A Universal Format for Architectural Program of Requirement

A prerequisite for adding architectural programming information to BIM data models

Ehsan Barekati[1], Mark Clayton[2]
[1,2]Texas A&M University
[1]ehsan.barekati@tamu.edu [2]mark-clayton@tamu.edu

This paper is a report on authors' ongoing effort in creating a universal model for architectural programming. Authors analyse three well-known formats for architectural programming and devise a UML model representing each format. The UML models are further analysed and compared to form a super UML model that can bring together all the three formats under one roof and act as a universal format for architectural programming (UFPOR). The results improve the integration of architectural programming and BIM data models and are of value to the software development field and architectural programing.

Keywords: *Building Information Modelling, Data Modelling, Architectural Programming, Interoperability*

A UNIVERSAL FORMAT FOR ARCHITECTURAL PROGRAM OF REQUIREMENT

A prerequisite for adding architectural programming information to BIM data models

This paper is a report on authors' ongoing effort in creating a universal model for architectural programming. Authors analyse three well-known formats for architectural programming and devise data models representing each format. The data models are further analysed and compared to form a parent data model that can bring together all three formats under one roof and act as a universal format for architectural programming (UFPOR). The results improve the integration of architectural programming and BIM data models and are of value to the software development field and architectural programing.

INTRODUCTION

BIM technology has improved communication between the disciplines involved in built environment projects by providing a repository for the project information. Every discipline can access the most updated status of the project and be informed on the most recent changes made by other players (Eastman 2008). Open source standards for BIM data models such as IFC give every discipline the opportunity to plug into the BIM model and share information with other disciplines (Björk, Laakso 2010).

While the benefits are abundant, there are some required steps to join the platform. Disciplines can get connected to the BIM data models by providing a compatible internal data model. Providing such model is the first step toward BIM integration. While

data modelling is a standard procedure in information technology, it can be rather challenging when the targeted discipline involves qualitative information or lacks a comprehensive ontology. Architectural programming is one of the key components of every design project that happens to fall into this category (Cherry 1999).

AIA defines a program of requirement (POR) as "An organized collection of the specific information about the client's requirements which the architects need in order to design a particular facility." (American Institute of Architects., Palmer 1981). There are different interpretations of this definition in Architectural, Engineering, Construction and Operation (AECO) industry (Hershberger 1999). Exclusion of architectural programming information makes BIM data models fall short from the BIM core premise, which advocates the inclusion of the whole lifecycle of design information. "Can we devise a structure that is comprehensive enough to embed all the existing formats for architectural programming?" This is the question that this paper will answer.

METHODOLOGY

Among all the existing POR structures, the ones offered by William Pena (Peña, Parshall 2001), Henry Sanoff (Sanoff 1977) and Donna Duerk (Duerk 1993) are chosen to represent a scaled collection of diverse POR formats. These scholars are chosen based on the popularity and comprehensiveness of their work. Our goal is to translate the text to object-oriented models. Object-oriented design is an approach in software design where a system is conceptualized as a collection of interacting objects. An object is a group of related methods and variables that represent a unique entity (Bruegge, Dutoit 2004). Object-oriented design is the approach selected by the de-facto and official standards in the AECO industry. Representing POR structures through Object-oriented systems is an important step in making them interoperable with the existing BIM standards.

The object-oriented model is presented through simplified UML class diagrams. UML is a widely accepted and ISO endorsed method for analysing and documenting processes so that they can be implemented as software (Bruegge, Dutoit 2004). We chose to include a simplified version of the UML diagrams to increase readability.

Three distinct UML models are compared with each other to distinguish the similarities and the differences. Through this step a UML model is proposed that can embed all the three UML models. We call this model the UFPOR (Universal Format for Program of Requirements).

POR MODELS

Sanoff's Model

Sanoff describes a program as a statement of intent. He believes a program is a collection of desired set of events. In Sanoff's method Objective is the key concept in organizing a program. Designers try to achieve some objectives through the design and programmers contribute by setting the objectives. Sanoff believes "Management by objectives" (MBO) would help the designer throughout the design. Sanoff introduces two more concepts besides "Objectives" to articulate this six step cycle; "Activity" and "Performance Requirements." Below I will define each of these three steps in more detail and explain their relationship.

Objectives. As we saw, objectives are the backbone of this model. Sanoff looks at objectives in different scales and believes they help an organization to:

- Define its "purpose of existence"
- Define areas of responsibility
- Evaluate accomplishments
- Clarify individual pursuits
- Make personal views visible

Activity. Objectives can be defined without any direct environmental indication. In the context of an architectural project, we accomplish the objectives by modifying or creating built environments. Activities

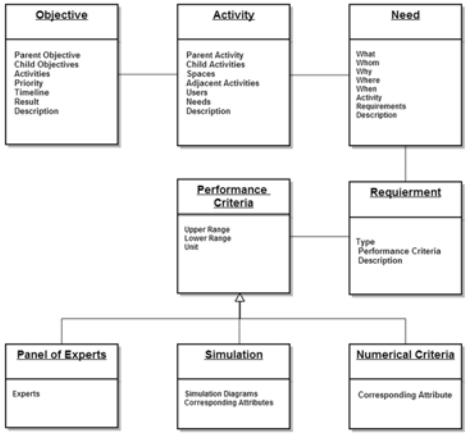


Figure 1
UML diagram that shows the five main classes along with their attributes and relationships. As you can see, every instance of the Objective class has references to other instances of the same type, defining them as its children and also a reference to a parent instance of the same class. This pattern is known as Composite pattern in software engineering (Gamma et al., 1995).

are our achievement plans. The concept of "Activities" is introduced to connect the objectives to a built environment. Activities describe how people utilize spaces.

Sanoff suggests breaking down each activity into sub-activities and also drawing an interaction matrix to discover adjacency and grouping requirements. Sanoff defines a requirement as a quantifiable statement about a proposed behaviour. A requirement can be based on fact, intuitition, or assumption.

Performance. Sanoff introduces the concept of performance as a bridge between requirements and the built environment assessment. Performance is a procedure to calculate the attributes of the physical environment to compare with the requirements posed in the previous steps. Performance requirements are supposed to answer to the users' needs.

After developing a performance requirement, its performance criteria should be defined. Performance criteria are characteristics that will be used in evaluating whether the requirements are being met or not.

Summary of Sanoff model (A narrative description for UML model)

To convert the processes that we described to UML diagrams, let's recite Sanoff ideas briefly.

- A program consists of three interrelated systems. Objectives, Activities, and Requirement.
- Objectives can be broken into Sub-Objectives and a priority value can be assigned to them.
- Each object should start with "To" and should have a single result to accomplish.
- To accomplish the result, a timeline should be added to an objective.
- Activities connect objectives to the built environment.
- Activities describe how people would utilize spaces
- Activities can be broken down into sub-activities.
- Activities describe how people would utilize spaces.
- Several activities may reside in one physical space or a single activity may occupy a group of spaces.
- A requirement can be based on fact, intuitive, or assumption.
- Requirements are supposed to answer to the users' needs.
- Evaluation techniques can rely on numerical techniques, simulation techniques, or a panel of experts.

From these bullet points we can identify four major components (classes) in Sanoff's model Objectives, Activities, Needs, Requirements, and Performance Criteria (Figure 1).

Pena's Model

Pena considers programming the architect's first and often most important task. Pena's model is simple and straight forward, yet yields tangible results. That is probably why this model is one of the most widely used models within practitioners. Pena declares five steps for programming.

Establish Goals, Collect and analyse Facts, Uncover and test Concepts, Determine Needs, State the Problem.

Pena defines four main areas, or as he calls them design determinants, to structure the types of information needed to define a comprehensive architectural program. Function, Form, Economy, Time.

The design determinants are applied to each of the five steps that we already talked about leaving the programmer with a five by four table. Pena defines the scope of each of the key players:

1. Goals: What does the client want to achieve and Why?

2. Facts: What do we know, What is given?

3. Concepts: How does the client want to achieve the goals?

4. Needs: How much money and space? What level of quality?

5. Problem: What are the significant conditions affecting the design of the building? What are the general directions the designer should take?

Pena believes only practical goals that can be achieved through concepts should be part of the program. Facts should be related to goals otherwise they bring no value to the program.

Pena breaks down each of the determinants into smaller parts. It is important for us to study all the parts of the determinants since they will be essential parts of our data model.

1. Function implies "what's going to happen in their building." It concerns activities, relationship of spaces, and people-their number and characteristics, key words are people, activities, and relationship.

2. Form related to the site, the physical environment (psychological, too) and the quality of space and construction. Form is what you will see and feel. It's "what is there now" and "what will be there." Key words are: Site, Environment and Quality.

3. Economy concerns the initial budget and quality of construction, but also may include consideration of operating and life cycle costs. Key words are: Initial Budget, Operating Cost, Life Cycle Costs.

4. Time has three classification-past, present and future-which deals with the influences of history, the inevitability of changes from the present and projections into the future, Key words are: Past, Present, Future.

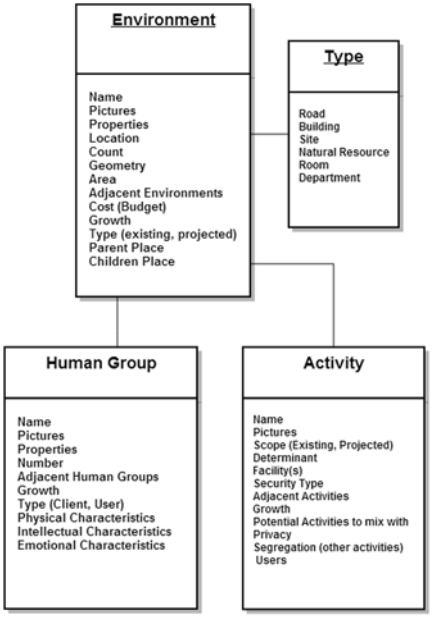
Summery Pena and Parshall model (A narrative description for UML model). A closer look at Pena's model reveals three main classes: People, Activities, and Environment. Rows and column of Pena's table are more appropriate to be considered as the properties of the classes than being the classes themselves. Relationship, as the name describes is to explain how classes relate to each other and cannot be taken as an independent class. Site is a special instance of environment and Quality can be represented through the attributes. All sub-categories of the Economy determinant are different aspects of "Cost" which can be applied as an attribute to all the base classes. Time has a similar story as it can be applied to the base classes as an attribute.

Introducing the main three base classes

Environment. Environment class represents any form of a built or natural environment that the

project is related to, or contains regardless of form and scale. This would include the project site, neighbouring buildings, existing facilities on site, roads, proposed departments, rooms and etc. The main attributes to represent an environment are: Name, Location, Number Geometry, Adjacent environment, Cost, Growth, Parent environment, Children environment (Figure 2).

Figure 2
UML diagram representing Pena's model



People. People class would represent all the people that would have an influence on the project. That includes Users, Clients, Society, and etc. The main attributed to represent a group of people are: Number, Related People, Growth, Physical Characteristics, Intellectual Characteristics, Emotional Characteristics.

Activity. People are connected to the environments through activities. Activities represent how and why people would interact with the environment. The main attributes for activities are: Privacy level, Users, Environment, Growth, Potential activities to mix with, Segregation (other activities.)

Duerk's Model

Duerk's model is the most recent one out of the three. Duerk defines architectural programming as "A systematic method of inquiry that delineates the context within which the designing must be done. It also defines the requirements that a successful project must meet." She considers programming the first part of the design process and holds the program responsible for fulfilling the future inhabitants' dreams, hopes, wishes and desires.

Analysis of the existing state. Existing state is the context of the project and includes all the information about site analysis, user profiles, codes, constraints and climate. Duerk uses design issues as the main category for searching about the existing state. She believes by using this method we can avoid collecting more information that we know how to apply or as she calls it "merely interesting facts."

Issues. Duerk defines an issue as "Any matter, concern, question, topic, proposition, or situation that demands a design response in order for the project to be successful." She believes there are generic issues (e.g. circulation, safety, territoriality, privacy, image, and flexibility) that apply to every architectural project, however, the priorities may be different. Duerk considers Form, Function, Economy and Time in Pena's model design issues.

Facts. Duerk defines facts as the objective information about the site, climate, context and other verifiable measurements. Some of the facts may raise an issue and require a design answer. On top of the facts that are related to the project specific issues, there are some facts that are related to the generic issues and thus should be collected in any project.

Projection of the future state.

Solutions. Any potential design candidate in response to an issue is a solution. At the most basic level all the solutions take a physical form. Solutions can regard different design topics to achieve their goals. Some solutions can affect the used materials

whiles others would affect Lighting or Form.

Values and Goals. In Duerk's model values are where the people involved in a project including clients, users, authorities can weigh in. Different people have different values and require different solutions. To reflect the values more systematically Duerk proposes a method in which issues turn into goals by going through the filter of values. Based on this model the project goals are directly or indirectly connected to an issue/value pair. The indirect route goes through parent goals. Duerk defines a goal as "A concise statement of the designer's promise to the client about the quality of the design in relationship to a particular issue."

Performance requirement. Performance requirements are associated with the goals and guarantee that the qualities defined by the goals will be met in the project. Each goal will result in several PRs. Duerk declares three characteristics for a PR, it should be: specific, measurable and operational.

Summary of Duerk's model and base classes. Although Duerk's model presents a more clear structure for a POR, it is rather challenging to propose an object oriented system to capture all the data. Some of the bold topics such as "Facts" and "Issues" look like the base classes at first, but a more accurate investigation reveals that facts are the attributes of other classes such as "User", "Context", and "Site" (Figure 3). Issues are also a required attribute for the "Goals".

User. This class represents the needs, attributes, values and grouping of the users. In that sense, this class represents not only facts, but also values. Each instance of this class can contain sub-categories of users within itself. For example in designing a library while library visitors can be described through an instance of the "User" class, it can also contain smaller groups that are defined based on their age such as "Children" and "Adults" where their specific characteristics will be described.

Context. This class contains mainly qualitative information about the context where the project has been defined. The attributes of this class are very helpful in identifying the issues.

Site. Site is a critical class since it provides all the information on codes, climate and other specifications.

Values. Values are qualitative statements that form the identity of our clients and users. In Duerk's model we cannot propose any solution without considering the values.

Solutions. All the elements representing the future state can be represented through one class with a hierarchical order. Solutions are the result of passing the issues through the filter of values, therefore each solution is related to one or more values. Each solution can be broken down into smaller solutions (children) and can be part of a bigger solution. Some solutions can be measurable (performance requirements.)

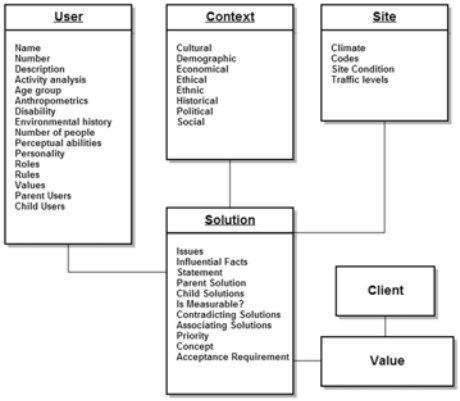


Figure 3
UML diagram representing Duerk's model

UFPOR

UFPOR is a super data model that can encompass the three models that we analysed in the previous chapter. We start with the similarities shared by all the three models to form a solid foundation, then we cover the unique characteristics of each model through adding flexible and generic parts to our model.

By looking at Pena's, Sanoff's and Duerk's model, we distinguish a lot of similarities. They all draw a

path from present to future, from where we are now to where we want to be. They propose methods to study the clients and where they are coming from through their values, goals, and financial budget to discover where they should go.

They all also talk about similar topics. "Goal" and its synonyms are ubiquitous in all models. All models try to draw a path to clarify the big goals and break them down into attainable and measurable requirements. They talk about "People" and try to know them better. Users, clients, society and owners are all human beings with goals, values and desires that can contradict each other. All three methods tried to propose different ways to discover different groups of people who are influential in a project. They also talked about "Environments" whether existing or proposed for future, whether as small as a room or as big as a city. They all advocate studying the existing environments such as the city in which we are proposing a new project, the neighbourhood, project site, and existing structures in the site to understand what we should add to this collection. They also talked about "Activities" the interaction between people and environments. We cannot decide what our buildings should be and what they should not be if we do not know how people would use the place and interact with its components. All the models also propose a hierarchy of goals, objectives and requirements. These are different ways to describe what the project or a specific part of a project "Need"s to succeed. An area requirement is what a room needs to operate successfully and a "Mission Statement" is what the whole project needs to succeed.

We identified four primary classes that are part of all the three models that we studied. They may be referred to with different names, but the concept is shared by all three major models that we studied in the previous chapter.

UFPOR BASE CLASSES

Environment

The purpose of creating a POR is to design an environment. Environments, in their most generic sense, are the main topic in a POR. We define a place as an entity that can host and contain people or other environments. By our definition a city is an environment because it can host buildings (other environments) and the people who live in that city. A bathroom or a walk-in closet is also an environment because they can temporarily host people. As you can see, our definition is scale independent. It is also not limited to the ones that are built by human beings.

People

People are the reason that we care for places. People can take different roles in a project. Users, clients, society, and authorities are all groups of people with values, requests and powers. While they have different roles, they share a lot of attributes. Similar to places, people can be broken into categories as well. "Users" of a school can be divided into students, teachers, and the administrative staff. Here are "People" attributes based on the three models that we defined previously.

Activities

Activities define the connection between the people and the places. Activities are the reason why people make places and they are the main topic in a POR. We study people to know what they want to do (activities) and then define places where those activities can happen. Similar to the places and people, activities can be broken down into sub-activities. For example "Preparing food" as an activity can be broken into "Storing Ingredients," "Washing Dishes," "Cooking" and some more activities. As you can see activities are usually associated with a verb. Here are "Activity" attributes based on the three models that we defined previously.

Needs

One of the main purposes behind a POR is to empower the clients to answer their needs. "Need" is a special class in our model. Needs create a web where all parts of a POR get connected to each other. We start with people and by discovering their needs proceed to activities, then by discovering ac-

tivities' needs we discover Environments. Detailed and quantified space requirements are attached to environments as their needs. From the data modelling stand point, a project mission (the largest need of a project) and a required area assigned to a room are both needs, even though they are at the two ends of a tree structure. While large needs are hard to measure, it is very easy to measure the success of smaller and quantified needs. That is why we analyse and evaluate large goals that are hard to measure to produce smaller and more measurable goals. PORs break down large and qualitative goals into smaller quantitative goals that are easy to measure, and then we can evaluate the success of answering the large needs based on the success of their children needs. Here are need attributes based on the three models that we defined previously.

Assessment

Needs may project their fulfilment assessment to their children needs either directly or by proposing Activities or Environments and defining their needs. Eventually we will face a tree structure of connected needs. Fulfilment of each need should be either directly or indirectly assessed. Direct assessment happens through execution of the "Assessment" attached to the need, and indirect assessment happens through evaluating the fulfilment of the child needs. The assessment class contains the following attributes.

Type. The assessment can be based on Nominal, Ordinal, Interval or Ratio.

Attribute under Assessment.

THE ATTRIBUTE THAT IS BEING MEASURED

Acceptance Criteria. Based on the assessment type, acceptance criteria should be defined. It can be a numerical range for Interval and Ratio types. For Nominal and Ordinal assessments we rely on human judgement and an acceptance criteria should be defined, for example it can be approval from both the client and users.

Assessor. For Nominal and Ordinal assessments involved "People" should be mentioned. For Interval and Ratio assessments, an "Assessor" implementation that defines the mathematical equations used in the assessment should be referenced.

Corresponding IFC Attribute. For Interval and Ratio assessments, the corresponding attribute from the IFC model should be mentioned, this attribute will be used by the assessor to set the result value.

Corresponding Revit Attribute. For Interval and Ratio assessments, the corresponding attribute from the Revit model should be mentioned, this attribute will be used by the assessor to set the result value.

Result:. The assessment will produce several types of results based on the assessment type and the result will be assigned to this attribute.

UFPOR Key Concepts

To further define the three base classes that we just define and identify their sub-classes and attributes, there are some key concepts to keep in mind. Knowing these key concepts will help us in defining the subclasses and their attributes.

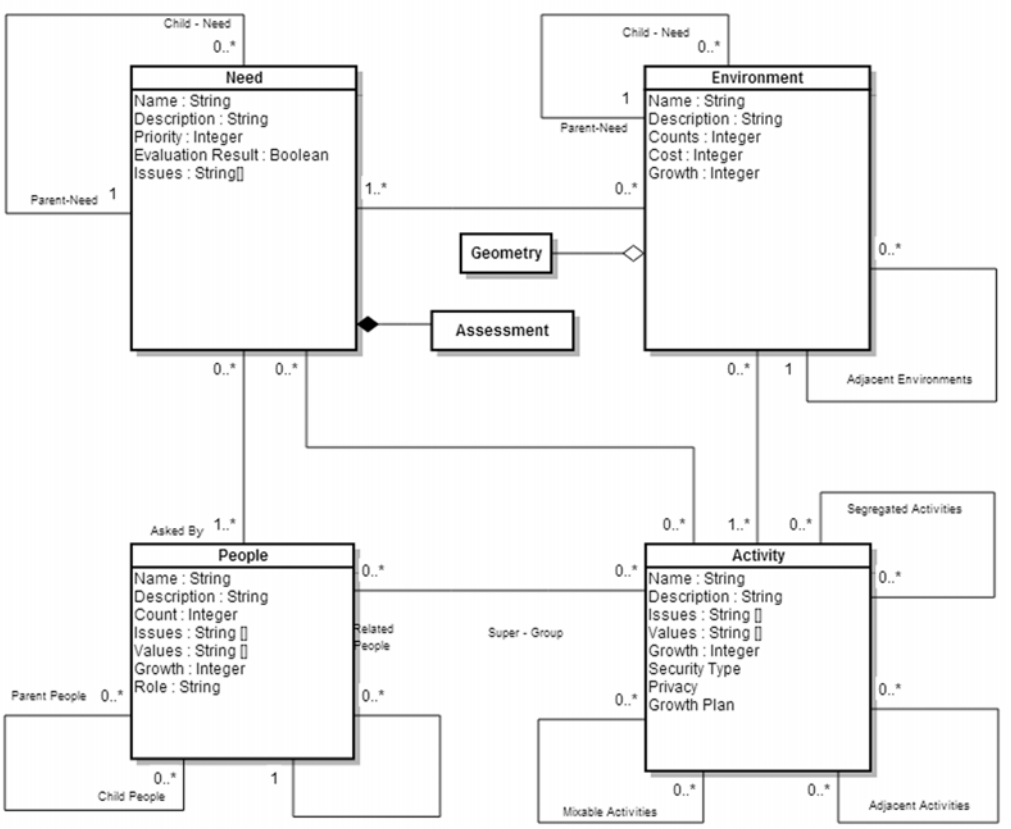
Time

By looking at every base class through the filter of time, we can collect valuable data and in result make confident decisions. It also helps us in identifying all the sub-classes that need to be instantiated for every base class. "User" is a sub-class of "People" in the future, while "Client/Owner" is more related to the present time. When it comes to "Place", we are dealing with past, present, and future. While the existing structures represent the past, neighbours and the site belong to the present time. The proposed spaces and departments which are the result of architectural programming phase represent the future.

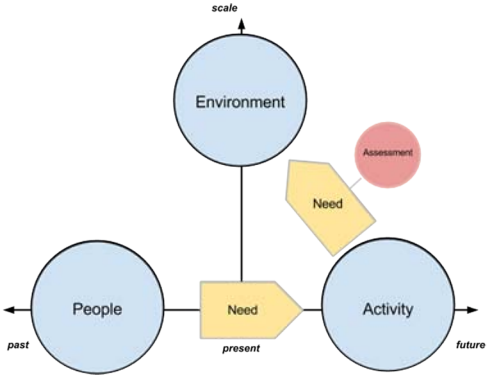
Composition

Composition is a design pattern in object oriented design where instances of a certain type can form parent-child relationships with each other and form a tree structure representing the whole system. By

Figure 4
UFPOR UML diagram



using the composition pattern we can theoretically model a building by using only one class (Environment). Rooms will be defined as children of the departments and departments will be children of the building. The same pattern can be used to create activities and sub activities as well as creating a hierarchical model for needs (Figure 4).



CONCLUSION

Figure 5 is the UML model representing all the main base classes and their relationships. To simplify the model, class methods are not added to the diagram and only major attributes are shown. This data model can be used as the base to create comprehensive data models supporting a wide range of POR structures.

REFERENCES

American Institute of Architects and Palmer, M.A. 1981, *The architect's guide to facility programming*, Architectural Record Books, Washington, D.C.

Björk, B and Laakso, M 2010, 'CAD standardisation in the construction industry — A process view', *Automation in Construction*, 19(4), pp. 398-406

Bruegge, B 2004, *Object-Oriented Software Engineering Using UML, Patterns and Java*, Prentice Hall

Cherry, E 1999, *Programming for design : from theory to practice*, John Wiley, New York

Duerk, D.P. 1993, *Architectural programming : information management for design*, Van Nostrand Reinhold, New York

Eastman, C.M. 2008, *BIM handbook : a guide to building information modeling for owners, managers, designers, engineers, and contractors*, John Wiley, Hoboken, N.J.

Gamma, E., Helm, R., Johnson, R. and Vlissides, J. 1995, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley Longman, Amsterdam

Hershberger, R.G. 1999, *Architectural programming and predesign manager*, McGraw-Hill, New York

PEÑA, W and Parshall, S 2001, *Problem seeking : an architectural programming primer*, John Wiley, New York

Sanoff, H 1977, *Methods of architectural programming*, Dowden, Hutchinson & Ross, Stroudsburg, PA

Figure 5
UFPOR base classes and their relationship