CIFE CENTER FOR INTEGRATED FACILITY ENGINEERING

Design Scenarios – Enabling Requirements-Driven Parametric Design Spaces

By

Victor Gane and John Haymaker

CIFE Technical Report #TR194
August 2010

STANFORD UNIVERSITY

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

c/o CIFE, Civil and Environmental Engineering Dept.,
Stanford University
The Jerry Yang & Akiko Yamazaki Environment & Energy Building
473 Via Ortega, Room 292, Mail Code: 4020
Stanford, CA 94305-4020

Design Scenarios: Enabling Transparent Parametric Design Spaces

Victor Gane[1], John Haymaker[2]

[1]PhD Candidate, Center for Integrated Facility Engineering, Civil and Environmental Engineering, Stanford University, vgane@alum.mit.edu
[2]PhD, AIA, LEED AP, Founder of Design Process Innovation, johnrhaymaker@gmail.com

Keywords
Conceptual design, ontology, parametric modeling, process mapping, requirements modeling, design spaces.

Abstract

This paper presents a novel methodology called Design Scenarios (DS) intended for use in conceptual design of buildings. DS enables multidisciplinary design teams to streamline the requirements definition, alternative generation, analysis, and decision-making processes by providing a methodology for building and managing requirements driven design spaces with parametric Computer Aided Design (CAD) tools. DS consists of four interdependent models: (1) *Requirements Model* – stakeholders and designers explicitly define and prioritize context specific design requirements; (2) *Scenarios Model (SM)* – designers formally transform these requirements into actions necessary to achieve them, and determine the geometric and material parameters, interrelationships, and potential conflicts; (3) *Parametric Process Model (PPM)* – CAD experts build and represent the technical implementation of a SM in a parametric model to enable design teams to manage and communicate its CAD models; (4) *Alternative Analysis Model* – analyze and visually report performance back to the designers and stakeholders. This paper motivates the need for the DS methodology thorough industry case studies, and establishes points of departure for the methodology through literature review. Next, the paper details the elements and methods in the methodology, describes its implementation into a software prototype, and provides an example to illustrate how DS can potentially

enable multidisciplinary teams to generate and communicate larger and better performing design spaces more efficiently than with traditional methods.

1 Introduction: the need for effective conceptual design processes

The conceptual design of buildings is a complex process in which multidisciplinary teams construct and assess four spaces. The *objective space* consists of the *constraints* and *goals* determined by project stakeholders. The *alternative space* includes all possible design options describing geometric and/or material design decisions to be made, as well as the options chosen to generate specific *alternatives*. The *impact space* captures the performance of *alternatives* on *goals* and *constraints*. The *value space* shows how well the *alternatives* meet the stakeholder preferences on *goals* and supports the selection of successful *alternatives* [i]. The construction and exploration of these spaces is difficult with today's methods because the translation of multi-stakeholder requirements into specific parameters used to generate an alternative space with a clearly understood value has not yet been formalized in AEC.

Researchers [ii, iii, iv] argue that successful designs emerge from exploring large design spaces, while designers' bounded rationality forces them to narrow these spaces [v]. Designers first reduce the alternative space to address constraints such as the boundaries of the project site, and then seek to find designs that maximize value to their goals. To assist them in generating alternatives, designers often adopt a scenario – a collection of structures and behaviors that represent the design intent [vi]. In other words, a scenario is a set of selected constraints, which restrict the design space that needs to be further considered by capturing a set of related design decisions [vii]. A scenario is a "design pattern" [viii], or a "solution to a problem in a context" [ix]. In computer science a scenario (also called Theme, Pattern or Style) provides a "set of predefined subsystems, specifies the responsibilities of these subsystems, and includes rules and guidelines for organizing the relationships between them" [x] and has an explicit representation [vii]. In the AEC industry a scenario is a less formal construct generally communicated verbally or through hand sketches. Figure 1 illustrates an example of a traditional AEC scenario, in which the design architect with little or no participation of other design stakeholders might use the local context (e.g., Austrian Alps widely used for skiing) to implicitly constrain the design space (e.g., propose a building shape that draws its reference from skiing activity).

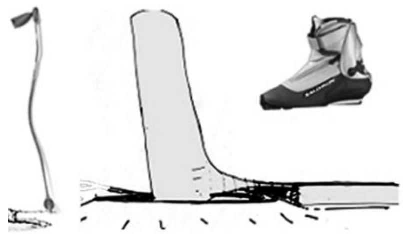


Figure 1: Example of AEC scenarios – concept sketches of a high-rise located in the Austrian Alps, which are derived from a ski pole (left) and a ski boot (right) (Gane and Haymaker, 2010). With permission from SOM.

Constructing effective design spaces require explicitly communicating the design rationale but existing methods lack the structure required to efficiently capture and reuse knowledge, generate new insights, and develop consensus [xi]. Communicating design rationale is especially important for building design problems, which require “a multiplicity of views, each distinguished by particular interests and derived from an understanding of current problem solution techniques in the respective domain” [xii].

Researchers [xiii, xiv] identify two primary strategies to search through a design space – *high breadth, low depth*, which leads to multiple scenarios with a broad spread of options but little analysis, and *low breadth, high depth*, which leads to few scenarios with low spread of options but more comprehensive analysis.

In [xv], we documented how traditional high-rise conceptual design process leads to a low breadth, low depth search strategy, in which the objective space is ill defined and the rationale used to create design spaces is poorly captured and communicated. Today’s methods do not ensure clarity of objectives and good practice depends entirely on the personal approach of individual designers. This does not make good design practices scalable, repeatable, and “automatable”. Today, with parametric methods designers can generate large alternative spaces using a high breadth, low depth (only geometry-based requirements can be assessed) search strategy. However, with traditional conceptual design methods, they are unable to leverage parametric methods to understand the impact and value spaces to select best alternatives. Design Theory and Systems Engineering researchers argue that to solve these shortcomings, design teams must address the following **needs**:

1. Capture and prioritize stakeholders' and decision makers' requirements [xvi, xvii];
2. Develop scenarios by decomposing requirements into actionable descriptions about 'how' to achieve them;
3. Translate the scenarios into qualitative and quantitative input and output parameters to describe physical and functional characteristics of a [xviii];
4. Represent and manage geometry, dependencies, constraints, and CAD operations illustrating the parametric CAD model structure [xii];
5. Manipulate and record parameter values to generate design alternatives [xvii];
6. Visualize the alternatives;
7. Evaluate the alternatives [xix];
8. Compare evaluations and facilitate objective decision-making [xx].

Most of these needs are not new and several have been addressed by prior research. An integrated solution to enable effective use of parametric CAD is lacking, however. Specifically, this paper addresses the following primary question about these gaps:

- What is an ontology and method for designers and CAD experts to develop well defined and comprehensive alternative spaces that connect well defined and comprehensive objective, impact, and value spaces?

Figure 2 summarizes the main concepts introduced so far.

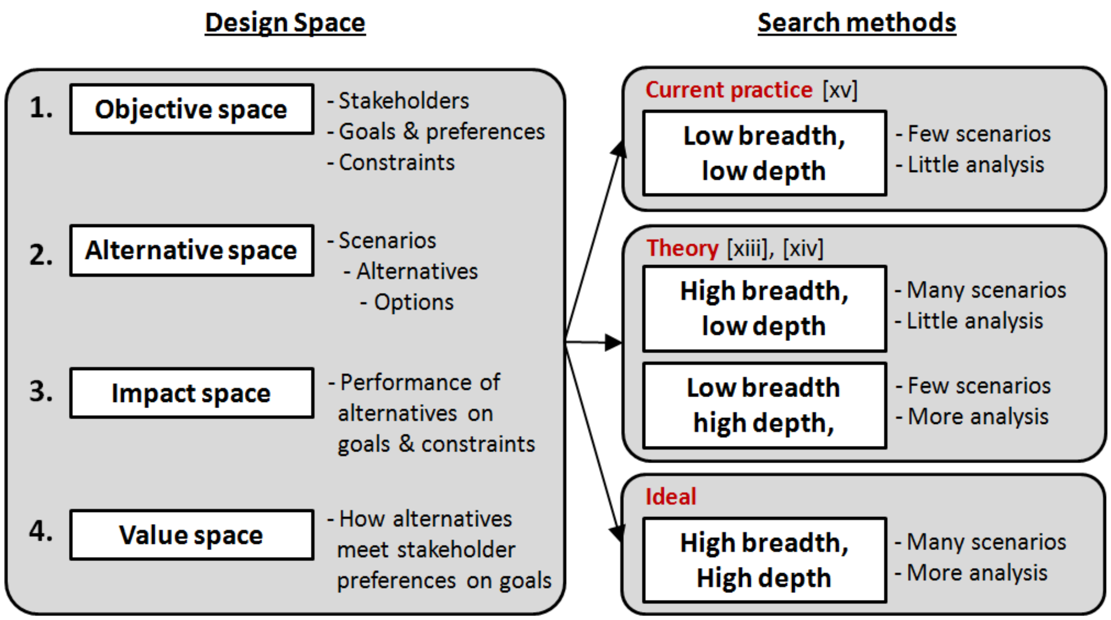


Figure 2: Summary of the concepts used to describe a design space and search methods.

To understand the quality of design spaces we need to measure how well-defined and comprehensive they are. In section 2 we introduce a framework for measuring the quality of design spaces. The framework is not the focus of this paper. However, the importance of this framework is to enable testing and understanding the impact of Design Scenarios and other proposed methodologies on the quality of design spaces. Our intuition is that as design teams generate and analyze scenarios explicitly, they can explore better-defined design spaces that lead to better designs. Section 3 reviews literature to establish the relevant Points of departure for developing a methodology that can help designers construct higher quality design spaces. Section 4 details the Design Scenarios methodology, and illustrates its use of a test case. Section 5 concludes with a discussion of strengths and weaknesses of Design Scenarios methodology and previews its application in industry.

2 Framework for measuring the design space quality

Since traditional conceptual design methods don't ensure clarity of the objective, impact, and value spaces, but the ideal practice rejects such ambiguities, we define a framework to enable designers to assess the quality of parametric design spaces (Table 1). Previous research review and synthesize metrics and methods for measuring design space quality [i], and clarity [xvi]. In this paper we build on these to synthesize the following framework for assessing how well-defined and comprehensive the parametric design spaces are:

Table 1: Framework for measuring design space quality.

	Metric	**Definition**
Objective space	Objective Space Size	What is the number of project goals and constraints considered?
	Objective Space Clarity	Is the value function explicitly and broadly communicated? The clarity is determined through documented statements describing stakeholders, goals, constraints, and preferences.
	Objective Space Quality	Are the project goals and constraints determined by all key stakeholders? A low quality denotes participation of <50% of stakeholders; medium quality: 51-80%; high quality: 81-100%.
Alternative space	Number of Scenarios	What is the number of design scenarios considered?
	Total Option Space Size	What is the total number of possible options comprising a scenario? Discrete versus continuous parameters are used to determine this metric by multiplying the constrained range of values of input parameters (e.g., building length between 30m and 40m) and their reasonable increment (e.g., 1m for building length).
	Generated Option Space Size	What is the number of the generated design options for a scenario?
	Options Space Quality	What is the ratio of Total Options Space Size to Generated Options Space Size? A 1.0 ratio is ideal because it covers the complete Design Space for a given scenario. Statistical sampling of the space could also yield high quality option spaces, but as none of the cases used in our research have involved this, we reserve this for future research.
	Alternative Space Size	What is the number of the generated design alternatives for a scenario?
	Alternative Space Clarity	Are the scenarios, designers' logic, and the parameters describing these scenarios clear?
	CAD Model Clarity	Are the structure of the parametric CAD model and the connection of parameters in the CAD model to requirements clear?
	CAD Model Quality	How many CAD models were generated for each design scenario? The target is to satisfy all geometry-based requirements with one parametric model, which denotes high quality.
Impact space	Impact Space Size	What is the number of formal model-based analyses performed to determine the value of each alternative?
	Impact Space Clarity	Is the process and results of performing each analysis explicitly depicted (i.e., repeatable)?
	Impact Space Quality	What is the ratio of Impact Space Size to Objective Space Size? A 1.0 ratio is ideal (i.e., for each requirement a formal analysis was performed).
Value space	Value Space Size	Out of the total number of generated alternatives, how many alternatives have a clear value determined? The value is determined by designers understanding how well each Objective Space requirement is met.
	Value Space Clarity	Is the total value of each generated alternative explicitly defined?

Our aim with these metrics is not to achieve a full characterization of the conceptual design exploration process, but rather to provide a set of standard terms and measurements that support observation, comparison, and improvement of existing and novel processes.

In the remainder of this paper, we summarize existing research and identify gaps in existing concepts addressing the needs outlined in section 1. The major contribution of this paper is the Design Scenarios (DS) methodology, which we introduce in section 4.

3 Points of Departure: design space exploration methods

This section discusses research and gaps in addressing the identified needs. Concurrent Engineering integrates and parallelizes multidisciplinary tasks. Quality Function Deployment, Requirements Engineering, and Axiomatic Design provide formal frameworks for defining requirements and the roles these play in decision making, as well as prescribing a recommended course of action for achieving these requirements. Process Modeling languages represent and measure design spaces. Parametric Modeling efficiently generates alternatives spaces. While each of these methods address important subsets of the identified needs, gaps remain in how design requirements can be translated into parametric design spaces, which is the contribution of this paper.

3.1 Concurrent Engineering – integrate and parallelize tasks

Several case studies show that poor definition or misunderstandings of requirements are major causes of system failure in software engineering [xxi], mechanical engineering [xxii], and in AEC [xxiii, xxiv]. Requirements-driven methods propose systematic approaches for generating, prioritizing, and managing design requirements.

Concurrent Engineering (CE) is a framework for achieving multidisciplinary objective spaces. CE addresses the limitations of traditional sequential design development methods by describing a set of technical, business, manufacturing planning, and design processes that are concurrently performed by elements of the manufacturing organization prior to committing to production [xxv]. Cross-process integration is at the core of concurrent design and consists of a multidisciplinary team method and engineering of product lifecycle [xxvi]. An already mature field, CE is at its third generation [xxvii]. The first generation addressed the limitations of sequential product development by noting the content of each design part, thus allowing independent parts to be processed in parallel. The second generation introduced the missing communication/negotiation among decision makers needed to determine the goals across the entire design process, and to relax some constraints. The current generation of CE helps determine the latest moment in the design process when binding decisions can be made. All four characteristics identified by Fukuda that describe successful application of CE apply to generating AEC design spaces as well: (1) high rate of design and process definition change *(change rate)*; (2) high rate and short cycles of new design developments *(speed)*; (3) designs with complex configurations that vary by client *(complexity)*; (4) design processes that require multiple teams to produce a single product *(multiple design teams)*.

A key issue in concurrent engineering from a designer's perspective is how to bridge the multitude of models required to support at various stages a complex design process. Although concurrent engineering is almost universally advocated today, it is hard to execute when large multidisciplinary projects are involved. CE requires a set of analytic tools and procedures to make its concepts operational [xxviii]. In the context of this research, CE partially addresses need #1 by enabling designers to capture design requirements. As concurrent engineering does not explicitly address the construction of parametric alternative spaces, the remaining needs are unmet.

3.2 Quality Function Deployment – translate user needs into design characteristics

Quality Function Deployment (QFD) is one of the methods comprising the field of systems engineering [xvii] and an important point of departure for this research. QFD is a multi-phase design to production management model, which captures and prioritizes customer needs (objective space) and translates them into engineering design characteristics (alternative space). Vagueness in requirements eventually yields indifference to customer needs, while trivial characteristics make the team lose sight of the overall design and stifle creativity [xxix]. QFD avoids ambiguity in interpreting engineering characteristics through a systematic analysis of each characteristic (impact space). In QFD large-scale systems are decomposed by multidisciplinary teams into modules and evaluated against target requirements and cost by means of matrices [xxx]. A popular matrix example is "House of Quality", which provides the means for inter-functional planning and communication. The QFD process starts with the customer requirements, continues with 'functions' required by the products or services to be developed, and ends with identifying the means for optimal 'deployment' of available resources to produce the desired products or services. Research shows that the competence of engineering designers is related to their ability to consider design constraints [xx]. Traditional QFD tools are enhanced by assessment methods that include constraints [xxxi].

QFD helps design teams determine the objective, alternative, and impact spaces. It enables understanding stakeholder requirements, engineering characteristics, and their relationships and target values, and has been extended to help guide designers in the translation of requirements into feasible design options [xxxii]. However, QFD does not enable translating requirements into parametric CAD models. In the AEC industry, PREMISS [xxiii], Decision Dashboard [xxiv], and MACDADI [xxxiii], are other examples of methodologies for eliciting requirements and relating them to building design alternatives. Similar to QFD, these methodologies also lack the means to reliably identify and relate parameters to drive geometric design spaces from requirements models. In the context of this research, QFD satisfies need #1 by enabling designers to capture and prioritize design requirements, partially satisfies need #2 by helping decompose requirements for a single scenario into actionable descriptions, and satisfies need #7 by evaluating design options against customer requirements.

3.3 Requirements Engineering – determine and manage requirements

Requirements Engineering (RE) provides another method to build objective and alternative spaces by formalizing the requirements gathering and specification process represented in the form of a checklist of requirements. Originating in systems and software engineering [xxxiv], RE overcomes the drawbacks of traditional software development methods, in which the developed systems are often technically good but unable to appropriately respond to user needs [xxxv]. RE states why a system is needed based on current and foreseen conditions, what requirements the system will satisfy, and how the system is to be constructed [xxxvi]. An expanded RE definition is concerned with making such goals operational by transforming them into services and constraints, and assigning responsibilities to agents, including humans, devices, and software [xxxvii]. Reasoning with goals can also help resolve conflicts among stakeholders. For example, it is important to capture the fact that one goal can prevent another from being satisfied. AND/OR graphs are used to capture goal refinement links. An OR node represents a choice between possible decompositions while an AND node represents a required decomposition. A conflict-link between two goals is introduced when the satisfaction of one goal may prevent another from being satisfied.

In the context of this research, RE partially addresses need #1 by enabling designers to capture but not prioritize design requirements; addresses need #2 by decomposing requirements into actionable descriptions of how to achieve them; partially addresses need #4 by representing and managing dependencies and constraints, but not geometry and CAD operations.

3.4 Axiomatic Design – generate requirements and enable parameters

Axiomatic Design (AD) provides a theoretical framework to help reduce the complexity of the design space and improve decision making at all levels [xxxviii]. AD represents design in terms of four domains: (1) Customer Domain identifies end user needs and design specifications (objective space); (2) Functional Domain identifies functional requirements needed to satisfy customer needs; (3) Physical Domain identifies designs satisfying the functional requirements (alternative space), and (4) Process Domain identifies the processes needed to determine design parameters [xxxix]. AD defines design as a process of mapping designers' requirements from the functional to the physical domain. Suh defines Functional Requirements (FRs) as the minimum number of independent requirements that characterize a design solution. AD stipulates two fundamental axioms that govern the design process. The Independence Axiom states that the independence of FRs has to be always maintained. In other words, in case of design problems with multiple FRs, a good design solution is made of design parameters (DP) that result in the independence of the FRs from each other. The Information Axiom states that information content of the design must be minimized.

The output of AD is a design matrix used to determine relationships between DPs and associated FRs. The shape of the matrix is used to distinguish between good and bad designs. Uncoupled designs are considered ideal because adjustments to the FRs are the easiest to make. A coupled design is less desirable given the increased complexity in relationships between a DP and several FRs. In the context of this research, AD partially addresses need #1 by helping to capture but not prioritize design requirements; partially addresses need #3 by enabling designers to translate requirements into design parameters without distinguishing between input and output parameters; and partially addresses need #4 by representing dependencies between requirements and parameters, but not constraints, geometry, and CAD operations.

3.5 Process modeling – represent and measure design spaces

Building a shared ontology is critical for increasing the effectiveness of multidisciplinary teams [xl]. Process modeling is a medium for building shared ontologies to help organizations plan, measure, compare, and adopt well-defined processes. Generally, there are three applications for process models: a) *descriptive* for describing what happens during a process; b) *prescriptive* for describing a desired process; c) *explanatory* for describing the rationale of a process [35]. Some languages help system developers define software and databases. IDEF (Integration DEFinition) is a family of modeling languages from systems engineering covering issues such as functional modeling, data acquisition, and simulation [xli]. Unified Modeling Language (UML), also from systems engineering, consists of structure and behavior diagrams to describe a system's functional requirements, structure, procedural flow of class objects, etc. [xlii]. Froese [xliii] describes many of the core and application process models for AEC, including IRMA – Information Reference Model for AEC, BPM - Building Project Model, ICON – Information / Integration for Construction, GRM – Generic Reference Model. GTPPM [xliv] integrates multiple use-cases with differing data requirements to define databases that facilitate collaboration among design teams. Other languages are intended for use directly by design teams. For example, Value Stream Mapping [xlv] helps teams illustrate the flow of activities, and information that produce value in a given process, while Narratives [xlvi] help teams model and manage the information and the sources, nature, and status of the dependencies between information in a process. These existing process modeling methods lack a representation formalism for communicating the structure of parametric CAD models (GTPPM being the exception), as well as their relationships to the requirements they address, and performance they achieve. In the context of this research, process modeling partially addresses need #1 by capturing but not prioritizing design requirements; partially addresses need #2 by showing actions but not decomposing requirements into actions.

3.6 Parametric modeling – generate alternative spaces

The development of procedures for generating design alternatives is an active research area. For example, shape grammars [xlvii] are a class of production systems used to generate geometric alternatives based on a set of transformation rules. Graph grammars consist of a set of rules that illustrate ways of constructing a design product or process as a graph represented by nodes denoting objects and arrows denoting relations between objects [xlviii]. Multidisciplinary Design Optimization methods guide generative methods to automatically select optimal designs [xlix]. Others [l, ii] adopt more human-centric approaches, regarding the concept of "brainstorming" as the backbone of creative thinking. A balance is needed between brainstorming for initial idea generation and geometric adjustment for refining alternatives. Parametric modeling can support building design spaces with great breadth (multiple geometric alternatives) and partial depth (analysis of geometry-based requirements only).

Parametric CAD is used to create and manage geometric alternative spaces. Also called constraint or feature-based associative modeling, parametric modeling can enable designers to shift from creators of single designs to designers of systems of inputs and outputs that generate design spaces. The concept of “features” encapsulates generic shapes or characteristics of a product with which designers can associate certain attributes and knowledge useful for reasoning about that product [li]. To design parametrically means to design a constrained system that sets up a design space that can be explored through the variations of parameters [lii]. Using parametric models, designers can create an infinite number of objects, geometric manifestations of a previously articulated schema of variable dimensional, relational or operative dependencies [liii]. Designing with multiple constraints without an efficient constraint management system is a daunting task. *Design sheets* are an example of a constraint management methodology in which design models are represented as constraints between variables in the form of nonlinear algebraic equations organized into bipartite graphs and constraint networks [liv]. Using design sheets to define parametric models, however, is not intuitive given the overwhelming number of constraints that need to be described at the schema level and the inability to visualize geometry, a capability that AEC designers need. Therefore parametric systems using geometric constraint programming to graphically impose constraints helps designers solve the relevant nonlinear equations without having to explicitly formulate them [lv]. Existing methods such as Building Object Behavior (BOB) [lvi], and software solutions such as Bentley’s Generative Components or McNeel’s Grasshopper for Rhinoceros address parts of the needs by helping designers define the structure of the parametric models (need #4), manage parameter values to generate alternatives (need #5), visualize alternatives (need #6), and geometrically evaluate alternatives with output parameters (partially need #7). However, parametric modeling needs a formal method for deriving constraints and parameters from requirements, and for relating the resulting alternatives to analyses performed outside of the parametric model.

In summary, each point of departure helps address parts of the identified needs. However, an integrated solution is still missing, including ontology for building multidisciplinary AEC alternative spaces and systematic transfer of design requirements from the objective space to the alternative, impact and value spaces. Figure 3 graphically summarizes the relationship of the points of departure to the needs we identified.

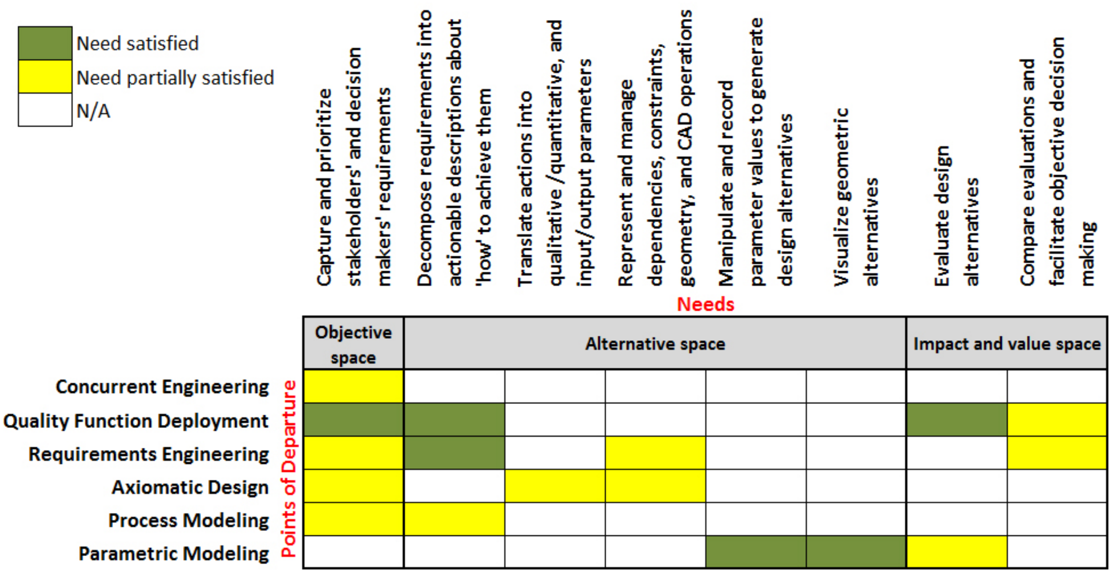


Figure 3: Summary of how points of departure satisfy the identified needs. This paper focuses on creating ontology for building multidisciplinary AEC alternative spaces and methods to translate design requirements from the objective space to the alternative, impact, and value spaces.

4 Design Scenarios - methodology description

The Design Scenarios (DS) methodology is an integrated solution to the identified needs. It enables design teams to develop and capture parametric scenarios in a relational network connecting objective, alternative, impact, and value spaces. DS builds models for each of the four spaces. However, to enable the translation of requirements into alternatives, we divided the alternative space into two subspaces: *Alternative Logic Space*, where designers capture their logic for addressing requirements, and *Alternative Geometry Space*, where CAD experts use designers’ logic to build requirements-driven parametric models. Because each space requires the participation of various roles, we introduce the concept of an *Enabler* (e.g., Architect, CAD expert) who uses a *Method* (e.g., Create objective) to generate an *Element* (e.g., Goal, parameter). Each model contains various enablers who are either humans or the computer. DS currently has a total of 30 methods and 29 elements. The selection of methods and elements varies for each model (see Chapter 1).

Figure 4 illustrates the DS process, which after the project administrator completes the project setup, starts with building the *objective space* in the Requirements Model (RM). The RM enablers are the stakeholders and designers, who **concurrently** create and prioritize project constraints and goals. The process continues with constructing the *logical alternative space* in the Scenarios Model (SM). The SM enablers are the computer and the designers, who **concurrently** decompose the requirements transferred by the computer from the RM into key geometric and/or material parameters and relationships. The computer then transfers the SM parameters into the *geometric alternative space* in the Parametric Process Model (PPM), where the CAD experts define the structure of dependencies between parameters, geometric constraints, CAD operations, and geometry. CAD experts use the PPM to construct the parametric model and generate design alternatives. The process continues with building the *impact space* in the Alternatives Analysis Model (AAM). The AAM enablers are the designers, who determine the performance of alternatives given the RM requirements. The process is finalized with building the value space, which in DS is also completed in the AAM. The enabler is the computer, which determines the value of each analyzed alternative in relation to the goal targets and preferences, thus enabling design teams to make objective decisions.

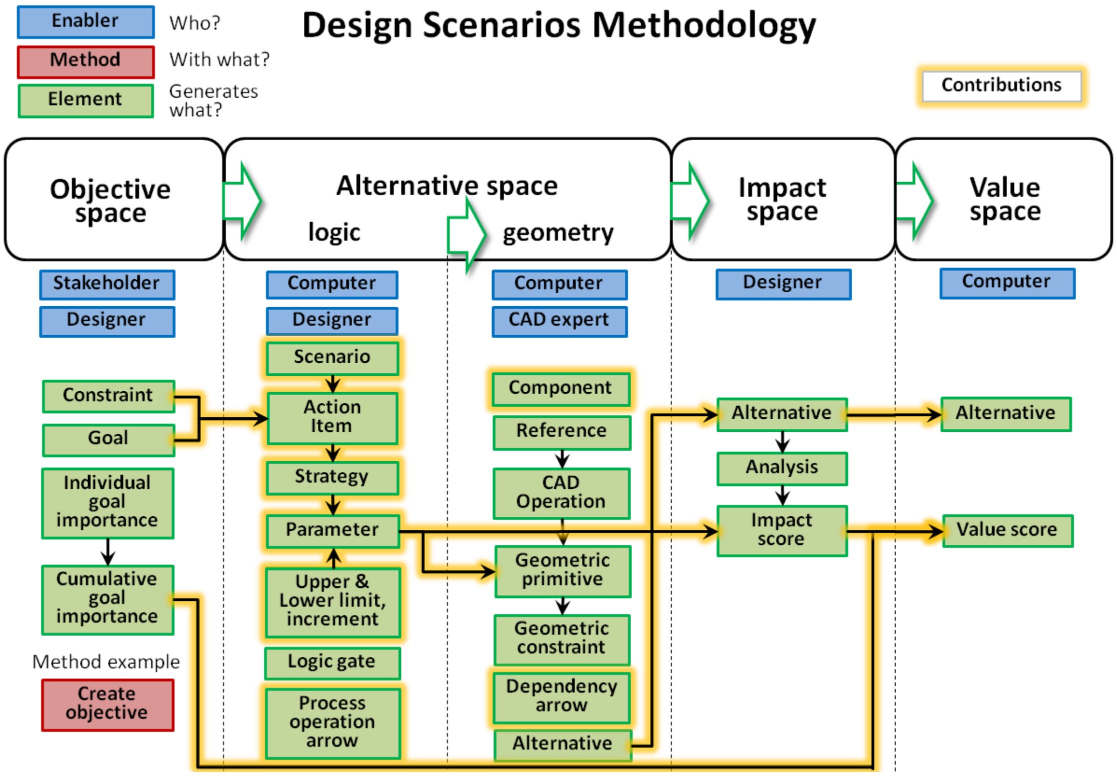


Figure 4: Design Scenarios methodology process description. The Objective Space is captured in the Requirements Model; the Logical Alternative Space in the Scenarios Model; the Geometric Alternative Space in the Parametric Process Model; the Impact and Value Spaces in the Alternatives Analysis Model.

We implemented the Design Scenarios methodology into a web-based software prototype with the same name developed in Java and Ruby on Rails, and supported by a MySQL database management system. In addition to the four DS models comprising the methodology and represented in either tabular or process model format, the software contains a Project Administration interface used to create new projects and add users, and a Project Setup interface used to create projects roles, assign users to these roles, determine access privileges to each model, and assign stakeholder influence weights. A description of these two modules can be found at www.designscenarios.com.

4.1 Requirements Model (RM)

Methodology description

Building on a Concurrent Engineering framework, the RM parallelizes the requirement definition process of all project stakeholders (e.g., client, architect, mechanical engineer, structural engineer) to enable collective understanding of each other's requirements. MACDADI and Requirements Engineering provided the foundation for the RM elements. An RM is a tabular model built by project stakeholders and designers who concurrently generate *constraints* and *goals* (two out of four RM elements). Constraints must be satisfied, while goals can be traded off against each other when finding an optimal design. When all stakeholders finish generating requirements, each stakeholder distributes 100 points over the identified goals to represent *individual preference* (third RM element). Many rating methods can be used, however this technique was chosen because of its simplicity and ease of implementation. When all stakeholders finish assigning preferences, the computer generates a *cumulative goal importance* score for each goal, the sum of which is normalized to 100 points (fourth RM element).

Software Implementation

In Table 2, we describe the RM in detail. The left column gives the visual representation and definition of each element of the RM, while the right column uses the EXPRESS data model [lvii] to describe each concept as implemented in the software prototype. Some user and system-defined inputs use free-form string data types that enable users to represent either text or values.

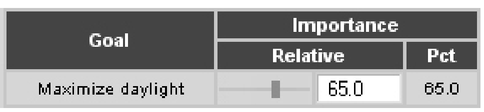
Weighting stakeholders is also relevant and has been implemented in the Project Setup interface of the software prototype. Requirements can be qualitative or quantitative, and can range from those defined by stakeholders (e.g., client: building use, space efficiency; planning department: shadows, density of development), to those established by the designers (e.g., architect: design language; mechanical engineer: daylight factor, energy comfort).

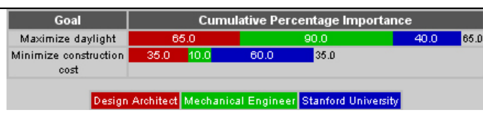
Table 2: RM graphical notation, definitions, and data schema in the Design Scenarios software prototype.

Term notation / definition	Schema description


Constraint – restriction on the quantitative or qualitative value of a design parameter.
	User inputs:
	`ENTITY Constraint`
	`discipline name: ARRAY OF STRING;`
	`discipline abbreviation: ARRAY OF STRING;`
	`constraint name: ARRAY OF STRING;`
	`value: ARRAY OF STRING;(free-form)`
	`unit: ARRAY OF STRING;`
	`END_ENTITY;`


Goal – quantifiable or qualitative value of a design parameter that is desirable to be achieved.
	User inputs:
	`ENTITY Goal`
	`discipline name: ARRAY OF STRING;`
	`discipline abbreviation: ARRAY OF STRING;`
	`name: ARRAY OF STRING;`
	`target value: ARRAY OF STRING; (free-form)`
	`unit: ARRAY OF STRING;`
	`END_ENTITY;`


Goal preference – a value expressing the relative importance of a goal to a stakeholder
	System inputs:
	`ENTITY Goal importance`
	`goal percentage value: ARRAY OF STRING;`
	User inputs:
	`goal relative value: ARRAY OF STRING; (free-form)`
	`END_ENTITY;`


Cumulative goal importance – a value expressing the sum of importance of a goal to all stakeholders
	System inputs:
	`ENTITY Goal cumulative importance`
	`goal cumulative value: ARRAY OF STRING;`
	`END_ENTITY;`

The major benefit of building a RM is the process of determining a comprehensive set of multidisciplinary requirements, which can help eliminate the non-productive ambiguity in current early building design decision making practice. The identified constraints and prioritized goals serve as inputs to the Scenarios Model. The RM also provides the formal value function for determining the value of design alternatives in the AAM and assists in decision-making.

The RM addresses need #1, and makes populating the Objective Space Size, Clarity, and Quality metrics in the proposed framework possible.

4.2 Scenarios Model (SM)

Methodology description

To enable building a well-defined alternative logic space and thus address needs #2 and #3, designers need to capture and communicate how they intend to address requirements parametrically. A prescriptive process model offers the means to do that. The SM is a prescriptive process model that builds on the *scenario* concept from Requirements Engineering. The authors' knowledge of the concepts that design teams currently use implicitly in the industry, as well research from Requirements Engineering (e.g., First Order Logic), provided the foundation for the SM elements. The enablers in the SM are designers, who begin the process of building the SM with the RM-established requirements. Building on the Concurrent Engineering framework, multiple designers concurrently decompose the same requirement into four inter-related levels of decision elements: *action items* → *strategies* → *parameters* → *parameter constraints.*

An *action item* is an actionable description of how to achieve a requirement. An action is generally addressed through multiple *strategies* - processes required to achieve an action. Both *actions* and *strategies* are decomposed into *parameters* - variables denoting properties impacting a design requirement. The last decision level is the *parameter constraint* - a fixed value or upper and lower limit and an increment that a parameter might be required to be within. When designers create multiple same-level decision elements (e.g., three action items for the same constraint), they specify how such decision elements relate to each other. AND/OR graphs widely used in Requirements Engineering can efficiently describe simpler relationships but are not efficient in more complex cases since this would lead to duplication of SM elements and result in model scalability issues. Instead, in SM designers use First Order Logic [lviii] to account for the more challenging logical conditions (Figure 5). First Order Logic formalisms generally describe a relation of inclusion [xii] represented in the SM as AND, OR, XOR logical gateways.

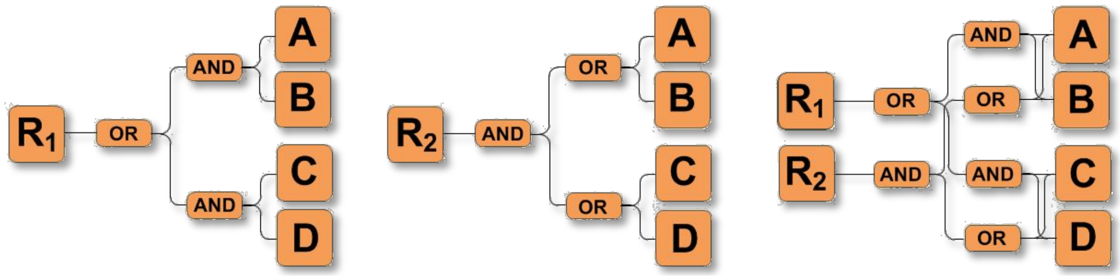


Figure 5: First Order logic implemented in the SM. R1, R2 are the requirement nodes generated in the SM; A, B, C, D represent the action, strategy, or parameter nodes in SM.

Some *actions* may conflict with *requirements*. Designers represent *potential conflicts* in SM that they identify either experimentally or based on expertise and intuition. For example, in addressing the goal of minimizing a building design construction cost, an experienced designer will see a *potential conflict* when choosing among several *strategies* for exterior wall systems that vary in cost. In such cases, designers draw a *potential conflict* arrow element from *action* or *strategy* decision element to the affected *requirement*(s). Identifying *potential conflicts* helps reduce the design space size by eliminating or mitigating conflicting *actions* and the dependent *strategies*, *parameters* and *parameter constraints*. Designers need to ensure that they provide enough decision nodes that do not result in conflicts to avoid prohibiting the development of a design.

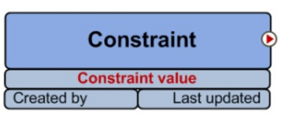
Designers distinguish between input and output parameters by drawing a *parameter dependency* arrow between two parameters. Designers assign concrete values to input parameters and build relationships (generally described as algebraic expressions in AEC design problems) driving the output parameter values. Output parameters have *parameter dependency arrows* pointing from input parameter nodes. *Parameter constraint* elements prescribe a range of values for the parent input parameter node and include the upper and lower extremes and the parameter increment.

Software Implementation

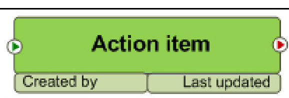
In Table 3 we describe the SM ontology, which includes several decision node types, and logical and process relationships among nodes that enable design stakeholders to generate and communicate multiple scenarios for the same design project. The system transfers constraints and goals between the RM and SM to serve as a starting point for designers to explicitly describe their logic to address requirements. To enable designers to determine parts of the SM that might be out of date, the mapping process captures any modifications in the RM as time stamps, which are reflected in the equivalent requirements nodes in the SM. The system enables designers to indicate the choice of scenarios by setting the decision node status: (a) *asserted* – indicates confident choice; (b) *retracted* – indicates a rejected choice. If designers retract a decision node, all other dependent decision nodes are retracted as well; (c) *assumed* – indicates uncertainty and a choice that might need revision. A retracted status propagates both down and upstream through SM arrows. The SM also enables designers to quantify the down and upstream dependencies for any node, and the number of requirements impacted by each action item to help determine high impact parameters.

Table 3: SM graphical notation, definitions, and data schema and in the Design Scenarios software.

Term notation / definition	Schema description

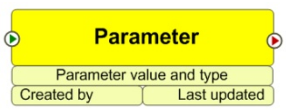

Restriction on the quantitative or qualitative value of a design parameter
	System inputs:
	`ENTITY Constraint`
	`name: STRING;`
	`value: STRING; (free-form)`
	`unit: STRING;`
	`created by: USER;`
	`last updated: DATE;`
	`node color: BLUE;`
	User inputs:
	`status: (ONEOF (Asserted, Retracted, Assumed));`
	`END_ENTITY;`

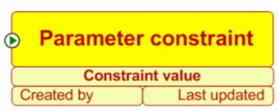

Quantifiable or qualitative value of a design parameter that is desirable to be achieved
	System inputs:
	`ENTITY Goal`
	`name: STRING;`
	`value: STRING; (free-form)`
	`unit: STRING;`
	`created by: USER;`
	`last updated: DATE;`
	`node color: BLUE;`
	User inputs:
	`status: (ONEOF (Asserted, Retracted, Assumed));`
	`END_ENTITY;`

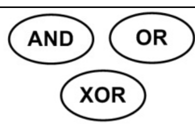
	**System inputs:**
	`ENTITY Action item`
	`created by: USER;`

An actionable description of how to achieve each requirement	`last updated: DATE;`
	`node color: GREEN;`
	`upstream dependencies: STRING;`
	`downstream dependencies: STRING;`
	User inputs:
	`ENTITY Action item`
	`description: STRING;`
	`status: (ONEOF (Asserted, Retracted, Assumed));`
	`END_ENTITY;`


A process required to achieve an action item
	System inputs:
	`created by: USER;`
	`last updated: DATE;`
	`node color: RED;`
	`upstream dependencies: STRING;`
	`downstream dependencies: STRING;`
	User inputs:
	`ENTITY Strategy`
	`description: STRING;`
	`status: (ONEOF (Asserted, Retracted, Assumed));`
	`END_ENTITY;`


A variable denoting properties that impact a design requirement
	System inputs:
	`created by: USER;`
	`last updated: DATE;`
	`node color: YELLOW;`
	`upstream dependencies: STRING;`
	`downstream dependencies: STRING;`
	User inputs:
	`ENTITY Parameter`
	`name: STRING;`
	`value: STRING; (free-form)`
	`type: STRING;`
	`status: (ONEOF (Asserted, Retracted, Assumed));`
	`END_ENTITY;`


A fixed value or range of values (shown as lower and upper limit nodes) that a parameter might be required to be within
	System inputs:
	`created by: USER;`
	`last updated: DATE;`
	`node color: YELLOW;`
	`upstream dependencies: STRING;`
	`downstream dependencies: STRING;`
	User inputs:
	`ENTITY Parameter constraint`
	`name: STRING;`
	`value: STRING; (free-form)`
	`END_ENTITY;`


Logical gates describing relationships between actions, strategies, and parameters.
AND – all on, OR – at least one on, XOR – at least one on and one off
	System inputs:
	`upstream dependencies: STRING;`
	`downstream dependencies: STRING;`
	User inputs:
	`ENTITY Logic gate`
	`function: (ONEOF (AND, OR, XOR));`
	`END_ENTITY;`

Enabling operation	**User inputs:** `ENTITY DS arrow`

arrow – denotes SM process operation.
 	
Parameter dependency arrow – distinguishes between intput and output parameters.
 	
Potential conflict arrow –illustrates conflicts between action items or strategies and constraints/goals
	`type: (ONEOF (Enabling operation, Parameter dependency, Potential conflict));`
	`connect: (Constraint AND Logical gate) OR`
	`(Goal AND Logical gate) OR`
	`(Logical gate AND Action item) OR`
	`(Logical gate AND Strategy) OR`
	`(Logical gate AND Parameter) OR`
	`(Parameter AND Parameter constraint)OR`
	`(Action item AND Parameter) OR`
	`(Parameter AND Parameter) OR`
	`(Action item AND Constraint) OR`
	`(Action item AND Goal)`
	`END_ENTITY;`

Inter-model transfer – connects and establishes a dependency of same requirement used in the RM and SM	**System inputs:**
	`ENTITY Inter-model transfer`
	`type: (Requirements Model to Scenarios Model);`
	`connect:(ONEOF (Constraint AND Constraint) OR`
	`(Goal AND Goal);`
	`END_ENTITY;`

The novel features introduced in the SM include the ontology for building the logical alternative spaces and the process of transferring SM parameters to the geometric alternative space. The SM enables populating the Total Option Space Size and Option Space Quality metrics in the proposed framework for measuring design space clarity and quality.

4.3 Parametric Process Model (PPM)

Methodology description

To enable building well-defined and comprehensive geometric alternative spaces, CAD experts need to determine, manage, and communicate how designers' scenarios, logic, and parameters are linked to geometry inside parametric models. This also entails addressing the CAD model technical issues, such as efficient navigation and management of large models. The PPM is both a descriptive and prescriptive process model that enables CAD experts to generate and communicate with 18 methods and 12 elements the logical construct and technical implementation of a chosen SM scenario in a parametric CAD model used to generate and search through alternative spaces (need #4). Concepts that CAD experts use to build parametric models provided the foundation for the PPM elements.

The PPM enablers are the CAD experts who connect the well-defined and comprehensive designers' logic to computable parametric models. The CAD experts begin with the SM-established parameters. The CAD experts refer back to the SM to understand the designers' scenario(s), logic, and decision choices and select the appropriate geometric elements (e.g., line, arc) for the identified scenario to which they link the corresponding input and output parameters. To enable predictable interaction with the parametric model, CAD experts constrain the geometric elements (e.g., tangency relation between two arcs). To create new geometry, CAD experts use CAD operations (e.g., extrude) and Reference elements (e.g., XY plane) to establish the CAD operation direction. CAD experts use the completed PPM to construct the parametric model, search through the alternative space delimited by the SM scenario (e.g., round building shape), evaluate the alternatives' performance against requirements that can be assessed geometrically through output parameters (e.g., building area), and extract the design alternatives that satisfy the geometry-based requirements for further analysis in discipline-specific tools (e.g., daylight, thermal comfort).

Software Implementation

In Table 4 we describe the PPM ontology, which includes several node types, and logical and process relationships among nodes. A PPM contains two levels of information abstraction. At the component-level the model illustrates the decomposition of the CAD model into components (e.g., floor plates, exterior wall) and their dependencies (e.g., exterior wall is dependent on floor plates). This is especially important when working with large CAD models that become overwhelming to manage if no decomposition is pursued. At the geometry-level, a PPM describes the composition of elements in each component. All nodes contain system-generated type-dependent attributes (e.g., Extrude CAD operation attributes include a Profile, Direction, and a Length value) that serve as prerequisite inputs for generating a node. Groups of geometry-level PPM nodes can be associated with or disassociated from a component. A node from one component can be cross-associated with another component. To help efficiently navigate large PPM models, CAD experts can “focus” a component to highlight the geometry-level grouped nodes that describe its composition and fade the rest.

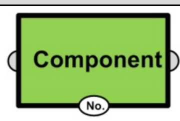
The system transfers input and output parameters between the SM and the geometry-level PPM to help link requirements-driven parameters to CAD models. The mapped parameter nodes serve as inputs to geometry nodes. CAD experts choose which geometry node types to use based on the actions and strategies captured by designers in the SM (e.g., Action → Generate building footprint; Strategies → Rectangular OR Round, which will lead to choosing either a Line or Arc attribute in the geometric element node and link either a Length or Radius parameter describing each strategy).

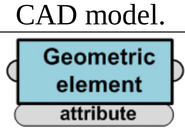
The PPM enables extracting input and output parameter nodes as tabular data and automates their generation in such parametric CAD tools as CATIA or Digital Project. The CAD expert manually builds the other PPM nodes in CAD following the PPM prescribed structure and dependencies.

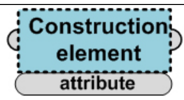
The novel features introduced in the PPM include the ontology for building the geometric alternative spaces and the process of transferring PPM parameters to the impact space. The PPM enables populating the CAD Model Clarity and Quality metrics in the proposed framework for measuring design space quality.

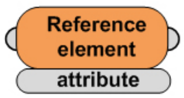
Table 4: PPM ontology and graphical notation.

Term notation / definition	**Schema description**

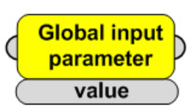

An information container describing a component-level decomposition of the CAD model.
	System inputs:
	`    node color: GREEN;`
	User inputs:
	`ENTITY Component`
	`    name: STRING;`
	`    label: STRING; (free form)`
	`    focus: (ONEOF (On, Off));`
	`END_ENTITY;`

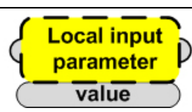

A predetermined geometric primitive used to create the geometric representation of the
	System inputs:
	`    set of attributes: SET OF STRINGS;`
	`    node color: BLUE;`
	User inputs:
	`ENTITY Geometric element`
	`  ABSTRACT SUPERTYPE OF (ONEOF (Point, Line, Circle,`
	`                    Spline Polyline, Arc, Ellipse));`
	`    name: STRING;`
	`    custom attribute: STRING; (free-form)`

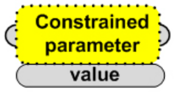
intended design	component association: (ONEOF (Associate, Disassociate); END_ENTITY;
 A geometric element used to construct the design intent but not explicitly featured in the final design representation	**System inputs:** identical to geometric element node type except node outline (dotted). **User inputs**: identical to geometric element node type.

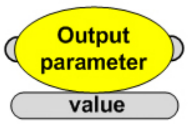

A plane of reference used to determine the orientation of the geometric elements
	System inputs:
	`set of attributes: SET OF STRINGS;`
	`node color: BROWN;`
	User inputs:
	`ENTITY Reference element`
	`ABSTRACT SUPERTYPE OF (ONEOF (Offset from plane, Parallel through point, Angle normal to plane, Through three points, Through two lines, Through point and line, Through planar curve, Normal to curve, Tangent to surface));`
	`name: STRING;`
	`component association: (ONEOF (Associate, Disassociate);`
	`END_ENTITY;`

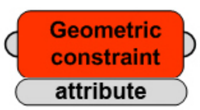

An action performed on geometric element(s) in a CAD model
	System inputs:
	`set of attributes: SET OF STRINGS;`
	`node color: GREEN;`
	`node shape: DIAMOND;`
	User inputs:
	`ENTITY CAD operation`
	`ABSTRACT SUPERTYPE OF (ONEOF (Project, Intersect, Extrude, Revolve, Offset, Fill, Loft, Blend, Join, Split, Translate, Rotate, Symmetry, Scale));`
	`custom attribute: STRING;`
	`name: STRING;`
	`component association: (ONEOF (Associate, Disassociate);`
	`END_ENTITY;`


A user-controlled parameter, which affects multiple geometric elements within a CAD model
	System inputs:
	`name: STRING;`*(if Parameter created in SM)*
	`value: STRING; (free-form)`
	`Node color: YELLOW;`
	User inputs:
	`ENTITY Global Input parameter`
	`name: STRING;`*(if New parameter)*
	`custom attribute: STRING;`*(if New parameter)*
	`component association: (ONEOF (Associate, Disassociate);`
	`END_ENTITY;`

 A user-controlled parameter, which affects a single	**User inputs:** identical to global input parameter. **System inputs:** identical to global input parameter node except node outline (dashed).

geometric element within a CAD model	
 An input parameter with a constrained value	**System inputs:** identical to global input parameter except node outline (dotted). **User inputs:** identical to global input parameter.


A parameter whose value is determined formulaically
	System inputs:
	`name: STRING;`*`(if Parameter created in SM)`*
	`value: STRING; (free-form)`*`(if created in SM)`*
	`node color: YELLOW;`
	`node shape: OVAL;`
	User inputs:
	`ENTITY Output parameter`
	`name: STRING;`*`(if New parameter)`*
	`value: input parameter 1 AND STRING (free-form)`
	`AND`
	`input parameter n;` *`(if New parameter)`*
	`component association: (ONEOF (Associate, Disassociate);`
	`END_ENTITY;`


A constant, non-numerical relationship between geometric elements
	System inputs:
	`Node color: RED;`
	User inputs:
	
	`ENTITY Geometric constraint`
	`ABSTRACT SUPERTYPE OF (ONEOF (Fixed, Horizontal, Vertical, Coincidence, Concentric, Perpendicular, Tangent, Parallel));`
	`set of attributes: SET OF STRINGS;`
	`name: STRING;`
	`component association: (ONEOF (Associate, Disassociate);`
	`END_ENTITY;`

Component sequencing – illustrates the sequence of construction process of components	**System inputs:**
	`start component name: STRING;`
	`end component name: STRING;`
	User inputs:
	`ENTITY PP Component arrow`
	`connect: Component AND Component;`
	`END_ENTITY;`

Input / output dependency – illustrates information dependency applicable to all but component nodes	**System inputs:**
	`start node name: STRING;`
	`end node name: STRING;`
	User inputs:
	`ENTITY PP arrow`
	`connect: (Geometric element AND Constraint) OR`
	`(Reference element AND Constraint) OR`
	`(Reference element AND CAD operation)`
	`OR`
	`(Reference element AND Geom. element)`
	`OR`
	`(Input parameter AND Geom. element) OR`
	`(Output parameter AND Geom. element) OR`
	`(Input parameter AND Output parameter)`

	`OR`
	`        (Input parameter AND CAD operation) OR`
	`        (Output parameter AND CAD operation) OR`
	`        (Input parameter AND Output parameter)`
	`END_ENTITY;`

Inter-model transfer – connects and establishes a dependency of the same parameter in the SM and PPM	**System inputs:**
	`ENTITY Inter-model transfer`
	`      type: Scenarios Model to Parametric Process Model;`
	`      connect (ONEOF (Input Parameter AND Input Parameter) OR (Output parameter AND Output parameter);`
	`END_ENTITY;`

4.4 Alternatives Analysis Model (AAM)

Methodology description

The AAM is a tabular model developed to evaluate how each alternative analyzed in parametric CAD or discipline-specific tools ranks in relation to the goals identified in the RM, thus enabling building the impact and value spaces and addressing need #8. MACDADI provided the foundation for the AAM elements.

The enablers in the AAM are the designers. DS method asks designers to perform a formal analysis (e.g., daylight) for every parametrically generated alternative and determine a well-defined impact score given the RM constraints and goals. A major benefit of DS method is that designers are ensured to perform analysis only on alternatives that satisfy all the geometry-based requirements. Benchmark-based scoring enables designers to determine and compare the impact of each alternative's performance against the RM goals' targets, calculated as a percentage of the goal target value. Designers assign scores measured in percentage points to each alternative based on low and high benchmarks (e.g., high benchmark: minimize cost to $80,000, low benchmark: minimize to $100,000). If an alternative achieves a goal, it receives a 100% score. If it exceeds it (e.g., $70,000, it receives the percentage scored above the high benchmark – 112.5%, etc.) Benchmark values vary for each requirement and are determined in the RM by the stakeholder who proposes the requirement.

To determine the final multidisciplinary performance value of each alternative, the DS method multiplies the impact score for each goal with the appropriate goal importance score transferred from the RM and sums these into a final value score.

Software Implementation

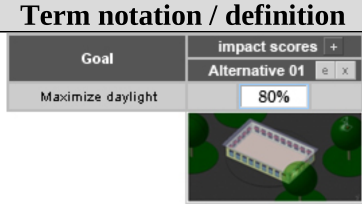
In Table 5 we describe the AAM ontology. The AAM consists of user-generated and system-generated inputs. The former includes goal impact scores and parameter values for each analyzed alternative. The latter includes alternatives' value scores for each goal and the total value score for each alternative.

The system transfers the goals between the RM and AAM, and the input and output parameters between the PPM and AAM as shown with the horizontal arrows in Fig. 4 and with the last definition in Table 5. Users input the alternatives that were analyzed and need to be scored and upload the alternatives' geometric previews.

The AAM offers multidisciplinary design teams a formal unifying structure and communication tool for describing and comparing the quantitative and qualitative analyses of design alternatives to enable improved decision-making. The AAM enables populating the Value Space Size and Clarity metrics in the proposed framework.

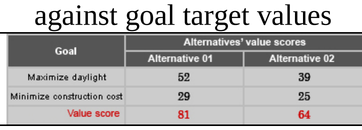
Table 5: AAM ontology and graphical notation.

Term notation / definition	Schema description


Alternative impact score determines the percentage of the goal target value.
	System inputs:
	`goal name: ARRAY OF STRING;`
	User inputs:
	`ENTITY Impact score`
	`alternative score: ARRAY OF STRING; (free-form)`
	`alternative preview: ARRAY OF IMAGE;`
	`END_ENTITY;`


Design parameter illustrates the parameter(s) and the value(s) used in generating a design alternative
	System inputs:
	`ENTITY Design parameter`
	`parameter name: ARRAY OF STRING;`
	`parameter alternative value: ARRAY OF STRING; (free-form)`
	`END_ENTITY;`


Alternative illustrates the design alternative(s) that satisfy constraints and selected to be analyzed against goal target values
	System inputs:
	`ENTITY Alternative`
	`alternative name: ARRAY OF STRING;`
	`END_ENTITY;`

	**System inputs:**
	`ENTITY Value score`
	`goal name: ARRAY OF STRING;`

Alternative value illustrates the score calculated by multiplying the alternative impact score and goal importance. **Value score** illustrates the sum of alternative's impact scores for all goals	`goal importance: ARRAY OF STRING; (free-form)`
	`goal alternative value score: STRING;(free-form)`
	`alternative value score: ARRAY OF STRING; (free form)`
	`END_ENTITY;`

Inter-model transfer – connects and establishes a dependency of the same goal or parameter in RM, PPM and AAM models	**System inputs:**
	`ENTITY Inter-model transfer`
	`type: (ONEOF (Requirements Model to Alternatives Analysis)OR (Parametric Process Model to Alternatives Analysis Model);`
	`connect (ONEOF (Goal AND Goal) OR (Input Parameter AND Input Parameter) OR (Output parameter AND Output parameter);`
	`END_ENTITY;`

4.5 Illustrative Example

This section explains the application of DS through a simple, hypothetical example that has three stakeholders – client (a University), architect, and mechanical engineer. Figure 6 illustrates the site between the four central planters in the University Quad, where a new teaching space is to be designed.

Figure 6: The site for a teaching space.

4.5.1 Requirements Model

The design process begins with the analysis of the site opportunities and constraints. The client creates a constraint in the RM – *minimum usable area* of 3,000 square feet. The site helps the architect identify an additional constraint – the *buildable area set back* of 10 feet from the four adjacent planters to allow circulation around the building. The mechanical engineer adds the goal to *maximize the use of daylight* to an average of 500 lux required for a teaching space and thus minimize the use of electric lighting. The client adds the goal to *minimize the construction cost* to below $100,000.

Once the requirements are synthesized and accepted by all parties, stakeholders and designers individually rank each goal according to their preference. All stakeholders are weighted equally in this example. Figure 7 illustrates the client's preference for minimizing construction cost by assigning a 60% relative importance value. When stakeholders complete assigning importance to goals, the system generates a cumulative importance percentage graph. By comparing weighted characteristics of goals, design teams can set priorities. *Maximizing use of daylight* emerged as the prevailing goal with a 65% cumulative percentage score. The RM helps clarify what the project requirements are, who generated them, and how important are they to the project stakeholders in an integrated, concurrently generated model.

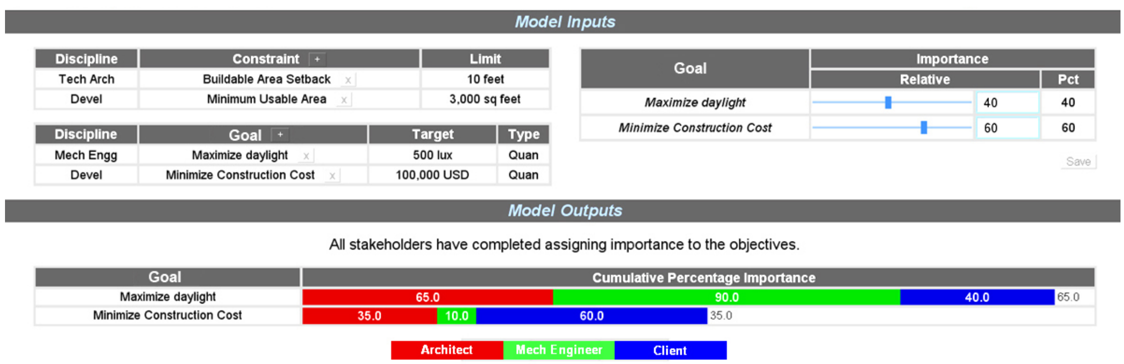


Figure 7: The Requirements Model captures the stakeholders' constraints, goals, and preferences for goals. Stakeholders distribute a percentage of preference (totaling 100%) to each identified goal.

4.5.2 Scenarios Model

Establishing a scenario enables the designers (architect and mechanical engineer) to determine the alternative space extremes based on the identified set of requirements. The architect suggests investigating two scenarios – one single large teaching space configuration and two smaller ones, both with perimeter windows to address the daylight goal. This decision clarifies the range of geometric variations – from a square to a rectangle (Figure 8).

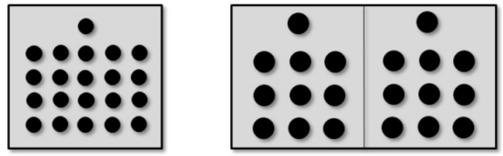


Figure 8: The architect suggests two scenarios (square and rectangular classroom) that enables determine the desired range of geometric variations.

Figure 9 shows the case study SM model. The model starts with the goal and constraints nodes mapped from the RM and is concurrently decomposed by the design stakeholders into action, strategy, parameter, and parameter constraint decision nodes. The design space extremes are explicitly recorded as strategy nodes (e.g., rectangular OR square footprint – both extremes need to be supported by the CAD model), which in turn describe the *Control building configuration* action, one of the two actions required to achieve the *Minimum usable area* constraint. Both strategies share the same set of geometric parameters – *Building width* (input) and *Building length* (output). Using his expert knowledge on minimum usable building width, the architect suggests a range decribed by two parameter constraints – lower limit of 30 feet, and upper limit of 95 feet, calculated by using Pythagora's theorem in view of the round site configuration.

In addressing the *Maximize use of daylight* goal, *Introducing lightshelves* – one of the five required actions, is identified as leading to a potential conflict with the *Minimize construction cost* goal which is important for the client. Further decomposing the action into strategies helps determines how to avoid the negative impact. For example, two strategies impact the geometry, the third suggests a material. Having the *Same depth on all sides* is a less costly solution than *Orientation dependent depth* strategy, which results in a higher number of custom building components, and thus is the chosen strategy. The strategy that wasn't chosen along with the subsequent dependent nodes is faded by the system and kept as a reference in case stakeholders change their preferences in the Requirements Model.

The SM enables design stakeholders to concurrently simplify complex design decisions by visualizing each others' logic and the repercussions, and identify key design parameters used to generate a requirements-driven logical alternative space.

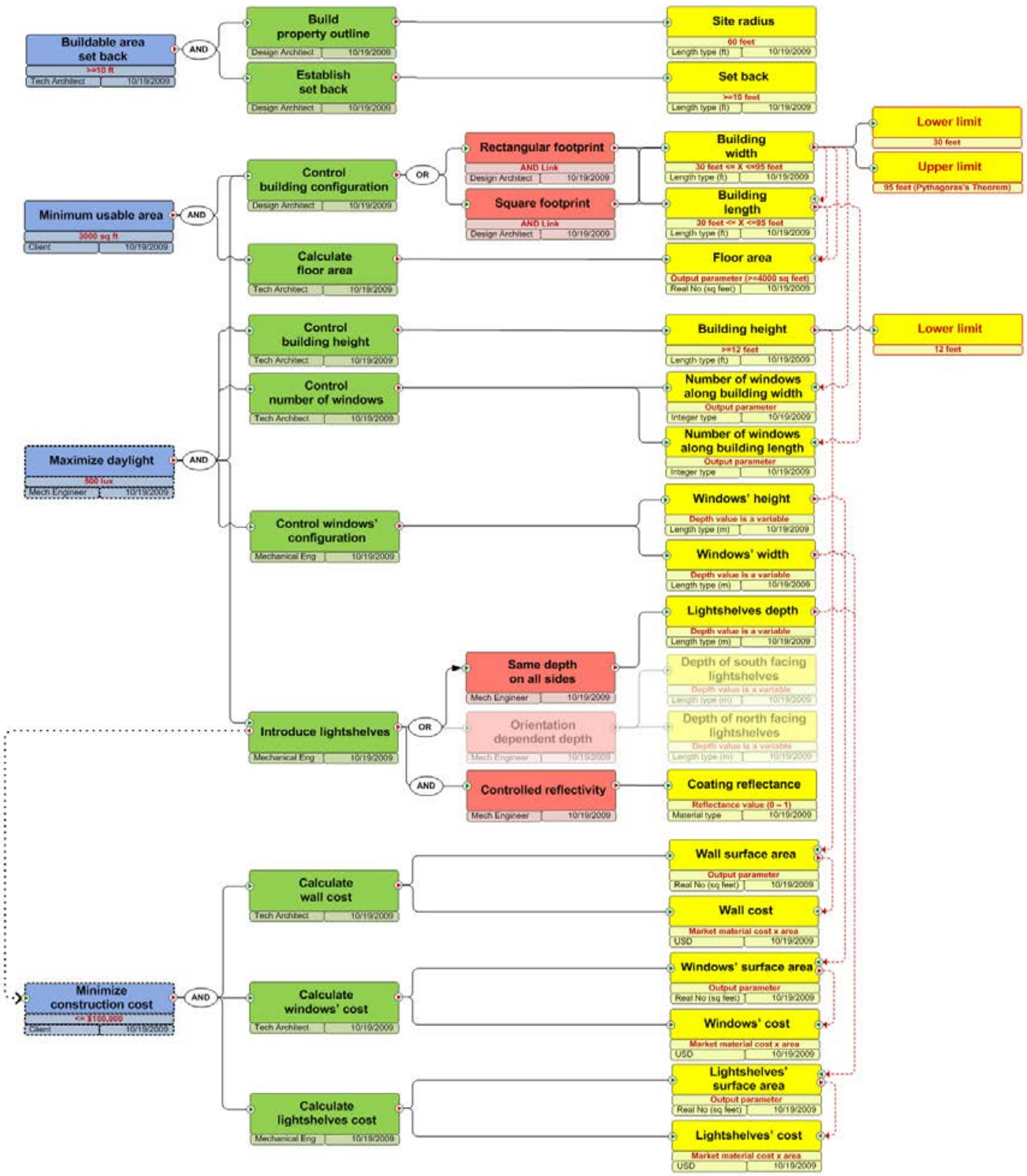


Figure 9: Scenarios Model for the University Quad illustrative example. The model starts with the two Constraints and two Goals transferred from the RM and design stakeholders rationalize them into Actions, Strategies, Parameters and Parametric Constraints. AND, OR, XOR logical gateways are used to describe relations between Actions, Strategies, and Parameters.

4.5.3 Parametric Process Model

With the SM finalized, the system transfers the input and output parameters into the PPM environment. The CAD expert is notified by the project administrator to begin building the PPM used to generate the geometric alternative space. He first examines the SM to understand the design stakeholders' logic for addressing requirements and the scenario(s) to be implemented in a CAD model. He begins building the PPM by decomposing the CAD model structure into six components created in the component-level PPM model space (Figure 10). In today's practice, this step is generally fraught with errors and leads to likely rework because a comprehensive RM/SM is missing. The CAD expert organizes the nodes to reflect the sequence in the model building process and the inter-component dependency. For example, in order to generate the *Windows*, the model must first have the *Walls* component constructed, which in turn is dependent on the *Building footprint* component.

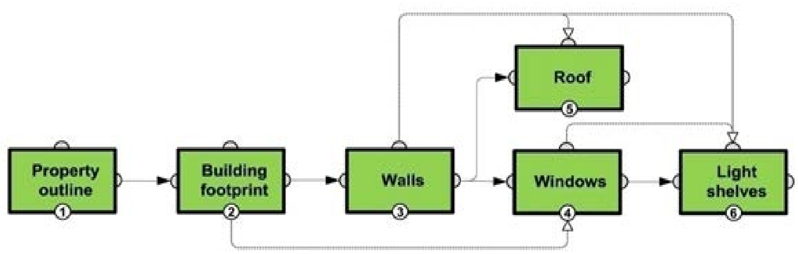


Figure 10: Component-level PPM illustrates the CAD model decomposition into six components shown in hierarchical order.

With the component-level structure of the parametric model in place, the next task is to determine the composition of each component at the schema level. For example, in describing the model's first component called *Property outline*, the CAD expert referenced the RM model that helped identify the site's circular configuration and its radius constraint. As a result, a circle was used as a starting point in building the geometry-level PPM. To help fix the circle in the work space, its origin was coincidentally constrained to the origin of the XY plane, and its radius was determined by the *Property radius* parameter, which includes the 10'-0" set back constraint (Property radius → 60'–10'=50') (Figure 11a).

To construct the *building footprint* component the CAD expert chooses a rectangle as the geometric element given the scenarios prescribed in the SM (i.e., square to rectangle). He then assigns a geometric constraint (i.e., coincident) that binds the rectangle's first three vertices to the circle outline, and connects with a dependency arrow the SM transferred *Building width* global input parameter to the vertical line on the rectangle's right hand side. The *Building length* output parameter and its value is dynamically measured after the CAD expert links it to the rectangle's horizontal line. This enables calculating the *Minimum usable area* constraint through the *Floor area* output parameter by multiplying the *Building length* and *Building width* parameters (Figure 11b). To prevent emergence of unpredictable geometry (e.g., changing the length of the rectangle that has not been geometrically constrained may lead to a parallelepiped), the pairs of lines are assigned vertical and horizontal geometric constraints.

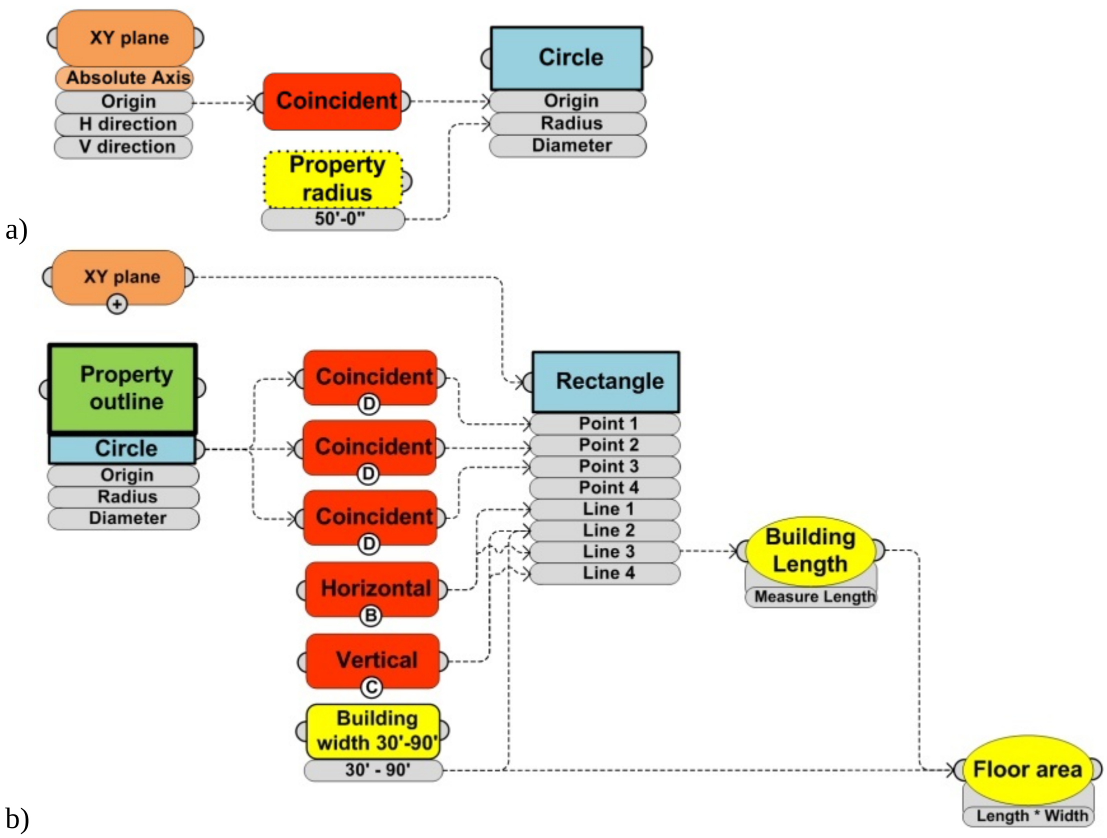


Figure 11: Schema-level PPM describes the composition of: a) "Property outline" component; b)"Building footprint" component.

A similar method is used to construct the model's remaining four components. Figure 12 illustrates the final composite PPM model, which helps understand the implications of changing the value of any input parameter on the rest of the model. CAD experts can navigate through the model by selecting component nodes and bringing into focus at the geometry-level only the nodes that are grouped into that component.

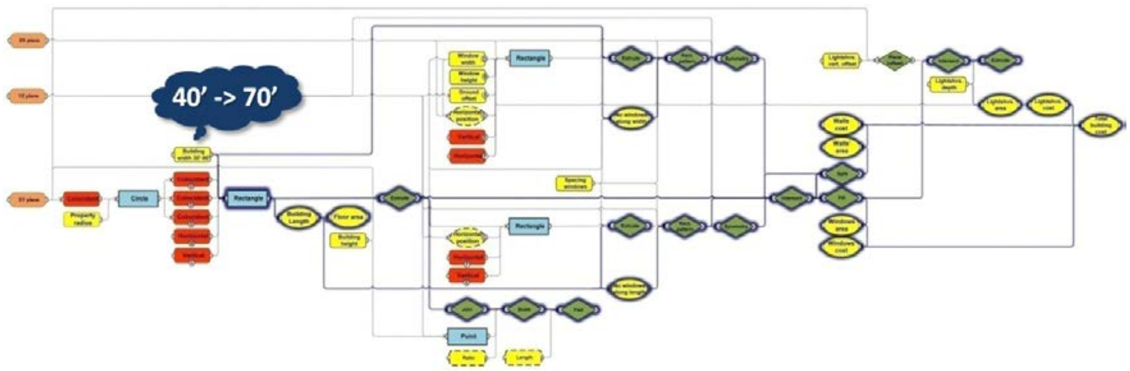
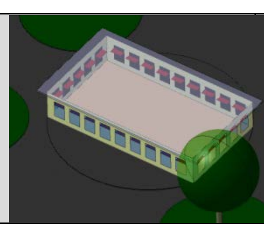
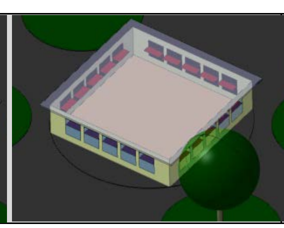
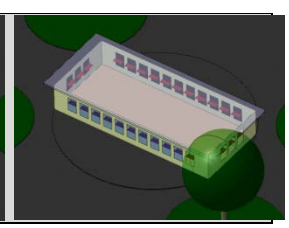


Figure 12: Final composite geometry-level PPM. Note that the nodes' attributes are toggled off to help simplify the model. The model helps understand which nodes are affected when parameter values are changed by highlighting them (i.e., the Building width value was changed from 40' to 70').

The completed PPM serves as a guideline to building the parametric CAD model in which design stakeholders manage the SM parameters to generate geometric design alternatives. Table 6 illustrates three such alternatives from potentially an infinite number that the CAD model can support within the SM-defined alternative space. A selection of model definition parameters is also included. Designers selected these alternatives for further analysis following both a qualitative (e.g., visual) and quantitative assessment, in which they use the *Floor area* output parameter to understand which alternatives satisfy the 3,000 ft^2 *Minimum usable area* constraint.

Table 6: Three geometric alternatives selected for further analysis and the input and resulting output parameters.

	Alternative 1	Alternative 2	Alternative 3
Floor Area (Constraint) - 3000 ft^2	**4,330**	**5,000**	**3,666**
Constr. Cost (Goal) - $<100,000	**112,809**	**130,655**	**139,250**
Building Height (ft)	**16**	**18**	**18**
Building width (ft)	**50**	**70**	**40**
Window Height (ft)	**9**	**10**	**8**
Window Width (ft)	**6**	**9**	**5**
Window Spacing (ft)	**3**	**2**	**2**
Light shelves Depth (ft)	**3**	**4**	**2.5**

4.5.4 Alternatives Analyses Model

Building the AAM enables the design stakeholders to make an objective decision amongst the three alternatives based on how well each alternative satisfies the value function established in the RM. First, the system transfers the RM goals and the PPM input and output parameters into the AAM model environment. Next, design stakeholders add the three alternatives to the project database, for which they record the parameter values used to generate them and the impact scores for each goal. Similar to the quantitative assessment of constraints in CAD used to select the three alternatives, the design stakeholders determine the impact scores of each selected alternative for the *Minimize construction cost* below $100,000 goal, calculated as the sum of *Window cost, Wall cost,* and *Light shelves cost* output parameters. For example, alternative 1 cost $112,809 or 17% above the goal target value and receives a score of 83%. Changing any of the input parameters affects the performance of this goal.

Not all goals, however, can be calculated by means of output parameters in CAD. Some require model-based analyses in specialized tools. For example, to address the *Maximize use of daylight* goal, the mechanical engineer optimizes the geometry of the selected alternatives (e.g., mesh) to perform daylight analyses in Autodesk Ecotect (Figure 13).

a) b) c)

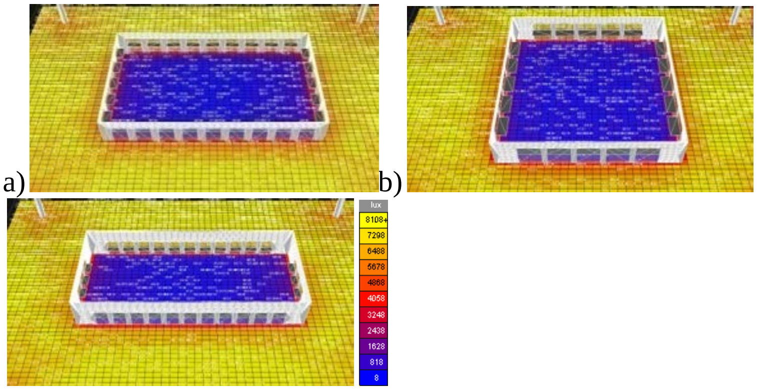
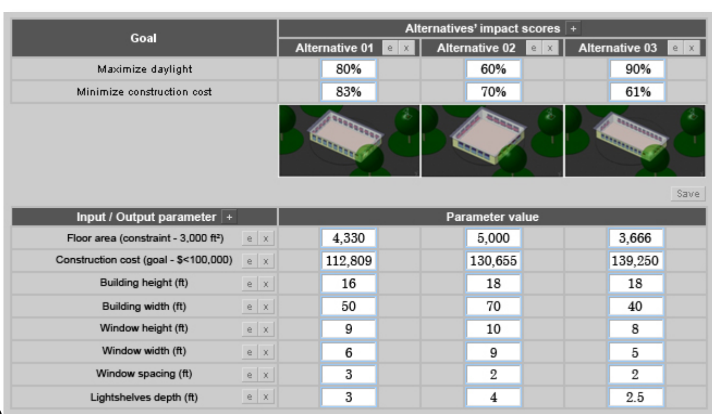


Figure 13: Some quantifiable goals require model-based analysis performed outside the parametric modeler. Autodesk Ecotect© is used to determine average daylight values in lux for all three alternatives. Note that the ceiling is omitted for clarity.

This enables extracting average daylight values and assigning impact scores for *Maximize daylight use* goal. For example, alternative 1 has an average of 400 lux, or an 80% score of the target value of 500 lux (400*100/500=80%); alternative 2 → 300 lux, or a 60% score (300*100/500=60%), and alternative 3 → 450 lux, or a 90% score (450*100/500=90%). All global input parameters listed in Table 6 impact this goal.

Once the stakeholders assign all the impact scores, the system generates the value scores for each alternative by multiplying the impact scores with the cumulative percentage importance scores of each goal from the RM. For example, alternative 1 value score multiplies 80*0.65=52. The system aggregates value scores for each alternative into a total value score (e.g., alternative 1 → 52+29=81) to enable the stakeholders to objectively select the highest performing alternative 1 (Figure 14b).



Goal	Alternatives' impact scores		
	Alternative 01	Alternative 02	Alternative 03
Maximize daylight	80%	60%	90%
Minimize construction cost	83%	70%	61%

Input / Output parameter	Parameter value		
Floor area (constraint - 3,000 ft²)	4,330	5,000	3,666
Construction cost (goal - $<100,000)	112,809	130,655	139,250
Building height (ft)	16	18	18
Building width (ft)	50	70	40
Window height (ft)	9	10	8
Window width (ft)	6	9	5
Window spacing (ft)	3	2	2
Lightshelves depth (ft)	3	4	2.5

a)

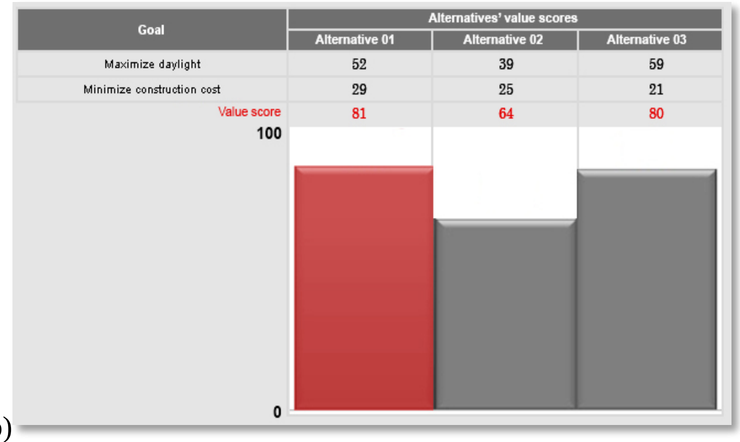
b) 

Figure 14: Alternatives Analysis Model: a) users input impact scores and geometric parameter values for each analyzed alternative; b) the system generates a value score for each alternative. Alternative 1 emerges as the preferred one based on the goals identified in the RM and the goal importance determined by stakeholders.

4.5.5 Testing the practical value of DS

We tested the practical value of Design Scenarios methodology on an industry project – the concept design of a high-rise building in Saudi Arabia by a leading Architecture Engineering firm (Figure 15). We present the findings in [lix]. When compared to the traditional conceptual design process, the results indicate an order of magnitude improvement across several metrics for measuring design space quality:

(1) *Objective space quality* – the project requirements were elicited from all project stakeholders and served in building a scenario-specific design space;

(2) *Option space size* –DS enabled quantification of the option space size with over 1,100 options versus an average of 3 that design team generate with the traditional conceptual design process;

(3) *Alternative space size* – the CAD expert generated and analyzed 10 alternatives versus an average of 3 with the traditional conceptual design process;

(4) *Alternative space clarity* – designers developed two well-defined scenarios and the relevant input and output parameters;

(5) *CAD model clarity* – the parametric CAD model structure was explicit and transparent;

(6) *CAD model quality* – the CAD expert built one CAD model per scenario used to generate the alternative space versus multiple CAD models required in the traditional process when requirements were not met;

(7) *Impact space size* – designers performed a formal analysis for each requirement;

(8) *Value space size* – designers determined a total value for each generated alternative versus a lacking valuation method in the traditional process;

(9) *Value space clarity* –designers explicitly defined a total value for the generated alternatives.

While Design Scenarios was tested on a single industry project, we believe DS can have a similar impact and power on other AEC projects and industries that undergo analogous process problems.

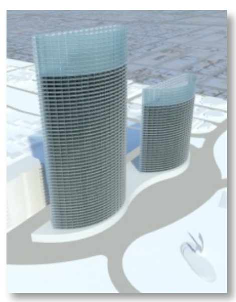


Figure 15: Jeddah mixed-use towers project in Saudi Arabia.

5 Conclusion

The process of designing a building requires the expertise of many disciplines. In practice today, domain experts tackle the same problem with different sets of requirements, ontologies, and work methods. The performance of a design project is, therefore, not only a function of the expertise of the individual experts, but also how well they work together [lx]. This is especially true during conceptual design, when decisions are the cheapest to make for design stakeholders and have biggest impact on the design cost and performance [lxi]. A system that offers a common ontology and work process for all stakeholders in the building design process can positively impact the design space quality.

This paper first introduced a framework for measuring design space quality through a set of key metrics for the Objective, Alternative, Impact, and Value subspaces. The paper's major contribution is an integrated methodology called Design Scenarios to improve the quality of the design spaces AEC teams are able to construct. The novel features include the ontology and methods for building multidisciplinary logical and geometric alternative spaces, as well as the process of systematically transferring design requirements from the objective space to the alternative, impact and value spaces through the means of four interrelated models.

With the Requirements Model stakeholders explicitly determine project requirements in a unified model. With the Scenarios Model, design stakeholders rationalize the requirements transferred from the RM into logical alternative spaces described in terms of scenarios and key input and output parameters. With the Parametric Process Model CAD experts explicitly communicate the construct of parametric CAD models, in which the parameters transferred from the SM are used to generate geometric alternative spaces within the boundaries of the SM-defined scenario. Finally, with the Alternatives Analysis Model stakeholders determine the impact and total value scores for the generated alternatives to enable an objective decision making process.

This paper described the application of Design Scenarios through an illustrative example. In [lix], we describe the application of Design Scenarios on an industry case study, and compare the resulting design space quality metrics with three other data sets: traditional, non-parametric conceptual design practice, and two applications of parametric modeling with no formal methodology to generate and communicate scenarios. The DS methodology was tested on two scenarios to illustrate its generality across scenarios. However, the current limitation is that one researcher implemented the methodology and performed the measurements from one industry case. Future work is to extend external validity by applying DS on more industry projects. DS offers opportunities to further automate the parametric CAD model generation from the PPM and to integrate the AAM with multidisciplinary optimization tools to automate the process of determining the impact scores for alternatives. The industry application of DS provides evidence that DS enables design teams with well-defined elements and methods for addressing requirements by creating parametric alternatives with clear multi-objective values that can potentially provide clients with better building designs.

6 Bibliography

[i] C. Clevenger, C., J. Haymaker, J., Metrics to Assess Design Guidance, Design Studies, In Press, doi:10.1016/j.destud.2011.02.001, 2011.

[ii] R. Sutton, Weird Ideas that Work - 11.5 Practices for Promoting, Managing, and Sustaining Innovation, The Free Press, New York, NY, 2002.

[iii] Ö. Akın, Variants of Design Cognition, Design Knowing and Learning: Cognition in Design Education. C. Eastman, W. Newsletter, & M. McCracken, Eds., New York: Elsevier, pp. 105–124, 2001.

[iv] C. Bock, C., X. Zha, H. Suh, J. Lee, Ontological Product Modeling for Collaborative Design, Journal of Advanced Engineering Informatics, Vol. 24, pp. 510–524, 2010.

[v] H. Simon, The Sciences of the Artificial. Cambridge: MIT Press, 1969.

[vi] E. Baniassad, S. Clarke, Theme: An Approach for Aspect-Oriented Analysis and Design, 26th International Conference on Software Engineering, pp.158-167, 2004.

[vii] S. Giesecke, W. Hasselbring, M. Riebisch, Classifying Architectural Constraints as a Basis for Software Quality Assessment, Advanced Engineering Informatics, Vol. 21, pp. 169–179, 2007.

[viii] C. Alexander, S. Ishikawa, M. Silverstein, A Pattern Language: Towns, Buildings, Construction, Center for Environmental Structure, vol. 2, Oxford University Press, New York, 1977.

[ix] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley Professional, 1995.

[x] F. Buschmann, R, Meunier, H. Rohnert, P. Sommerlad, M. Stal, Pattern-Oriented Software Architecture: A System of Patterns, John Wiley & Sons, 1996.

[xi] T. Moran, J. Carroll, Design Rationale: Concepts, Techniques, and Use, Lawrence Erlbaum Associates, Inc., Publishers, 1996.

[xii] R. Stouffs, Constructing Design Representations Using a Sortal Approach, Advanced Engineering Informatics, Vol. 22, pp. 71–89, 2008.

[xiii] R. Woodbury, A. Burrow, Whither Design Space? Artificial Intelligence for Engineering Design, Analysis and Manufacturing, Vol. 20, pp. 63-82, 2006.

[xiv] G. Goldschmidt, Quo Vadis, Design Space Explorer? Artificial Intelligence for Engineering Design, Analysis and Manufacturing, Vol. 20, pp. 105-111, 2006.

[xv] V. Gane, J. Haymaker, Benchmarking Current Conceptual High-rise Design Processes, ASCE Journal of Architectural Engineering. Vol. 16, No. 3, pp. 100-111, 2010.

[xvi] J. Chachere, J. Haymaker, Framework for Measuring Rationale Clarity of AEC Design Decisions, Journal of Architectural Engineering, In Press, doi: 10.1061/ (ASCE) AE.1943-5568.0000036, 2011.

[xvii] C. Struck, P de Wilde, C. Hopfe, J. Hensen, An Investigation of the Option Space in Conceptual Building Design for Advanced Building Simulation, Advanced Engineering Informatics, Vol. 23, pp. 386–395, 2009.

[xviii] M. Lin, L. Chen, M. Chen, M. An Integrated Component Design Approach to the Development of a Design Information System for Customer-oriented Product Design, Advanced Engineering Informatics, Vol. 23, pp. 210–221, 2009.
[xix] H. Zhang, H. Wang, D. Chen, G. Zacharewicz, G. A Model-Driven Approach To Multidisciplinary Collaborative Simulation For Virtual Product Development, Advanced Engineering Informatics, Vol. 24, pp. 167–179, 2010.
[xx] G. Colombo, A. Mosca, F. Sartori, Towards The Design Of Intelligent CAD Systems: An Ontological Approach, Advanced Engineering Informatics, Vol. 21, pp. 153–168, 2007.
[xxi] C. Rolland, C. Salinesi, Modeling Goals and Reasoning with Them, Engineering and Managing Software Requirements, Eds. A. Aurum, C. Wohlin, Springer Verlag, 2005.
[xxii] W. Hsu, I.M. Woon, I. M, Current Research in the Conceptual Design of Mechanical Products, Computer-Aided Design, 30, 1998, pp 377-389, 1998.
[xxiii] A. Kiviniemi, M. Fischer, V. Bazjanac, B. Paulson, PREMISS - Requirements Management Interface to Building Product Models: Problem Definition and Research Issues, CIFE Technical Report No. 92, Accessed at http://cife.stanford.edu/online.publications/TR92.pdf, 2004.
[xxiv] C. Kam, Dynamic Decision Breakdown Structure: Ontology, Methodology, & Framework for Information Management in Support of Decision-Enabling Tasks in the Building Industry, CIFE Technical Report No. 164. Accessed at http://cife.stanford.edu/online.publications/TR164.pdf, 2005.
[xxv] L. Miller, Concurrent Engineering Design: Integrating The Best Practices For Process Improvement, Society of Manufacturing Engineers. Publications Development Department, Reference Publications Division, 1993.
[xxvi] G. Xiong, Theory and Practice of Concurrent Engineering, Tsing Hua Press, Beijing, 2000.
[xxvii] S. Fukuda, Be Lazy: A Motto for New Concurrent Engineering, Springer London, 2007.
[xxviii] A. Yassine, D. Braha, Complex Concurrent Engineering and the Design Structure Matrix Method, Sage Publications, Vol. 11, No. 3, pp. 165-176, 2003.
[xxix] J. Hauser, D. Clausing, The House of Quality, Harvard Business Review. Vol. May-June, pp. 63-73, 1988.
[xxx] S. Takai, S., K. Ishii, Integrating Target Costing Into Perception-Based Concept Evaluation of Complex and Large-Scale Systems Using Simultaneous Decomposed QFD, Journal of Mechanical Engineering, Vol. 128, Issue 6, pp. 1186-1196, 2006.
[xxxi] M. Leary, C. Burvill, Enhancing the quality function deployment conceptual design tool, ASME Journal of Mechanical Design, Vol. 129, Issue 7, pp. 701-709, 2007.
[xxxii] D. Chen, W. Pai, A Methodology for Conceptual Design of Mechanisms by Parsing Design Specifications, Journal of Mechanical Engineering, Vol. 127, Issue 6, pp. 1039-1045, 2005.
[xxxiii] J. Haymaker, J. Chachere, Coordinating Goals, Preferences, Options, and Analyses for the Stanford Living Laboratory Feasibility Study, Intelligent Computing in Engineering and Architecture 13th EG-ICE Revised Selected Papers, Lecture Notes in

Computer Science, Ed. I. Smith, Springer-Verlag, Berlin, Heidelberg, New York, Vol. 4200/2006, pp. 320-327, 2006.

[xxxiv] P. Laplante, What Every Engineer Should Know about Software Engineering, CRC Press, Taylor & Francis Group, 2007.

[xxxv] C. Rolland, C. Pernici, A Comprehensive View of Process Engineering, Proceedings of the 10th International Conference CAiSE'98, B. Lecture Notes in Computer Science 1413. Pisa, Italy: Springer, pp. 191, 1998.

[xxxvi] D. Ross, E. Schoman, Structured Analysis for Requirements Definition, IEEE Transactions on Software Engineering, Vol. SE-3, No. 1. Pp. 6-15, 1977.

[xxxvii] A. Lamsweerde, Goal-Oriented Requirements Engineering: A Guided Tour, Proceedings RE'01, 5th IEEE International Symposium on Requirements Engineering, Toronto, pp. 249-263, 2001.

[xxxviii] N. P. Suh, Axiomatic Design Theory for Systems, Research in Engineering Design Vol. 10, pp. 189–209, Springer-Verlag London Limited, 1998.

[xxxix] N. P. Suh, Axiomatic Design of Mechanical Systems, Journal of Vibration and Acoustics. Vol. 117, pp. 2-10, 1995.

[xl] G. Xexe, A. Vivacquaa, J. Moreira de Souzaa, B. Bragaa, J. D'Almeida Jr, B. Kinder Almenteroa, R. Castilhoa, B. Miranda, COE: A collaborative ontology editor based on a peer-to-peer framework, Advanced Engineering Informatics, Vol. 19, pp. 113–121, 2005.

[xli] National Institute of Standard and Technology (NIST), IDEF0 - Function Modeling Method, Accessed on http://www.idef.com/IDEF0.htm, 1993.

[xlii] M. Fowler, UML Distilled Third Edition – A Brief Guide to Standard Object Modeling Language, Pearson Education, 2004.

[xliii] T. Froese, Models of Construction Process Information, ASCE Journal of Computing in Civil Engineering, Vol. 10, No. 3, pp. 183-193, 1996.

[xliv] G. Lee, R. Sacks, C. Eastman, Product Data Modeling Using GTPPM – A case study, Automation in Construction, Vol. 16, pp. 392 – 407, 2007.

[xlv] D. Tapping, T. Shuker, Value Stream Management, Productivity Press, 2002.

[xlvi] J. Haymaker, J. Kunz, B. Suter, M. Fischer, Perspectors: Composable, Reusable Reasoning Modules To Construct an Engineering View from Other Engineering Views, Advanced Engineering Informatics, Vol. 18/1, 49-67, 2004.

[xlvii] T. Knight, Shape Grammars In Education And Practice: History And Prospects, International Journal of Design Computing, Vol 2. Accessed at: http://www.arch.usyd.edu.au/kcdc/journal/vol2, 2000.

[xlviii] G. Rozenberg, Handbook on Graph Grammars and Computing by Graph Transformation: Foundations, Vol.1, World Scientific Publishing Company, 1997.

[xlix] AIAA, Current State Of The Art In Multidisciplinary Design Optimization, American Institute for Aeronautics and Astronautics Inc. MDO Technical Committee, 1991.

[l] D. Kelley, Design Thinking, Accessed at http://www.extrememediastudies.org/extreme_media/1_navigating/pdf/navigating_design_thinking.pdf, 2006.

[li] J. Shah, M. Mäntylä, Parametric and Feature-Based CAD/CAM: Concepts, Techniques, and Applications, Wiley, John & Sons, Inc., 1995.

[lii] H. Pottmann, A. Asperl, M. Hofer, A. Kilian, Architectural Geometry, Bentley Institute Press, 2007.
[liii] B. Kolarevic, Architecture in the Digital Age: Design and Manufacturing, Taylor & Francis, 2003.
[liv] S. Reddy, K. Fertig, D. Smith, Constraint Management Methodology for Conceptual Design Tradeoff Studies, Proceedings of the 1996 ASME Design Engineering Technical Conferences and Computers in Engineering Conference, Irvine, California, pp 1-11, 1996.
[lv] E. Kinzel, J. Schmiedeler, G. Pennock, G., Function Generation With Finitely Separated Precision Point Using Geometric Constraint Programming, Journal of Mechanical Engineering, Vol. 129, Issue 11, pp. 1185-1191, 2007.
[lvi] G. Lee, R. Sacks, C. Eastman, Specifying Parametric Building Object Behavior (BOB) For A Building Information Modeling System, Automation in Construction, Vol. 15, pp. 758 – 776, 2006.
[lvii] ISO, International Standard 10303-11 Industrial Automation Systems And Integration — Product Data Representation And Exchange - Part 11: Description Methods: The EXPRESS Language Reference Manual, International Organization for Standardization, Geneva, Switzerland, 1994.
[lviii] P. Andrews, An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof, Kluwer Academic Publishers, 2002.
[lix] V. Gane, J. Haymaker, M. Fischer, V. Bazjanac, Application of Design Scenarios Methodology to Evaluate the Effectiveness of Transparent Parametric Design Spaces, CIFE Technical Report No. 199 (http://cife.stanford.edu/online.publications/TR199.pdf), 2011.
[lx] A. Garcia, J. Kunz, M. Ekstrom, A. Kiviniemi, A., Building A Project Ontology With Extreme Collaboration And Virtual Design And Construction, Advanced Engineering Informatics, Vol. 18, pp. 71–83, 2004.
[lxi] P. Ellis, P. Torcellini, Energy Design Plug-in: An Energy Plus Plugin for SketchUp, ed. SimBuild Proceedings, Berkeley, California, NREL/CP-550-43569, 2008.