  formula="h*w">  200     </Area>

        <Mom_Of_Inertia_about_xx formula="w*h^3/12"> 6877.7
</Mom_Of_Inertia_about_xx>

      <Mom_Of_Inertia_about_yy  formula="h*w^3/12">  1666.6
</Mom_Of_Inertia_about_yy>
      <Section_Modulus_about_xx  formula="w*h^2/6">   877.4
</Section_Modulus_about_xx>
      <Section_Modulus_about_yy  formula="h*w^2/6">   333.3
</Section_Modulus_about_yy>
      <Radius_of_Gyration_about_xx
formula="sqrt(Ixx/A)">5.77</Radius_of_Gyration_about_xx>
            <design_parameters>
      <height name="h"  units="in">    20      </height>
      <width name="w"  units="in">     10      </width>
            </design_parameters>
      </cross_section>
```

As one may see from the above representation, the way to define an object like a beam cross-section is to enclose its properties within an opening tag <cross_section> and a closing tag </cross_section>.

Inside the start tag, of the cross_section or any of its properties, one can add some helpful attributes like name="rct1" or material="steel". This study proposes to use opening and closing tags like <cross_section> and </cross_section> to define design objects that they enclose more detailed information about their properties. Objects themselves, may include sub-objects (with their own tags), in which case one may simply include them inside the opening and closing tags of their parent object. To give more meaning to the tags, one may add additional attributes inside the opening tags like name="rct1" and material="steel" in the case of the <cross_section> object. The user must be careful in this case, to use the equal sign and quotes as shown above, as it is required by the XML specification.

There are many more syntax requirements for the XML standard in order for an XML object to be well formed and interpretable by web browsers such as Internet Explorer 5 (see http://www.xml.com/xml/pub/98/10/guide0.html). But this study utilizes only a subset of this specification; in order to make the design of XML objects much simpler. Therefore in addition to the basic requirement that objects must be enclosed between start and closing tags, this study proposes to use just one more feature of the XML standard, that offers an equivalent form for defining simple objects with only a few properties:

The object with the attributes Area and Volume can simply be written as:

```
<object Area="25" Volume="34"/>
```

As compared to the more explicit form of the same object:

```
<object>
        <Area> 25 </Area>
        <Volume> 34 </Volume>
</object>
```

This much simpler object definition will be used in the cases of basic or primitive objects when an object has only a few attributes and can not be broken into more detail. This is especially useful in the cases where a simple primitive object needs to be included as a property of a much more complicated parent object or as a reference to a more detailed representation.

The user must also be aware that the word naming an object tag can be more descriptive by using more than one word, such as <beam_object> instead of just <beam>. In this case though, all words must form a contiguous string like <beam_object_in_2nd_floor>; otherwise it will not be a valid XML tag.

Integrative Axiomatic Design

The Integrative Axiomatic approach to design attempts to create a philosophical framework for the design process. Utilizing concepts developed by the ancient grammarians of Sanskrit, it creates a simple, well-ordered view of all the elements involved in design.

Design may be characterized as the "providing of a set of prescriptive rules for reorganizing the elements of creation according to some purpose". It is a "way of doing things" characterized by decision-making and applicable to all "thought-out" activities [Le Mee92,]. Thus, design is a particularly human endeavor.

In axiomatic design, the design process is formally defined as "the creation of a synthesized solution in the form of products, processes or systems that satisfy perceived needs through the mapping between the functional requirements (FRs) in the functional domain and the design parameters (DPs) of the physical domain, through the proper selection of DPs that satisfy FRs" [Suh90][3].

Thus, design can be seen as a three level system. At the conceptual level, the specifications for the design are defined and the functional requirements selected to satisfy a set of needs. At the physical level, the design parameters (the "form" of the design) are embodied. In between the two is an intermediary level where the mapping of the conceptual to the physical takes place. This middle ground is a large part of the designer's task: to map the functional requirements of a design to physical design. The mapping process typically consists of iterative interaction between the conceptual and physical levels by the designer until an optimal solution is reached.

Here is one way to describe the above concepts using XML:

```
<Design>
  <Conceptual_level>
        <Specifications/>
        <Functional_Requirements/>
  </Conceptual_level>

  <Mapping explanation="Mapping of the conceptual to the physical level">

  <Physical_level>
        <Form>
              <Design_parameters/>
        </Form>
  </Physical_level>
</Design>
```

According to the axiomatic approach, "the final design cannot be better than the set of functional requirements it was created to satisfy".

Therefore, the other major task of the designer is the proper selection of functional requirements. Without the proper posing of requirements – the problem - it is impossible to satisfy the needs the design is intended to address.

This view of the design process as linked to conceptual and physical levels is mirrored in the theory of language held by the ancient grammarians of Sanskrit. They distinguish a physical level of language - sound - from a conceptual level - ideas. In the middle is a layer, which takes ideas and translates them to sound according to rules of language. The physical layer may be though of as DPs; it is where the choice of words and sounds is embodied and where XML can serve the purpose to define these words and sounds. The conceptual layer may be thought of as the functional domain, where ideas are created to satisfy needs and where again XML can be used to define hierarchies of functional requirements.

Of course, the similarity between the design process and language is not surprising. After all, language is at the core of design and is a creative process itself. Therefore, the basic rules of language must be the rules of design.

The preceding overview of design is concise but does not introduce a clear human element into the process. Design is a human activity with a social dimension, but an artificial intelligence machine could easily carry out the mapping process described above (as it often is today). Where does the moral and ethical responsibility of the designer come in?

One of the tenets of axiomatic design is that the selection of functional requirements depends on the way the designer hopes to satisfy a set of needs.

Thus, it follows that the designer has a viewpoint from which he or she selects functional requirements. To choose a viewpoint, however, one must have an awareness of who and where one is. That is, the designer must be self-aware.

The axiomatic approach to design puts forward a series of postulates, the first of which is simply: "I am." That is the designer has feelings and thoughts: consciousness. From this simple start, one may infer that the designer has a field of awareness in which he or she is a witness to thoughts, feelings and actions. The designer also has a viewpoint, which may change based on the level of awareness and understanding.

Axiomatic design theory further holds that "Not only do I perceive, feel and think, but I can act" [Le Mee92]. That is, through communication or action, the designer can change his or her surroundings. The designer has a standpoint for action based on his/her function and the set of needs to be satisfied. The standpoint governs the reach of the designer's authority in both space and time, from local surroundings to the entire universe and from an instant to the whole of time. Note that the reach of authority may be different than the reach of viewpoint introducing ethical and moral considerations.

A second postulate holds that action is a process leading to a result. The designer is responsible for investigating the consequences of an action before implementing it.

The third postulate introduces another idea from the grammarians of Sanskrit: "Factors of Action." Any action requires the presence of six factors:

* The fixed point or set of functional requirements which is the point of departure for design. The FRs spell out the design objectives.

* The object the action most seeks to reach. In axiomatic design, this is the set of design parameters that embody the physical product of the design process.

* The instrument, the most effective way to reach the object. In axiomatic design, it is the iterative mapping process used to relate the FRs to the DPs.

* The locus which defines the field of action. In design, this is equivalent to constraints, whether moral, ethical, technical, etc.

* The recipient or beneficiary of the action.

* The agent. In axiomatic design, this is the designer, the person responsible for the action.

Utilizing the XML format, the above action object, can be expressed as:

```
<action>
        <Functional_Requirements name="FRs or action objectives"/>

        <Object>
                <Design_parameters name="DPs"/>
        </Object>

        <Instrument name="mapping between FRs and DPs/>

        <Constraints/>

        <Recipient name="Beneficiary or Client"/>

        <Agent name="Designer"/>
</action>
```

Thus, design can be seen as a process where the agent uses an instrument to reach an objective (DPs) based on the requirements (FRs) and subject to constraints to satisfy the recipient.

In this process, only the agent is independent with respect to the action.

Thus, the agent is ultimately a conscious being who takes a stand with regard to a particular action and is responsible for its consequences. At all times, the designer must keep the recipient in view through the functional requirements and select the most effective design parameters to satisfy the FRs while respecting the imposed constraints.

Constraints fall into three categories:

* Input constraints are constraints in design specifications, for example the prescribed use of wood rather than steel because of local conditions.

* System constraints are imposed by the environment in which the design solution must function. For instance, a building must be designed to function subject to the physical laws of the universe in which exists.

Or a building must be built to resist vibration in an earthquake zone.

* Responsibility constraints pertain to the moral, ethical, intellectual and professional requirements that arise from the designer's responsibility for his/her actions.

Here is the previously described <action> object that includes the expanded constraints:

```
<action>
        <Functional_Requirements/>

        <Object>
                <Design_parameters/>
        </Object>

        <Instrument/>

        <Constraints>
                <Input/>
                <System/>
                <Responsibility>
                        <moral/>
                        <ethical/>
                        <technical/>
                </Responsibility>
        </Constraints>

        <Recipient/>

        <Agent/>
</action>
```

Finally, axiomatic design puts forward two imperatives in design. First, one must maintain the independence of the functional requirements. That is, they should express only the requirements to satisfy a set of needs. Secondly, one must minimize the information content of the design. In this sense, the design process follows the Second Law of Thermodynamics, which states that increasing the amount of order within a system requires increasing the disorder of the surroundings by an even greater amount. Thus, the most effective design is one that introduces the least amount of disorder into the environment. Since design is essentially a reordering of the elements of creation, the best design has the least reordering - the least information content.

(The preceding summary of the axiomatic approach to design is largely derived from [Le Mee92] and [Suh90]. The reader is referred to those works for a fascinating and more complete discussion of the subject)

An AEC Product Model based on Axiomatic Design

This study concentrates on the design Data rather than the Processes or actions that create it. Using the principles of axiomatic design, it proposes a domain Product model to store AEC (Architecture Engineering and Construction) design data using XML format that is flexible, efficient, manageable and maintainable. These attributes are obtained by modeling the structure of the data based on the Design process that created it. Rather than ordering data in a way that reflects the underlying architecture of the computer (i.e. dxf files), the present study proposes to order data in a way that reflects the design process that creates and uses it.

Any AEC information integration system must support actions - the design process. Therefore, the design objects must be composed of "factors of action," those entities that must be present for any action to take place. In fact, the proposed "axiomatic" objects will include only three out of the six discrete factors of action: design parameters, functional requirements and constraints.

In XML format:

```
<axiomatic_object>
            <design_parameters/>
            <functional_requirements/>
            <constraints/>
</axiomatic_object>
```

The agent and recipient (a.k.a. the designer and the client) are external to the system. Instruments are typically applications that query the XML data in a database, although as one may see, in some cases they too may be integrated into model XML objects.

Functional requirements (FRs) serve to define the problem the design seeks to address on a conceptual level. That is, they represent the design objectives. An object may have many individual functional requirements forming a hierarchy. At the top of the hierarchy are the abstract functional requirements. For example, most columns would have abstract FRs to "transfer loading to ground" and to "resist axial loading." As one travels down the hierarchy, however, the FRs become more specific and quantifiable ("minimize cost" or "keep stress less than allowable" for instance). These low-level FRs, called performance FRs, are easy to relate to the low-level performance design parameters discussed below.

Utilizing the XML format, one may express the above-described functional requirements for the Column, into the following XML object:

```
<FRs name="Functional Requirements" for="Column">
            <transfer_loading_to_ground/>
            <resist_axial_loading>
                        <performance_FRs>
                              <minimize_cost/>
                              <keep_stress_less_than_allowable/>
                              </keep_stress_less_than_allowable/>
                        </performance_FRs>
            </resist_axial_loading>
</FRs>
```

Design parameters (DPs) represent the form of the object at the physical level. That is, they describe the physical manifestation of the object in the real world. As such, they have something of a dual nature.

At a detailed low-level, DPs characterize an object in terms of physical form, e.g. shape, orientation, topology, material, etc. However, at a higher level, one may have a set of derived performance design parameters that arise from the low-level design parameters and the constraints subject to the functional requirements. Examples of performance DPs include internal stress, deflection and cost. In general, the iterative mapping process between FRs and DPs described in the previous section takes place between the low-level performance FRs and high-level performance DPs.

Considering that the DPs depend on the choice of FRs, the present study proposes to nest the physical representation of an object (DPs), inside the functional Requirements (FRs).

Here are the previously shown Functional Requirements for the Column, in a more detailed format enclosing the physical representation (DPs) inside them:

```
<FRs name="Functional Requirements" for="Column">
        <transfer_loading_to_ground/>
        <resist_axial_loading>
                <performance_FRs>
                        <minimize_cost>
                          <performance_DPs/>
                        </minimize_cost>
                        <keep_stress_less_than_allowable>
                          <performance_DPs>
                                <internal_stress>
                                        <DPs name="Design parameters">
                                                <cross_section_Area/>
                                        </DPs>
                                </internal_stress>
                                <deflection/>
                                <cost/>
                          </performance_DPs>
                        </keep_stress_less_than_allowable>
                </performance_FRs>
        </resist_axial_loading>
</FRs>
```

One should notice in the above XML representation, the nesting of the DPs inside the performance_FRs, and specifically inside the <internal_stress> component of the performance_DPs.

Constraints are limitations that must be satisfied by a design. There are three types of constraints: input, system and Responsibility (human).

Input constraints are proscribed initial design specifications. A requirement that only ceramic materials be used in high temperature systems is an example of an input constraint.

System constraints arise from the universe in which the modeled entities exist. For example, a beam is constrained to deform in a certain way or the stress in a column is constrained to vary in a prescribed manner because of the laws of physics, mechanics of materials, etc.

Responsibility constraints arise from human factors. Ethical, moral, aesthetic, intellectual, professional or other considerations determine design constraints. For instance, zoning laws may constrain a building to a certain height for purely aesthetic reasons, even if there are no technical, economic or other "hard" reasons for such a limit.

Here is yet another way to integrate the above concepts into an XML object called <axiomatic_object>:

```
<axiomatic_object>
  <conceptual>
        <FRs>
                <performance_FRs>
                <physical>
                        <performance_DPs>
                                <DPs/>
                        </performance_DPs>
                </physical>
                </performance_FRs>
        </FRs>
  </conceptual>
  <mapping_Instruments/>
  <constraints>
        <Input name="Design Specifications"/>
        <System/>
        <Responsibility kind="Ethical, Moral, aesthetic, etc"/>
  </constraints>
</axiomatic_object>
```

Instruments are the most effective means by which the design objectives may be reached. Typically, this is an iterative mapping of FRs into DPs. Instruments may also include synthesis (the combination of solutions), analysis (the conversion of form DPs into performance DPs) or even simple change of DPs (translation or rotation of an entity, for example).

Actions change objects using instruments. In the present study, most actions on stored information will be carried out using an external application as the instrument. In fact, this arrangement has traditionally been the case with database systems such as possible XML object databases. The database stores the data and applications manipulate it. Object-oriented database systems blur the distinction between executable code and data with the introduction of methods. With Object Oriented Database Base Management Systems, data and code are intertwined, leading to an intriguing possibility: "intelligent data." This idea will be explored further in the next section, when the implementation of axiomatic ideas in an object-oriented context will be discussed. For now, it is enough to note that in the axiomatic approach, instruments are usually external to the XML database objects, but may in some cases be integrated directly into the XML database objects.

Incorporating all the of above ideas, this is the final XML representation, that the present study proposes, for any Design entity that includes both data and processes:

```
<axiomatic_entity>
  <Product type="data">
        <FRs name="functional_requirements">
                <performance_FRs>
                        <performance_DPs kind="derived">
                                <DPs name="design_parameters"/>
                        </performance_DPs>
                </performance_FRs>
        </FRs>
  </Product>

  <Processes type="methods">
        <Constraints kind="equations">
                <Input kind="Design Specifications"/>
                <System/>
                <Responsibility kind="Ethical, Moral, aesthetic, etc"/>
        </Constraints>
        <Instruments kind="Mapping methods and local data">
                <Synthesis/>
                <Analysis/>
                <local_data/>
        </Instruments>
  </Processes>
```

```
</axiomatic_entity>
```

As one may see in the proposed XML representation of a Design entity, both data and methods are included in the <axiomatic_entity> in order to produce "intelligent objects" that know how to behave. What needs to be observed, is the inclusion of the design parameters into the corresponding functional requirements that need to satisfy. It is believed that, this is a better way to express the relationship between DPs and corresponding FRs and to capture the form, function as well as the behavior of a design entity. Therefore it is proposed that each object's data will be structured in terms of a number of FR's that include in them the performance_FRs, the corresponding performance_DPs and, the design parameters.

In addition to data, an axiomatic entity can include Processes that can provide the design methods required to change the state of the data and effectively map the FR's into DPs.

This study also proposes to use only data to represent design objects as shown below:

```
<axiomatic_object>
  <Product type="data">
        <FRs name="functional_requirements">
              <performance_FRs>
                    <performance_DPs kind="derived">
                          <DPs name="design_parameters"/>
                    </performance_DPs>
              </performance_FRs>
        </FRs>
</Product>
</axiomatic_object>
```

This is a much simpler representation that this study will actually use to demonstrate the usefulness of this approach.

Implementation using an Object-oriented Data Model

In the previous section, the concepts between functional requirements, design parameters, constraints and instruments in axiomatic objects were illustrated. Ideally, one must implement these concepts using the object-oriented data model. The object-oriented model provides several mechanisms that facilitate the axiomatic approach.

Implementing the axiomatic conceptual entities is actually quite simple. The first step is to understand the nature of each of the three (or four, if instruments and Processes are included) concepts.

The functional requirements of an object are completely characterized by static values or inequalities. Thus FRs can be modeled using only attributes, although it may be more convenient to use methods to implement the inequalities.

Representation of design parameters requires both attributes and methods. While form DPs can be represented by static values, performance DPs are often derived from a number of form DPs using mathematical equations that must be implemented as executable code.

Like performance DPs, constraints are frequently expressed as mathematics equations (deflections=FL/EA for example) or inequalities and thus are implemented as code.

Finally, instruments, when they are integrated in an object as part of its Processes, are always characterized by methods, since they do something to the data, but are not data in and of themselves. (As a practical aside, even instruments may require attributes for the internal use of their methods, i.e. local data).

Engineering data are highly interrelated. Thus, it is imperative to support relationships between objects and classes. Object-oriented systems usually support a single type of inter-object relation implemented as a logical pointer from source to destination. The pointer does not capture any of the semantics inherent in the relationship.

To implement the axiomatic approach using object oriented software, one may support only three semantic inter-object relation types: aggregation (source is a part of destination), reference (source is referred to destination(s)) and derivation (source is derived from destination(s)).

Aggregation provides the ability to assemble objects into large composite entities. It is the means by which one may assemble composite objects from primitive classes, for example.

Reference allows an object to refer to any number of other objects.

Derivation enables the attribute values of an object instance to be determined by the state of other object instances. Performance DPs may be calculated from form DPs using a derivation relationship, for example.

The above relation types are typically implemented as semantic bearing logical pointers. That is, they are pointers with a type.

The object-oriented data model directly supports two other types of relations. Generalization and specialization, arises from the use of inheritance hierarchies (source is a superclass of destination; source is a subclass of destination). Instantiation (object is an instance of class) is directly supported by the class mechanism of object-oriented systems. Any instance is related to its class by an instantiation relationship. A class may have zero or more instances. An instance may have only one class.

Intelligent Data

Integrating instruments of action directly into objects allows for an intriguing possibility: "intelligent data." Usually, the separation between action, object and instrument is maintained. The designer (agent) uses an external application (instrument) to carry out some action on an object or set of objects. In the proposed system, for example, the designer might run a program on his/her workstation to iteratively design a beam object in the centralized database based on established functional requirements. However, if one may decide to include instruments in individual objects, one would effectively give them the "intelligence" to design themselves, for example. The inclusion of instrument methods in individual classes (typically composite classes) is analogous to distributing "application fragments" over the database, blurring the traditional line between database and application. In practical use, one would only integrate the most generally useful instruments, because highly specific instruments would be useless to most users. Specialized design tasks (most tasks, in fact) would still be the province of external applications. Indeed, instruments to be integrated would likely be very generic tasks such as iterative design mapping and evaluation or simple attribute replacement (e.g. "move object", "rotate object"). It should be noted that even objects without integrated instruments have some "intelligence." Constraint methods enable an object to check itself for validity and keep all of its parts in a compatible state. However, an object is only as "smart" as its methods allow it to be. Care should be taken in the design of instruments and constraints.

Demonstration using Spreadsheets

Since the main scope of this work is to give design data meaning, programming the above concepts in a truly object oriented language such as C++ is beyond the scope of this work. For demonstration purposes Microsoft Excel spreadsheets have been used to implement the above concepts, instead.

Spreadsheets are not object oriented and one has to compromise. Even though, the reader can better visualize the previously discussed ideas by modeling these concepts in spreadsheets as it is proposed below.

A class of objects is simulated as a Workbook in excel, with individual spreadsheets as instances of this class

Each object in a spreadsheet is expressed using XML format.

Hyperlinks between Workbooks are used to simulate the Reference relationship.

Copy in a spreadsheet object and, paste special (with Paste Link) is used to update object properties in a spreadsheet object that depend on the values of other object properties in a different spreadsheet.

Formulas are used in a spreadsheet, to derive properties from others and, to simulate the Derivation relationship.

To simulate the Aggregation relationship, one can group related Workbooks in the same Folder. Or use Folders with sub-folders that group related groups of objects together.

Customized Workbook templates in XML format can be used to provide a uniform representation for a specific type of Class of objects (like structural Beams).

The outlining features of the excel spreadsheets, can be used to hide Form data or Design parameters in order to highlight their function.

One may also simulate the Instruments or methods applied to object data, using Macros. Macros that are stored in the particular type of Workbook (simulating a class of objects) can be used to apply methods that are applicable only to the specific workbook class. But Macros that are stored in the "Personal WorkBook" provided in MS Excel, can store Instruments or methods applicable to all objects or WorkBooks.

In the Demonstration Excel files that are attached to this document as Appendices 2 to 11, the following files are included:

House.xls (Appendix 2) to represent a one story house that needs to be designed.

Please note how the conceptual design has evolved from the spreadsheet named "house (1)" to spreadsheet "house (5).

In addition one may notice the use of outlining to hide data and simplify the view of data.

Frame.xls (Appendix 3) to represent the corresponding structural frame of the house.

Note that there are 10 versions of this frame that shows the path from the conceptual design, to the final design.

Beams.xls (Appendix 4) to represent the beams of the structural frame.

Please note that each workSheet contains the XML representation of the beam at the top, followed by the XML representation needed for the exchange of data with SAP2000 and AutoCAD.

Columns.xls (Appendix 5) to represent the columns of the structural frame.

Please note that each workSheet contains the XML representation of the column at the top, followed by the XML representation needed for the exchange of data with SAP2000 and AutoCAD.

CrossSections.xls (Appendix 6) to represent the cross-sectional properties of the Beams and Columns

Footings.xls (Appendix 7) to represent the Footings of the frame

AcadPoints.xls (Appendix 8) contain the points from an AutoCAD drawing that are used as centroids of Footings.

Connections.xls (Appendix 9)to represent all the connection between beams and columns

Materials.xls (Appendix 10) to represent the Steel and Concrete material properties used to build the frame.
Loads.xls (Appendix 11) to represent the loads acting on the beams and columns that also hold their response (stresses and deflections).

Regarding Loads, this study proposes to be modeled as actions on objects. While the previously discussed axiomatic objects have form, function and possibly behavior (instruments), the Loads are actions on them like move or paint. This means that actions should have input, and output, a section for the object that they are applied to, as well as possible resources or control constraints. In general Load actions may be modeled as follows:

```
<action  name=          "load" >
         <input>        <dead_load/>
                        <live_load/>
                        <wind_load/>
         </input>
         <control/>
         <mechanism>  structural_element     </mechanism>
         <output>       <reactions/>
                        <stresses/>
                        <internal_forces>      <Moment/>
                                               <Shear/>
                        </internal_forces>
                        <displacements/>
         </output>
</action>
```

In the demonstration files, links are provided that builds the frame from the ground up. This means that the acadPoints.xls file provides the AutoCAD points from which the footings.xls file receives it data, and from which the bottom support property of the column.xls file is derived and from which the beams.xls file derives its connection points. This means that the beam-ends are receiving their data from the column tops, as the column bottoms are receiving their data from the footing centroids and so on. This is a fine example of how one can model a frame the way it is actually build and designed rather than as individual objects that can float in space.

The audience this report addresses

The audience it addresses is the Engineer of any level including Engineering students working on Design projects.

Classes that can use these concepts

Classes that can use these concepts are any Design classes such as: EID101 (Engineering Design and Problem Solving), EID110 (Engineering Design Graphics), EID111 (Design, Illusion and Reality), EID142 (programming Languages), ME 120(Design Elements), CE121 and 122 (Structural Engineering), CE341 (Design of Steel and Concrete Structures), ME320 (Mechanical Design), CE363 and 364 (Civil Engineering Design), CE421 (Matrix Methods of Structural Analysis), CE422 & EID422 (Finite Element Methods), CE426 (Structural Dynamics), CE426 (Advanced Structural Design), CE450 (Civil Engineering Construction), ME420 (axiomatic design), ME453 (Computer-Aided Design).

It must be emphasized here, that multidisciplinary classes that consider the different functions of the same design object are better able to take advantage of the concepts described in this study.

Deliverables

The deliverables are this report, and a list of XML based Microsoft Excel Workbooks to be used as templates for the Conceptual and final Design of Beams, Columns and Structural Frames.

Future work

Use the above concepts in an actual class Design project that includes students from different disciplines.
Improve the excel WorkBooks to address more complex projects.
Write translators to and from XML objects to existing design software such as AutoCAD and FEM software.
Finalize the concept of Processes and Instruments as they apply to axiomatic objects.

Conclusion

The above XML representation of the design data shows that it is possible to capture the function as well as the form of designed objects. It also shows that through the careful naming of the tags and the nesting of design parameters or form inside their corresponding functional requirements helps capture the intent of the designer.

The design objects modeled in XML format, is easy to interpret and modify. One needs only a text processor to write and modify XML objects. In addition the hierarchical structure of the XML format combined with the nesting of form tags inside function tags, enables the designer to model all phases of the design including conceptual design. One can start with a simple XML object that only describes the functional requirements of the object and keep improving it by adding more tags or modifying existing ones, until he or she reaches to the final constructable object.

Personnel:

Prof. of Civil Engineering Cosmas Tzavelis, Ph.D., P.E.

Timetable:

Summer 1999

The support of Gateway coalition is greatly appreciated.

APPENDICES

Appendix 1. Sample of a DXF file format

```
BYLAYER
 70
     0
  3

 72
    65
 73
     0
 40
0.0
  0
LTYPE
  5
5
100
AcDbSymbolTableRecord
100
AcDbLinetypeTableRecord
  2
CONTINUOUS
 70
     0
  3
Solid line
 72
    65
 73
     0
 40
0.0
  0
ENDTAB
  0
TABLE
  2
LAYER
  5
A
100
AcDbSymbolTable
 70
     2
  0
LAYER
  5
4
100
AcDbSymbolTableRecord
100
AcDbLayerTableRecord
```

```
  2
0
 70
    0
 62
    1
  6
CONTINUOUS
  0
LAYER
  5
1B
100
AcDbSymbolTableRecord
100
AcDbLayerTableRecord
  2
STRUCTURAL
 70
    0
 62
    7
  6
CONTINUOUS
  0
ENDTAB
  0
TABLE
  2
STYLE
  5
B
100
AcDbSymbolTable
 70
    3
  0
STYLE
  5
C
100
AcDbSymbolTableRecord
100
AcDbTextStyleTableRecord
  2
STANDARD
 70
    0
 40
16.0
 41
1.0
 50
15.0
 71
    0
 42
0.2
```

```
  3
txt.shx
  4

  0
STYLE
  5
1C
100
AcDbSymbolTableRecord
100
AcDbTextStyleTableRecord
  2
DIMTXT
 70
    0
```

Appendix 2. House.xls

```
      <house_of_CT>
        <function name="to have enough space for a family of 4 and guests">
                <performance_FRs name="to have 4 bedrooms and 3 baths">
                        <performance_DPs name="number of bedrooms, bathrooms and theirArea and
locations">
                                <design_parameters name="Area of Bedroom1" value="" units="sqft"/>

                                <design_parameters name="Area of Bedroom4" value="" units="sqft"/>
                        </performance_DPs>
                        <performance_DPs name="number of bedrooms, bathrooms and theirArea and
locations"/>
                </performance_FRs>
                <performance_FRs name="to have 200sqft kitchen and 500sqft livingroom">
                        <performance_DPs name="Area of kitchen and livingroom">
                                <design_parameters name="Area of kitchen" value="" units="sqft"/>
                                <design_parameters name="Area of livingroom" value="" units="sqft"/>
                        </performance_DPs>
                </performance_FRs>
        </function>

        <function name="to satisfy zoning requirements">
                <performance_FRs name="max living area=3000sqft">
                        <performance_DPs name="Total area of house">
                                <design_parameters Area_of_room1="  sqft"/>
                                <design_parameters Area_of_room9="  sqft"/>
                        </performance_DPs>
                </performance_FRs>
                <performance_FRs name="set backs more than 20 ft">
                        <performance_DPs name="coordinates of footings">
                                <design_parameters name="coordinates of footing1"/>

                                <design_parameters name="coordinates of footing4"/>
                        </performance_DPs>
                </performance_FRs>
        </function>
        <function name="to have a durable structure">
                <performance_FRs name="have durable frame">
                        <performance_DPs name="durable beams and columns">
                                <design_parameters>
                                cross sections of beams and columns
                                </design_parameters>
                        </performance_DPs>
                </performance_FRs>
                <performance_FRs name="have durable Architectural elements">
                        <performance_DPs>
                                <design_parameters>
                                </design_parameters>
                        </performance_DPs>
                </performance_FRs>
        </function>

        <function name="to cost not more than $300,000">
                <performance_FRs name="the Summation of the costs of the house Parts not to exceed
$300,000">
                        <performance_DPs>
                                <design_parameters>
      <cost_of_frame>   $17,517.2       </cost_of_frame>
      <cost_of_exterior>        </cost_of_exterior>
      <cost_of_interior>        </cost_of_interior>
      <cost_of_landscaping>             </cost_of_landscaping>
                                </design_parameters>
                        </performance_DPs>
                </performance_FRs>
        </function>

        <function name="to be aesthetically pleasing">
                <performance_FRs>
                        <performance_DPs>
                                <design_parameters>
                                </design_parameters>
                        </performance_DPs>
                </performance_FRs>
        </function>

      </house_of_CT>
```

Appendix 3. Frame.xls

```
      <frame_of_House>
      <FR name="satisfy the Architectural requirements">
        <FR name="to satisfy zoning requirements and hidden beams and columns">

                <performance_FR name="zoning set backs more than 20 ft">
                <performance_DP name="locations of footings">
                        <DesignParameters name="coordinates of footings">
  <footing No="    1        " centroid="    1233.34           432.75  0.00    "/>
  <footing No="    2        " centroid="    1391.38           436.23  0.00    "/>
  <footing No="    3        " centroid="    1392.99           770.29  0.00    "/>
  <footing No="    4        " centroid="    1233.80           770.02  0.00    "/>
  <footing No="    5        " centroid="    1233.12           867.07  0.00    "/>
  <footing No="    6        " centroid="    908.13   306.60  0.00    "/>
  <footing No="    7        " centroid="    671.58   600.83  0.00    "/>
  <footing No="    8        " centroid="    692.91   863.96  0.00    "/>
                        </DesignParameters>
                </performance_DP>
        </performance_FR>
        <performance_FR name="columns on top of footings and hidden inside walls">

                <performance_DP name="location of columns">
                        <DesignParameters name="column locations">
  <column No="    1        " supported_on="      Footings.xls - '1'!A1      " height="      8
    ft"/>
  <column No="    2        " supported_on="      Footings.xls - '2'!A1      " height="      8
    ft"/>
  <column No="    3        " supported_on="      Footings.xls - '3'!A1      " height="      8
    ft"/>
  <column No="    4        " supported_on="      Footings.xls - '4'!A1      " height="      8
    ft"/>
  <column No="    5        " supported_on="      Footings.xls - '5'!A1      " height="      8
    ft"/>
  <column No="    6        " supported_on="      Footings.xls - '6'!A1      " height="      8
    ft"/>
  <column No="    7        " supported_on="      Footings.xls - '7'!A1      " height="      8
    ft"/>
  <column No="    8        " supported_on="      Footings.xls - '8'!A1      " height="      8
    ft"/>
                        </DesignParameters>
                </performance_DP>
        </performance_FR>
        <performance_FR name="beams on top of columns and hidden inside walls">

                <performance_DP name="location of beams">
                        <DesignParameters name="beam conections">
  <beam No="      1        " Start_support_on="    columns.xls - '5'!A1      " end_support_on="
    columns.xls - '8'!A1   "/>
  <beam No="      2        " Start_support_on="    columns.xls - '4'!A1      " end_support_on="
    columns.xls - '5'!A1   "/>
  <beam No="      3        " Start_support_on="    columns.xls - '3'!A1      " end_support_on="
    columns.xls - '4'!A1   "/>
  <beam No="      4        " Start_support_on="    columns.xls - '2'!A1      " end_support_on="
    columns.xls - '3'!A1   "/>
  <beam No="      5        " Start_support_on="    columns.xls - '1'!A1      " end_support_on="
    columns.xls - '2'!A1   "/>
  <beam No="      6        " Start_support_on="    columns.xls - '1'!A1      " end_support_on="
    columns.xls - '6'!A1   "/>
  <beam No="      7        " Start_support_on="    columns.xls - '6'!A1      " end_support_on="
    columns.xls - '7'!A1   "/>
  <beam No="      8        " Start_support_on="    columns.xls - '7'!A1      " end_support_on="
    columns.xls - '8'!A1   "/>
                        </DesignParameters>
                </performance_DP>
                </performance_FR>
        </FR>
      </FR>
      <FR name="frame to be durable">
        <Performance_FR name="all beams and columns to be durable">

                <performance_DP name=" max stress less than allowable in ALL elements" value="
                true or false        ">
                        <design_parameters>
      <beam No="        1         ">      <x_section_geometry shape="rectangular" name="     rct2        "
material="      Concrete          "/>     </beam>
      <beam No="        2         ">      <x_section_geometry shape="rectangular" name="     rct2        "
material="      Concrete          "/>     </beam>
      <beam No="        3         ">      <x_section_geometry shape="rectangular" name="     rct2        "
material="      Concrete          "/>     </beam>
      <beam No="        4         ">      <x_section_geometry shape="rectangular" name="     rct2        "
material="      Concrete          "/>     </beam>
      <beam No="        5         ">      <x_section_geometry shape="rectangular" name="     rct2        "
material="      Concrete          "/>     </beam>
      <beam No="        6         ">      <x_section_geometry shape="rectangular" name="     rct2        "
material="      Concrete          "/>     </beam>
      <beam No="        7         ">      <x_section_geometry shape="rectangular" name="     rct2        "
material="      Concrete          "/>     </beam>
      <beam No="        8         ">      <x_section_geometry shape="rectangular" name="     rct2        "
material="      Concrete          "/>     </beam>
      <column No="      1         ">      <cross_section shape="rectangular" name="   rct1      " material="
        Materials.xls - steel!A1  "/>     </column>
      <column No="      2         ">      <cross_section shape="rectangular" name="   rct1      " material="
        Materials.xls - steel!A1  "/>     </column>
      <column No="      3         ">      <cross_section shape="rectangular" name="   rct1      " material="
        Materials.xls - steel!A1  "/>     </column>
      <column No="      4         ">      <cross_section shape="rectangular" name="   rct1      " material="
        Materials.xls - steel!A1  "/>     </column>
      <column No="      5         ">      <cross_section shape="rectangular" name="   rct1      " material="
        Materials.xls - steel!A1  "/>     </column>
      <column No="      6         ">      <cross_section shape="rectangular" name="   rct1      " material="
        Materials.xls - steel!A1  "/>     </column>
      <column No="      7         ">      <cross_section shape="rectangular" name="   rct1      " material="
        Materials.xls - steel!A1  "/>     </column>
      <column No="      8         ">      <cross_section shape="rectangular" name="   rct1      " material="
        Materials.xls - steel!A1  "/>     </column>
                        <cross_section name="3"/>
                        </design_parameters>
                </performance_DP>
        </Performance_FR>
        <Performance_FR name="able to resist all combinations of Loads">

                <performance_DP name="LOADS">
      <DL name="dead Load"/>
      <LL name="Live Load" units="lbs/sqft">                  40        </LL>

      <WL name="Wind Load" units="lbs/sqft">                  27        </WL>

                </performance_DP>
        </Performance_FR>
      </FR>
      <FR name=" frame to be economical">
```

```
    <Performance_FR name="minimize cost of labor and materials">

            <performance_DP name="cost of labor and materials">

                    <design_parameters>
                    </design_parameters>
  <cost_of_materials>               #VALUE!        </cost_of_materials>

  <cost_of_labor>                    </cost_of_labor>
            </performance_DP>
    </Performance_FR>
  </FR>
  <FR name="frame to be aesthetically pleasing"/>

  <FR name=" frame to be constructable"/>

  </frame_of_House>
```

Appendix 4. Beams.xls

```
  <beam No="      1      " Start_support_on="     columns.xls - '5'!A1      " end_support_on="
    columns.xls - '8'!A1      ">
    <specification name="orientation">
  <local_x_axis name="perpendicular to cross_section"/>

  <local_y_axis name="pointing Upwards towards the Global Vertical direction (or in the Global X if
member is vertical)"/>
  <local_z_axis name="perpendicular to local x-y plane"/>

    </specification>
    <function name="durable to take the applied Loads">

  <linear_element_load name="      le1      "/>
    <performance_FRs name="strength">
            <performance_DPs name="Max bending and shear stresses less than allowable">

                    <design_parameters>
  <x_section_geometry shape="rectangular" name="      rct2      " material="      Concrete         "/>

  <start_elementDOF name="Local" >  1      1      1      1      1      1
    </start_elementDOF >
  <end_elementDOF name="Local">     1      1      1      1      1      1
    </end_elementDOF>
                    </design_parameters>
            </performance_DPs>
    </performance_FRs>
    <performance_FRs name="serviceability">
            <performance_DPs name="Max deflections less than allowable">

            </performance_DPs>
    </performance_FRs>
    </function>
    <function name="Architectural">
    <performance_FRs name="supported on shown abovc">

  <start_support>     <connection No="          5         " centroid="        1233.116884
    867.0666024      96        "/>        </start_support>
  <end_support>       <connection No="          8         " centroid="        692.9104195
    863.9562678      96        "/>        </end_support>
    </performance_FRs>
    </function>
    <function name="economy">
    <performance_FRs name="minimize cross-sectional area">

              <performance_DPs>
  <cost kind="material" units="dollars/linearFt">          $77.92   </cost>

  <cost name="of material" units="dollars">    $3,507.65          </cost>
  <Length units="ft">          45.02    0
              </performance_DPs>
    </performance_FRs>
    </function>
    <function name="aesthetics"/>
    <function name="constructability"/>
  </beam>
```

Appendix 5. Columns.xls

```
  <column No="        1          " supported_on="           Footings.xls - '1'!A1        " height="         8
    ft">
    <function name="durable to take the applied Loads">

  <linear_element_load name="          le1        "/>
    <performance_FRs name="stability">
              <performance_DPs >
                        <design_parameters>
  <cross_section shape="rectangular" name="       rct1      " material="       Materials.xls - steel!A1     "/>

  <bottom_elementDOF name="Local" >       1        1        1        1        1        1
    </bottom_elementDOF >
  <top_elementDOF name="Local">    1        1        1        1        1        1
    </top_elementDOF>
                        </design_parameters>
              </performance_DPs>
    </performance FRs>
    </function>
    <function name="Architectural">
    <performance_FRs name="supported on footing shown above and reach floor above">

  <bottom_support> <footing No="     1          " centroid="       1233.335618      432.7463669       0
    "/>        </bottom_support>
                        <design_parameters>
  <top_support>       <connection No="          1         " centroid="        1233.335618
    432.7463669      96        "/>        </top_support>
                        </design_parameters>
    </performance_FRs>
    </function>
    <function name="economy">
    <performance_FRs name="minimize cross-sectional area">

            <performance DPs>
  <Length units="ft">          8          </Length>
  0          <performance_FRs name="minimize cross sectional Area">    0

  <cost name="of material" units="dollars">      #VALUE!          </cost>

            </performance_DPs>
    </performance_FRs>
    </function>
    <function name="aesthetics"/>
    <function name="constructability"/>
  </column>
```

Appendix 6. CrossSections.xls

```
  <cross_section shape="rectangular" name="     rct1       " material="       Materials.xls - steel!A1     ">
    <specification name="orientation">
  <local_x_axis name="perpendicular to cross_section"/>
  <local_y_axis name="pointing Upwards towards the Global Vertical direction (or in the Global X if
member is vertical)"/>
  <local_z_axis name="perpendicular to local x-y plane"/>
    </specification>
    <function name="durability">
    <performance_FRs name="have geometric properties that produce acceptable stresses">

            <performance_DPs>
  <A name="Area" units="in sq"          formula="h*w">           180.0    </A>
  <J name="Polar_Mom_Of_Inertia" units="in^4"          formula="">             </J>
  <Izz name="Mom_Of_Inertia_about_zz" units="in^4"    formula="w*h^3/12">    4,860.0  </Izz>
  <Iyy name="Mom_Of_Inertia_about_yy" units="in^4"    formula="h*w^3/12">    1,500.0  </Iyy>
  <Azz name="Shear_Area_about_zz" units="in^2"        formula="w*h*5/6">     150.0    </Azz>
  <Ayy name="Shear_Area_about_yy" units="in^2"        formula="w*h*5/6">     150.0    </Ayy>

  <Szz name="Section_Modulus_about_zz" units="in^3"   formula="w*h^2/6">     540.0    </Szz>
  <Syy name="Section_Modulus_about_yy" units="in^3"          formula="h*w^2/6">     300.0
    </Syy>
  <rzz name="Radius of Gyration about zz" units="in"   formula="sqrt(Izz/A)"    5.2      </rzz>
  <ryy name="Radius of Gyration about yy" units="in"   formula="sqrt(Iyy/A)"    2.9      </ryy>
  <Modulus_of_Elasticity units="psi">          29,000,000       </Modulus_of_Elasticity>

  <EIzz name="section stiffness" units="lbs*in^2"        formula="ModOfElasticity*Ixx">  1.4094E+11
    </EIzz>
            <design_parameters>
  <h name="height or thickness" units="in">     18        </h>
  <w name="width" units="in">          10        </w>



            </design_parameters>
    </performance_DPs>
    </performance_FRs>
    </function>
    <function name="economy">
    <performance_FRs name="minimize cross sectional Area">
            <performance_DPs name="stress just less than allowable">
<unit_cost kind="material" units="dollars/cFt">          $34.0   </unit_cost>
<cost kind="material" units="dollars/linearFt">          $42.5   </cost>
    </performance_FRs>
    </function>
</cross_section>
```

Appendix 7. Footings.xls

```
<footing No="      7        " centroid="     671.5843775    600.8340037    0        ">
<function name="Transfer Frame Loads to soil">
  <performance_FRs name="Durability">
          <performance_DPs name="shape">
                  <design_parameters>
                  </design_parameters>
          </performance_DPs>
          <performance_DPs name="Degrees of Freedom of the connection to frame">

                  <design_parameters>
<DOF>   1       1       1       1       1       1       </DOF>
                  </design_parameters>
          </performance_DPs>
  </performance_FRs>
</function>
<function name="Architectural">
  <performance_FRs name="connection to frame">
          <performance_DPs name=" location and elements connected at contact point">

<centroid>        671.5843775    600.8340037    0        </centroid>
<connected_elements>                            </connected_elements>
          </performance_DPs>
  </performance_FRs>
</function>
</footing>
```

Appendix 8. AcadPoints.xls

```
<ACAD_entity name="     Point8D9C      ">
<EntityType>      22      </EntityType>
<Color>  256      </Color>
<coordinates>     692.9104195    863.9562678    0        </coordinates>
<Handle>  8D9C    </Handle>
<Layer>   STRUCTURAL  </Layer>
<Linetype>        BYLAYER       </Linetype>
<LinetypeScale>   1       </LinetypeScale>
<Normal>          0       0       1       </Normal>
<Thickness>       0       </Thickness>
<Visible>         TRUE    </Visible>
</ACAD_entity>
```

Appendix 9. Connections

```
<performance_FRs name="minimize cross-sectional area">   0      0      0      0      0      ">

          <function name="Transfer Frame Loads">
          <performance_FRs name="Durability">
                    <performance_DPs name="shape">
                              <design_parameters>
                              </design_parameters>
                    </performance_DPs>
                    <performance_DPs name="Global Degrees of Freedom of the connection joint">

                              <design_parameters>
<Joint_DOF name="Global">   0      0      0      0      0      0       </Joint_DOF>
                              </design_parameters>
                    </performance_DPs>
          </performance_FRs>
          </function>
          <function name="hide within Architectural form">
          <performance_FRs name=" connect frame elements so that they are hidden from view">

                    <performance_DPs name=" location and elements connected at contact point">

<centroid>      0      0      0      </centroid>
<connected_elements>                   </connected_elements>
                    </performance_DPs>
          </performance_FRs>
          </function>
</connection>
```

Appendix 10. Materials

```
<steel>
<Mass_density units="lbs/cubIn"> 7.32E-04        </Mass_density>
<Weight_density units="lbs/cubIn">         0.283    </Weight_density>
<Modulus_of_Elasticity units="psi">        29000000       </Modulus_of_Elasticity>
<Poisson's_ration>        0.3      <Poisson's_ration>
<Temperature for="shown_properties">       0        </Temperature>
<coeff_of_Thermal_Expansion>     6.50E-06        </coeff_of_Thermal_Expansion>
<Fy name="Yield_stress" units="psi">       36000    </Fy>
<Fu name="Ultimate stress" units="psi">    58000    </Fu>
</steel>
```

Appendix 11. Loads.xls

```
<joint_load>
<input type="load">
          <Force units="kips" direction="global" type="dead_load">     0      0      -2.44    </Force>
```

```
            <Moment units="kips*ft" direction="global" type="dead_load">          0       0       0
            </Moment>
    </input>
    <mechanism type="point">        reference          </mechanism>
    <output>            <reactions comment="only_at_supports">
            <Force units="kips" direction="global" type="dead_load">
            <Moment units="kips*ft" direction="global" type="dead_load">                          </Force>
            </Moment>
            </reactions>
            <displacements>
            <displacement units="in" direction="global" type="dead_load">         0       0       0
            </displacement>
            <rotation units="rad" direction="global" type="dead_load">     0       0       -2.44    </rotation>
            </displacements>
    </output>
    <control/>
    </joint_load>
```

[1] Suh 90 Suh, N.P., "The principles of Design". Prentice Hall, Inc., Engewood Cliffs, NJ, 1962
[2] Le Mee Jean, Le Mee, J.,"Integrative Axiomatic Approach to Design" The Cooper Union for the Advancement of Science and Art, NY, 1992
[3] Suh 90 Suh, N.P., "The principles of Design". Prentice Hall, Inc., Engewood Cliffs, NJ, 1962