META-MODELING DESIGN EXPERTISE

A Thesis
Presented to
The Academic Faculty

by

Marcelo Bernal

In Partial Fulfillment
of the Requirements for the Degree
Ph.D. in the
School of Architecture, College of Architecture

Georgia Institute of Technology
August, 2016

META-MODELING DESIGN EXPERTISE

Approved by:

Professor Charles M. Eastman, Advisor
School of Architecture
Georgia Institute of Technology

Dr. John R. Haymaker
Research Director
Perkins + Will

Dr. Russell T. Gentry
School of Architecture
Georgia Institute of Technology

Dr. Ashok K. Goel
School of Interactive Computing
Georgia Institute of Technology

Dr. Dirk Shaefer
School of Mechanical Engineering
University of Bath

Date Approved: May 2nd, 2016

for Paula *

ACKNOWLEDGEMENTS

The accomplishment of this work could not have been possible without the support of my professors, classmates, friends, and love ones. I would like to express my deepest appreciation to my five committee members for their support, patience, sharp comments, suggestions, and challenging academic standards. I cannot begin to express my thanks to Professor Chuck Eastman, my thesis advisor, for sharing his unparalleled knowledge, experience, wisdom and example. He has been a source of inspiration and motivation. Under his guidance I have had wonderful opportunities during my yeas in the Design Computing Program at Georgia Tech. I would also want to extend my sincere gratitude to Dr. Russell Gentry who has encouraged me to pursue my goals and helped me to better understand the scope of my own work. I'm deeply indebted to Dr. John Haymaker who has been a constant support in many ways. He has invited me as teaching assistant of his class, helped me with paper writing and provided constructive criticism of my work. I'm extremely grateful to Dr. Dirk Schaefer who, first through his class and then as committee member, introduced me to the field of Engineering Design that lead me to adopt the Model-based approach of this study. I would also extend my deepest gratitude to Dr. Ashok Goel who through his reading of my work has helped me to identify the elements that bring coherence to this thesis. Each one has contributed his own way to the successful completion of this work, and I am profoundly indebted to all the professor of my committee.

The completion of this work would not have been possible without the contribution of Marc Simmons, Professor of the Practice of the School of Architecture, who has generously shared his expertise by providing the three case studies of this study, and my friend Axel Reichwein who helped me with his valuable advice to develop the computational infrastructure of this research. They have contributed to the success

of two important aspects of this study, validation and functionality of the proposed meta-modeling process.

I am grateful to my Professors from the George W. Woodruff School of Mechanical Engineering Dr. Chris Paredis, Dr. David Rosen and Dr. Suman Das; from the College of Computing Dr. Collin Potts, Dr. John Stasko, Dr. Ashwin Ram, Dr. Jarek Rossignac and Professor David Smith; and from the College of Architecture Dr. Godfried Augenbroe, Dr. Craig Zimring, Dr. Ellen Do, and Tristan Al-Haddad who also became a close friend. What I have learned from all of them though the coursework has been instrumental in obtaining my major in Architecture, minor in Mechanical Engineering, develop mu skills in Computer Science, and ,more importantly, in the completion of this research.

I am also wish to thank Professors George Johnston and Michael Gamble, formers Chair and Co-chair of the School of Architecture for believing in my teaching skills. They gave me the opportunity to closely work and learn from Professors Dr. Athanasios Economou, Dr. Harris Dimitropoulos, Lars Spuybroeck, Daniel Baerlecken, Dr. John Haymaker, Marc Simmons and Charles Rudolph while assisting them in both design studios and computing courses.

I cannot leave Georgia Tech without mentioning Mercedes Saghini, Academic Assistant, and Robin Tucker, Academic Advisor from the School of Architecture. I am deeply grateful for all their advice and assistance in in whatever I needed through my years in the Ph.D. program. Special thanks to Professor Jane Chisholm from the Language Institute whose patience and knowledge have had a great impact in my communication skills, even in my own native language.

I had great pleasure of closely working with my colleges and friends Paola Sanguinetti, Maher El-Khaldi and Matthew Erwin under the supervision of Professor Augenbroe during my early years at Georgia Tech with great success. I am also grateful to Donghoon Yang from the Design Computing program who provided invaluable

comments and suggestions. Special thanks to my friend Pedro Soza for his helpful advice regarding the methodological approach to analyze the case studies. And many thanks to Jin-kook Lee, Yong Cheol Lee, Mehdi Nourbakhsh, Matt Swarts, Andres Cavieres, Hugo Sheward, Shani Sharif, Kereshmeh Asfari, Sherif Abdelmosen, Samaneh Zolfagharian, Sabri Gokmen, Altug Kasali, Myrsini Mamoli, Alice Vialard, Martin Scopa and Jean Savinen. Our discussions and conversations on many topics along my years at Georgia Tech directly or indirectly contribute to find my own path in this journey, and also I had the wonderful opportunity to be exposed to a wide variety of cultures and points of view.

Very special thanks to my friends Alfredo Varas and Rodney Cone who were always there for me whenever I needed along the way. I am grateful of the chance to meet Cesar Pastén in Atlanta who became my fiend and part of our family. I am also deeply glad to have shared memorable times with my friend Francisco Valdés while pursuing the highest academic standards.

I am eternally indebted to Paula, my wife who has been also pursuing her Ph.D. at Georgia Tech, for all her love and support during our time in Atlanta. This adventure at Georgia Tech is one of the most incredible events in my life, and I thank God that I shared it with you. I do not have the words to express my gratitude to my parents Juan and Irene, my sister Carolina, and my family-in-law for their constant and unconditional support, for being so understanding for our long absences and for their warm welcome every year.

Finally, I would like extend my gratitude to the Digital Building Lab from the College of Architecture, the Fulbright commission and the Universidad Técnica Federico Santa Maria from Chile, and the MECESUP program from the Chilean Government for their support to succeed in this enormous endeavor.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

Assembly	Overall assembly of the product or project
BIM	Building Information Modeling
Block	Object representation in SysML
B-Rep	Boundary Representation
CAD	Computer Aided Design
CBD	Case-based Design
Chunk of Constrains	Already known general design restrictions
CBR	Case-based Reasoning
Conceptual Structure	Abstract architectural elements that locally organize the design such as corridors or hallways
CS	Configuration Specification
CSG	Constructive Solid Geometry
Design Schema	Overall pattern of organization of the design
DL	Design Language
DM	Domain Model
DP	Digital Project parametric modeling tool
DR	Design Rule
EMF	Eclipse Modeling Framework
FP	Failure Preventive approach
GD	Generative Design
GM	Geometric Model

GR	Geometric Representation
IM	Instance Model
Mapping	Linking UML models with CAD or BIM models
MD	Magic Draw modeling tool
MM	Meta Model
MBSE	Model Based System Engineering
OMG	Object Modeling Group
Part	Single physical part
PBD	Performance Based Design
PM	Parametric Modeling
Profile	Collection of Stereotypes
SBF	Structure Behavior and Function framework
Stereotype	UML extension mechanism
Sub-assembly	Component with specific sub-function made of parts
TO	Target Oriented approach
UML	Unified Modeling Language
TSV	Theoretical Structural Validity
ESV	Empirical Structural Validity
EPV	Empirical Performance Validity
SysML	Systems Modeling Language based on UML
TPV	Theoretical Performance Validity
Wireframe	Auxiliary geometry that implements the Schema

SUMMARY

The general problem that this research addresses is that despite the efforts of cognitive studies to describe and document the behavior of designers in action and the evolution of computer-aided design from concept to fabrication, efforts to provide computational support for high-level actions that designers execute during the creation of their work have made minimal progress. Studies on how designers frame a design situation, how they co-evolve problems and solutions, how they recall patterns of organization, and how they follow parallel lines of thought and generate design alternatives without extensive evaluation could benefit from computing capabilities the focus on simulating, reasoning, and predicting the behavior of objects and environments by manipulating abstract symbolic structures. However, the design-centered perspective, which focuses on the representation of the design artifact, has prevailed over the designer-centered perspective, which favors supporting the designer in action. Furthermore, what we see as representations of design products are only external auxiliary structures that do not represent the complexity of the expertise that resides in the designer's mind. Even though research efforts show progress in capturing and reusing knowledge in technical domains, expertise in design still remains an open research area.

This study seeks answers to the following questions: What is the nature of design expertise? How do we capture the knowledge that expert designers embed in their patterns of organization for creating a coherent arrangement of parts? And how do we use this knowledge to develop computational methods and techniques that capture and reuse such expertise to augment the capability of designers to explore alternatives? These questions pose a significant challenge: Such expertise is largely based on experience, assumptions, and heuristics, and lacking a process of elucidation

and interpretation, computational means of processing such unstructured information is currently out of reach. Although current parametric modeling systems can capture best practices in parametric relationships and facilitate the exploration of alternatives based on geometric variations in design configurations, they are limited because they are not able to support variations beyond the scope of the hierarchical structure of the parameters and geometric relationships. Thus, the range of design option prematurely decreases, revealing a need for migrating from parametric to topological modeling, which would more effectively support exploratory design tasks. The primary questions of this research lead to more specific questions regarding how these patterns structure a design configuration, how the process of reutilization of design knowledge works, and how a design space of alternatives is generated.

The hypothesis of this research is that the adoption of a meta-modeling process from the model-based systems engineering field (MBSE), understood as the creation of models of attributes and relationships among objects of a domain, can contribute to elucidating, structuring, capturing, representing, and creatively manipulating knowledge embedded in design pattern. The meta-modeling process relies on abstractions that allow the integration of myriad physical and abstract entities independent from the complexity of the geometric models; mapping mechanisms that facilitate the interfacing of a repository of parts, functions, and even other systems; and computer-interpretable and human-readable meta-models that enable the generation and the assessment of both configuration specifications and geometric representations. To explore how to connect abstract design concepts with a reusable repository of objects, this study integrates the flexibility of the meta-modeling process with parametric modeling techniques.

In addition to the introduction and conclusions, the content of this study contains five chapters. While Chapter 2 builds an interpretation of the nature of design expertise from the perspective of cognitive studies by identifying several distinctive

actions that experts execute during the design process, Chapter 3 reviews research efforts to embed various forms of design expertise into computational systems for generating, evaluating, and selecting design alternatives. These two chapters frame the contributions of the meta-modeling process to knowledge integration and reutilization. The first two chapters address the problem from a theoretical perspective while Chapter 4 does so from an empirical perspective using techniques of verbal analysis to distill and then depict actual design knowledge from the descriptions of three case studies by an expert designer in the field of custom façade system. Chapter 5 introduces the meta-modeling process, which employs the object-oriented non-system specific System Modeling Language (SysML) to interpret, capture, structure, model, and represent distilled knowledge from the three examples. Finally, Chapter 6 demonstrates the usefulness of the proposed approach, reusing design knowledge by producing design configurations and their corresponding geometric representations that range from reproducing the case studies and extending the scope of alternatives to exploring hybridizations based on normalization and the compatibility of e objects of the domain meta-model.

The results of this research include a framework for capturing and reusing design expertise, parametric modeling guidelines for reutilization, the multiplicity of geometric representations, and the augmentation of the design space of exploration. All of these tasks derive from the process of building a computable-interpretable and human-readable model of design knowledge. The framework is the result of generalizing verbal analyses of the three case studies that allow the identification of the mechanics behind the application of a pattern of organization over physical components. The guidelines for reutilization are the outcome of the iterative process of automatically generating well-formed parametric models out of existing parts. The capability of producing multiple geometric representations is the product of identifying ae generic operation for interpreting abstract configuration specifications. The

amplification of the design space is derived from the flexibility of the process to specify and represent alternatives. In summary, the adoption of the meta-modeling process fosters the integration of abstract constructs developed in the design cognition field that facilitate the manipulation of knowledge embedded in the underlying patterns of organization. Meta-modeling is a mental and computational discipline seeking for abstraction, generalization and reutilization.

CHAPTER 1

INTRODUCTION

1.1. The Problem of Capturing and Reusing Design Expertise

This research focuses on answering the question about how to capture design knowledge embedded in the underlying patterns of organization of architectural design so that the design can be reused in exploratory design tasks. In this regard, the first part of this thesis extends the discussion of the already published work (Bernal, Haymaker, & Eastman, 2015). Because of the diversity of disciplines involved in the building industry, research efforts related to capturing and reusing design knowledge have been devoted to the development of systems in technical domains such as structure, heating, ventilating, and air conditioning (HVAC), and mechanical/electrical/plumbing (MEP) systems (Amor & Faraj, 2001). A wide range of knowledge from these domains is currently embedded in parametric relationships, constraints, conditions, and functions (Shea, Aish, & Gourtovaia, 2003) for automatically generating details and evaluations of the design. However, the domains in architectural design entail a significant amount of tacit knowledge that enter into design decisions, and because of the ambiguity of the heuristics behind them, the translation to computational environments poses a challenging task. After all, such design knowledge is usually neither declared nor formalized in a set of rules or guidelines. On the contrary, it is strongly based on assumptions derived from experience, and only an expert can transform the declarative knowledge into procedural knowledge for reutilization (Akin, 1988). Such expertise still requires far more explicit definitions that must be implemented into computational environments (Eastman, 2004).

Even though designers differ in styles and preferences, the behavior of experts in architectural design can be characterized by common actions including interpreting

design situations (Gero, 1998), co-evolving problems and solutions (Dorst & Cross, 2001; M.L. Maher & Poon, 1996), recalling patterns of organization (Lawson, 2004), storing and reusing expert knowledge from design domains (Moreno et al., 2014; Popovic, 2004) and dividing tasks into distributed cognitive systems (Hollan, Hutchins, & Kirsh, 2000; Salomon, 1993). All of them are iteratively executed along the design process. All of these actions are highly sophisticated but nevertheless poorly declared. Capturing the expertise involved in any characteristic designer's action implies making explicit tacit considerations that represent the interaction of various forms of design knowledge while generating possible candidates as solutions.

When characterizing the role of computers supporting design tasks, we must ensure that what is in the mind of a designer and what is represented in the computer are not the same. While making decisions, designers must mentally consider multiple aspects such as design rules, norms, fabrication and installation inputs, costs, energy consumption, and lighting; however, they do not necessarily represent all of them in sketches or computational models. Eastman (2001b) states that the real structure supporting the design task is an internal representation in the designer's mind, and external representations are auxiliary structures (Figure 1.1). While the internal representations structure the knowledge, the external representations structure the design to facilitate the manipulation of such knowledge (Chandrasegaran et al., 2013).

Examples of external representations are mathematical or geometric models that could describe a completely different aspect of a design. However, as internal representations are more difficult to elucidate, they are still an active topic of research. In other words, we know what designers can do, but it is still not totally clear how they do it. In fact, representations of the design aspects of any model are integrated in even more complex cognitive systems. The notion of distributed cognition (Hutchins, 2000; Salomon, 1993) describes these systems, which include interactions between designers and team members, computer programs, the cultural context, and mental and external

representations. Therefore, the challenge for computational tools is supporting designers' behavior within a larger system of interactions instead of literally reproducing their internal mental mechanisms.

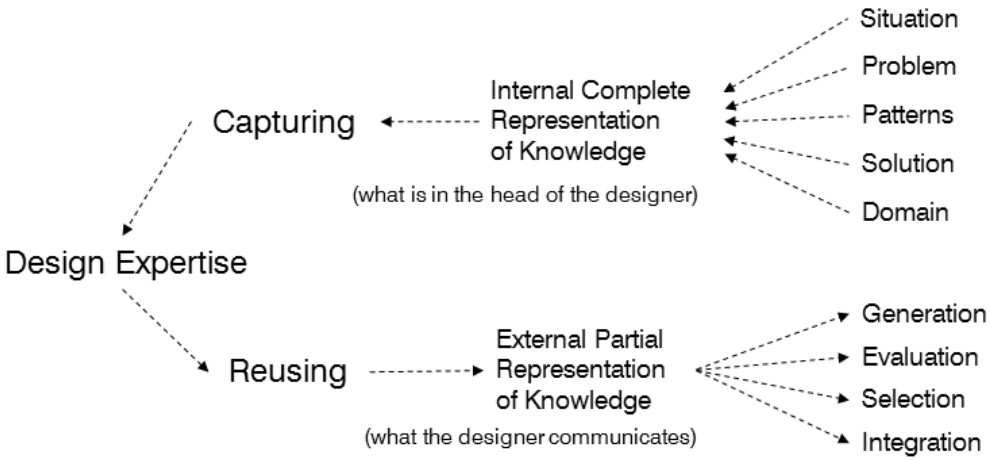


Figure 1.1. Scope of Capturing and Reusing Design Expertise

This research specifically focuses on the ability of expert designers to generate preliminary designs by combining features and parts of previous solutions and the development of methods and techniques to enable the generation of these multiple design configurations. This study raises the following primary research question*: How do we capture the knowledge that expert designers embed into patterns of organization that drive the production of design alternatives for reusability?* This question leads to the following secondary questions*:*

- How do the patterns of organization structure a design configuration?
- What is the fundamental process of the reutilization of design knowledge?
- How does one increase the design space to extend the exploration of alternatives?
- How does one build an extensible computational model of design knowledge?

To provide a more thorough understanding of the above questions, Chapter 2 includes an extensive review on the nature and the characteristics of designers in action

from a cognitive perspective. This review builds a framework for identifying the types and the characteristics of the Pattern of Organization, defining the taxonomy of the fundamental objects of a design domain, and visualizing the implications of the process of building a Design Domain.

1.2. Computational Support for Designers in Action

To build a clearer understanding of the contribution of computers to the actions that designers perform while designing, Chapter 3 of this study compares the well-documented characterization of designers in action mainly from journals that focus on design activities with design computing-oriented journals that pertain to computational approaches to design support. This study distills, sorts, and categorizes sixteen well-defined actions. First, those that address tacit knowledge involving implicit or non-declared assumptions were separated from those that address the reutilization of explicit knowledge. Second, the actions were organized into five categories*:* the understanding design situations, defining design problems, recalling patterns of organizations, building design solutions, and reusing domain knowledge. After the actions were categorized, each one was compared to fifteen computational approaches organized into four categories*:* generation, evaluation, selection of candidate solutions, and attempts at integrating the three. This comparison characterizes the role of computational approaches in their relationship with actions performed by designers to identify actions that are currently human-based with no other aids, including computer-aided or assisted by computer programs, computer-based aids that are fully automated, or computer-augmented aids that extend designer compatibilities.

The study reveals an open research area related to providing computational support for recalling the *Pattern of Organization*, including *Chunk of Constraints* applying restrictions to *Parts*, *Conceptual Structures* describing abstract architectural elements and *Design Schemas* describing the overall logic of designs. It also identifies the

emerging Model-Based System Engineering (MBSE) process as a promising alternative for integrating various sources of knowledge. The results support the following research hypothesis:

> *The adoption of the metamodeling process from MBSE can contribute to capturing and structuring design knowledge for creative manipulation for the purpose of the extending the capabilities of producing design alternatives.*

This hypothesis is based on the following working sub-hypotheses*:*

- The separation of the configuration specification from geometric representations allows the modeling of a variety of physical and abstract entities independent from any tool.
- The mapping mechanisms of metamodeling languages facilitate communication with a repository of parametric parts, functions, and other systems.
- Meta-models are both human readable and computer interpretable, facilitating the assessment of the models and the resulting design configurations.

The methodology integrates the flexibility of the metamodeling process with the capability of PM techniques to explore how to connect abstract conceptual design resources with a reusable repository of objects.

1.3. Methodology for Meta-modeling and Geometric Representation

The notion of "meta-model" (MM), which will be discussed in greater depth in Chapter 5, is adopted from model-based systems engineering (MBSE), which enables the modeling of large products and processes. MBSE, which enhances the capabilities of capturing mechanisms and facilitating reutilization (Gonnet, Henning, & Leone, 2007), can address the challenge of representing a Design Domain and complementing the capabilities of current BIM technology. The meta-model is a model of the data of the actual geometric model. Its purpose is to capture domain-specific semantics,

attributes, and relationships across parts in very abstract terms without any mediation of geometric models. The downside of the meta-model approach is its limited integration with CAD or BIM tools for the actual generation of a geometric model. This research explores the integration and the interaction of both technologies, meta-model and BIM, for capturing design knowledge in semantic terms for a specific domain (Eck & Schaefer, 2011), producing abstract specifications of possible configurations based on distilled design expertise, and generating geometric representations through CAD or BIM tools.

Since exploring alternatives implies myriad configurations, this study also introduces the notion of "topological modeling," which addresses the problem of specification for different design configurations derived from one design domain. Although the notion of topological modeling has been developed for engineering design domains (Wang, Wang, & Guo, 2003), the adoption in architecture requires far greater effort at declaring the tacit aspects behind the pattern of the organization of parts. Producing alternative design configurations implies creating the meta-model of the design domain as the main repository of design knowledge and developing mechanisms for creating candidate solutions according to the declared rules and heuristics of the domain. To achieve the necessary flexibility to produce topological variations, this research adopts the separation between configuration specification and geometrical representation. This autonomy provides flexibility of specifying configurations according to the shifting nature of design problems without dealing with the complexity and limitations of geometric models. Therefore, the proposed approach defines the configuration specification in very abstract terms and then proceeds with the geometric representation. Every resulting parametric model that corresponds to a configuration specification represents a design space of options of geometrical variation. Extending the range of specifications from the meta-model contributes to augmenting the design space of option to explore.

1.4. Validation Process

This study adopts the validation square process (Pedersen, Emblemsvåg, Bailey, Allen, & Mistree, 2000; Seepersad et al., 2006) that entails four stages of building confidence in a design methodology by an assessment of its usefulness based on the effectiveness and the efficiency of producing valid results at acceptable operation costs. This process evaluates the logical consistency, the appropriateness of case studies that effectively represent the scope of a problem, the usefulness of the results for the chosen case studies, and the generalization of its usefulness for other problems.

Chapters 2 and 3 address the logical consistency of the proposed metamodeling approach by building a theoretical framework for furthering our understanding of the nature of design expertise and describing the benefits of adopting the MBSE process. Chapter 4 presents the domain of the custom façade systems of three case studies. They derive from an expert design domain that involves a discrete number of components that will be captured and recalled to create distinct configurations. The elements of the domain were defined in collaboration with architect Marc Simmons, Ventulett III Distinguished Chair in Architectural Design at the Georgia Institute of Technology. Simmons is renowned for the implementation of high-end building skins at Front, Inc.

Figure 1.2 shows the entire process from start to end. It is divided into three main sections: distilling, capturing, and reusing design knowledge. The first section is based on techniques of verbal analysis (Chi, 1997) that distill knowledge from of the transcriptions of the description of the three case studies by an expert designer seeking units of knowledge regarding *Physical Components* and their *Pattern of Organization*. The second section describes the required implementation that enables capabilities of building a meta-model and mapping it with its corresponding parametric parts. The final section presents the design configuration specification of parts, attributes, and associations by creating instances of objects from the meta-model. It

also implements the interpreter, which reads the specification, identifies the mapped CAD parametric parts, and generates the parametric assembly of parts according to the design schema. Each will be discussed in detail in Chapters Four, Five and Six, respectively.

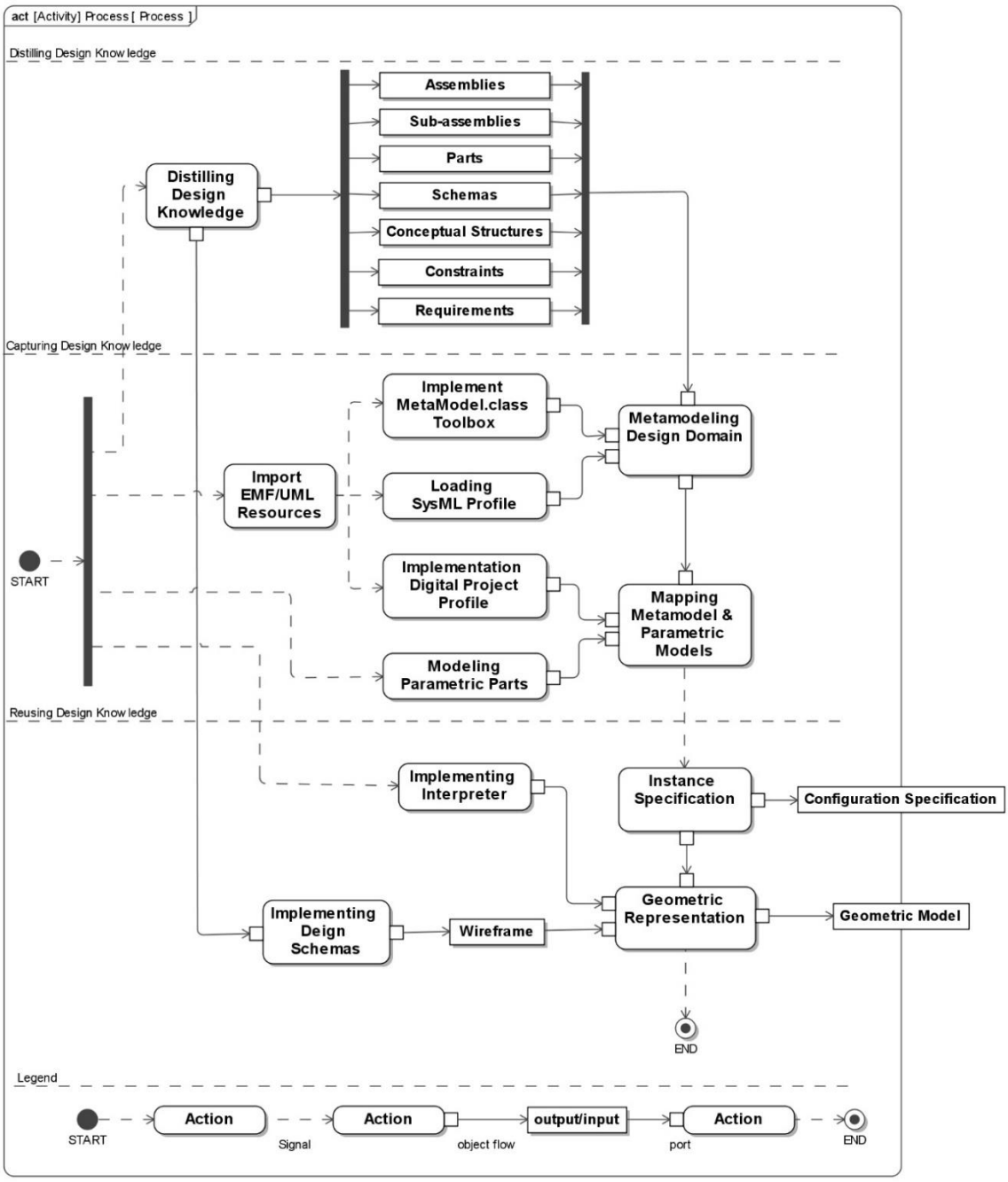


Figure 1.2. Diagram of the process of distilling, capturing and reusing design expertise

1.5. The Challenges of Implementing Design Knowledge Meta-models

The implementation of the proposed methodology presented in Chapter 5 structures methods into three layers to create the meta-model of the design domain, support specifications of the design configurations, and geometrically represent the resulting configurations. These layers, *specification, mapping,* and *representation,* are driven by two principles*:* abstraction and information hiding (Parnas, 1972). Since the *specification* layer lies where the meta-model is created, it is on the abstract level and mainly defined by design conceptualities. It corresponds to the formalization of expertise into *Physical Parts* and *Pattern of Organization*, which, combined, can produce novel configurations. The *representation* layer, the level on which the geometric models and the tool internals are handled, is the interface between the configuration specification and the means of representation that generate the actual geometric model (GM) (Figure 1.2). Finally, between these two layers lies the mapping layer, connecting and mapping the relationship between the high-level specification and the low-level geometric representation.

During the execution of the representation, an interpreter reads the abstract definitions and translates them into the scripting language of the CAD or BIM tool according to the mapping instructions. The principal elements of the generation process are the *Design Schemas* (Lawson, 2004), which are one of the most singular elements in architectural design. They constitute the underlying *Pattern of Organization,* which includes the auxiliary geometry and all of the necessary parameters and associations organized in a coherent manner. Indeed, the design is understood as an arrangement of Physical Components.

1.6. Migration from the Design-Centric to Designer-Centric Tools

While design knowledge is internally represented and connected in the mind of a designer, external representations are distributed among various file formats, but they

are not necessarily integrated. Although the ability to integrate and synthetize knowledge is a fundamental skill of designers, current computational tools are design-centric, with interfaces from the perspective of the physical components, rather than designer-centric, with a focus on supporting the actions that designers execute. Supporting creative actions behind patterns of organization seems to be the key to facilitating the reusability of design knowledge. In fact, as human intelligence can interpret and build relationships that the computer cannot, these actions mainly rely on mental rather than external representation. Although we know that designers recall patterns from previous designs, we still we have unresolved challenges with regard to providing computational support that positively impacts design quality.

The proposed process, based on meta-modeling, follows three main principles*:* the abstraction, mapping, and continuous growth of a reusable design knowledge repository. Modeling domain knowledge in abstract terms independent from any computational tool contributes to a clearer definition of conceptuality and avoids any references to the complexity of specific computational representations. The principle of mapping enables the linkage between abstract representation and actual geometric models of parts or scripts that embed functions. The last principle of continuous growth implies that capturing design knowledge is not a unique task, but instead, a continuous process of refining and adding new solutions that gradually extend the scope of the domain knowledge repository.

CHAPTER 2

UNDERSTANDING DESIGN EXPERTISE

Overview

The focus of this chapter is framing the general question of this research about capturing the design expertise embedded on the Patterns of Organization for reutilization in the generation of new designs. It reviews the main actions that expert designers perform and identifies the challenges of capturing their expertise from a computational perspective. The proposed interpretation of design expertise is based on an understanding of the shifting nature of design situations, the evolving relationship between a design problem and its solution, the particular notion of *design schema* that lends coherence to the arrangement of physical parts and preserves the integrity of the design intent, the mechanisms for generating design solutions, and the notion of design domain as a specific area of concentration. Finally, this chapter highlights the challenges regarding capturing design expertise such as the limits between explicit and tacit knowledge, the difference between declarative and procedural knowledge, formal declarations of design domains, the ambiguity of design problems, and the need to provide technological support to specify myriad design configurations.

2.1. Design Expertise

Design expertise combines knowledge and skills about an area of interest acquired after many hours of practice. Designers develop the ability to integrate declarative into procedural knowledge to address design problems. Depending on the background of the designers, the various levels of expertise include (Table 2.1) the novice, the beginner, the competent, the expert, the master, and the visionary (Crismond, 2001; Dreyfus, 1997; Lawson & Dorst, 2009). The degree of expertise varies among designers (Popovic, 2004). While novices can follow, with guidance, a given set of design rules, the visionary extends the boundaries of the field. They invent new ways of addressing problems and redefine the field itself by exploring its limits and learning from other domains. Although beginners can identify relevant aspects of design problem and competent designers can look for opportunities, neither is able to intuitively respond to design scenarios. By contrast, expert designers can produce such intuitive responses by recognizing the problem, selecting the relevant aspects to focus on, and linking the problem to possible solutions. They also constantly reflect on their decisions and reformulate the problem as well as the solution. Master designers, the remaining level, are characterized by their ability to apply more exploratory approaches to design beyond the application of already known formulas. They learn from success as well failures by always leaving room for experimentation. Based on the above classification, we determine that the expert designer level clearly defines an inflection point in this categorization. After all, from this level and higher, designers manifest autonomy and intuition. Expert designers appear to have the ability to identify "good things" and immediately recognize when their interpretation of the problem is "the right one" without extensive evaluation or further analyses (Dabbeeru & Mukerjee, 2008).

Table 2.1. Degrees of Expertise (Adapted from Lawson & Dorst, 2009)

Degree of Expertise	Characteristics
Visionary	Extends the field by learning from other domains
Master	Explores through experimentation
Expert	Produces intuitive responses by recalling previous knowledge
Competent	Seeks opportunities and interprets the problem
Beginner	Identifies relevant aspects of the problem
Novice	Follows given design rules

Designers can be competent in some areas and experts or masters in others (Dreyfus, 2003). The process of evolution from one level to the next is based on acquiring not only skills but also declarative knowledge and accumulation of experiences. Design expertise combines explicit and tacit knowledge (Woo, 2004) refined through years of professional practice. While the explicit corresponds to the formalization of knowledge into parametric relationships, rules, methods, procedures, or equations; the implicit corresponds to the non- declared and non-verbalized heuristics driving design decisions. Another important issue integral to our understanding of design expertise is that it does not necessarily reside in one single designer. Design expertise can be also distributed among a team of collaborators in the various levels of development in many areas. Therefore, the task of capturing expertise could involve multiple sources along the design stages.

Becoming an expert takes around 10,000 hours, or five years, of dedication to the field. This level of performance is the baseline of the focus of interest of this research. Even though experts have distinctive approaches to design, they share common characteristics (Table 2.2). They reference new problems to recognizable problems linked to previous solutions (Dorst, 2007), rapidly generate a small number of

possible alternatives (Lawson, 2004), co-evolve a problem and a solution (M.L. Maher & Poon, 1996), and integrate knowledge across fields (Kruger & Cross, 2006).

Table 2.2. Designers' main actions

Related to	Designer's Actions	Source	Motivation
Design Situation	Reformulating SBF	Goel & Stroulia, 1996	Shifting Design Direction
	Forming Analogies	Lindsey, 2006; Eastman, 2001	
	Looking for Emergence	Gibson, 1977; Schoen, 1983	
Design Problem	Framing	Lawson & Dorst, 2009	Matching Problem-Solution
	Building Ill-defined Problems	Dorst, 2003	
	Co-evolution	Maher, 1996; Dorst et al., 2001	
Pattern of Organization	Recalling Chunk of Constraints	Gobet, 2001	Design Coherence
	Recalling Conceptual Structures	Lawson & Dorst, 2009	
	Recalling Design Schemas	Lawson, 2004	
Design Solution	Following Parallel Lines of Thought	Cross, 2004	Feedback from Design Alternatives
	Evaluating Preliminary Solutions	Dabbeeru & Mukerjee, 2008;	
	Integrating Knowledge	Cross, 2004	
Design Domain	Recognizing Problems	Lawson, 2001	Knowledge Repository
	Reusing Physical Parts	Lawson & Dorst, 2009	
	Applying Design Rules	Schoen, 1988	
	Applying Evaluation Methods	Becker, 2008	

To better understand the complexity involved in the problem of capturing design expertise for reutilization, Figure 2.1 presents a model of interaction of the fundamental actions that designers execute when addressing *design situations* as the scenario in which the design occurs, formulating and solving *design problems*, recalling underlying *patterns of organization* that represent conceptual and abstract relationships embedded within a design solution that bring coherence to the design as a whole, producing such *solutions* and evolving their *design domain* as the main repository of knowledge in an area of interest

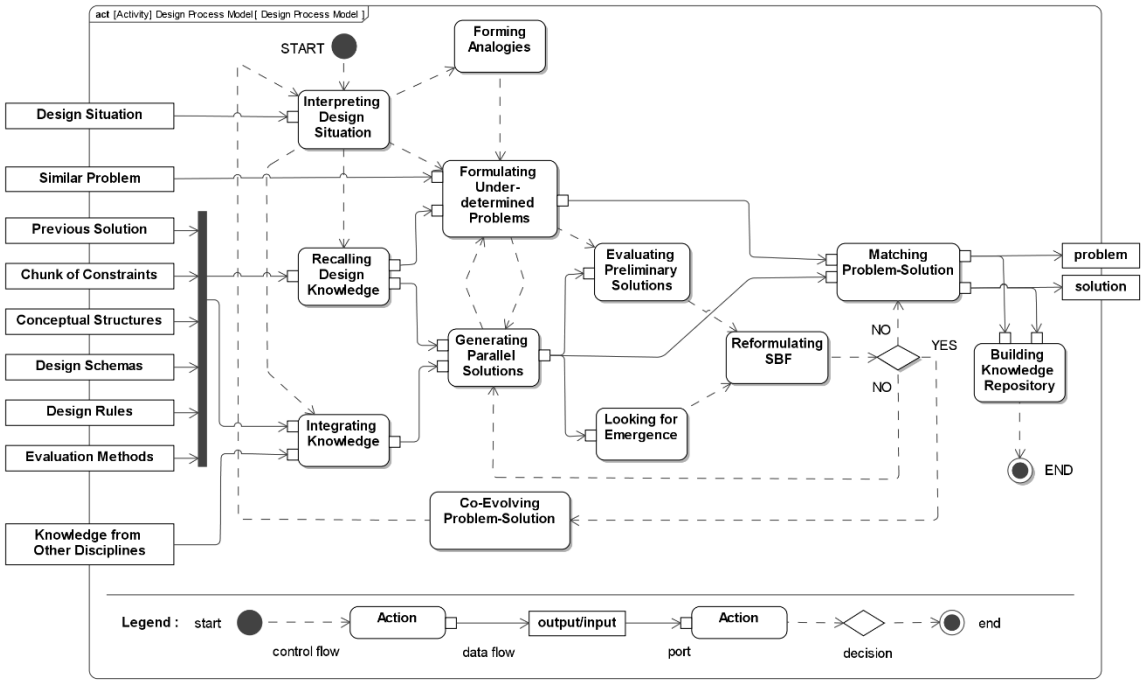


Figure 2.1. Model of interaction of designers' main action (Adapted from Bernal et al. (2015))

Designing is one of the most challenging cognitive tasks. Expert designers combine explicit and tacit knowledge and integrate declarative into procedural knowledge while executing design actions. These two perspectives of the knowledge they manipulate, tacit-explicit and declarative-procedural, suggests four possible combinations with blurred boundaries among them (Table 2.3): declarative-explicit, declarative –tacit, procedural-explicit and procedural-tacit. The survey of designers' main actions listed in Table 2.2 from this chapter is sorted from those actions that concentrate a higher percentage of tacit considerations in the top to those mostly with explicit knowledge in the bottom. However, since they are actions, they belong to the procedural domain. The problem is that in terms of capturing computers demand explicit definitions, but the most distinctive actions that expert designers execute are based on tacit considerations and those that are based on explicit knowledge represent lower level tasks. For example, while knowing the building code is declarative and explicit knowledge with no major impact in creative tasks, framing the design situation is procedural knowledge based on tacit assumptions. The first one can be implemented in

computational environments as procedural and explicit knowledge, and just understanding the second one is still an open research area far from implementation in computational environments yet.

Table 2.3. Design knowledge possible categories

knowledge	declarative	procedural
explicit	declarative- explicit	procedural-explicit
tacit	declarative -tacit	procedural-tacit

2.2. Interpreting Design Situations

To build an understanding of the complexity of design expertise, we will begin by reviewing the notion of "design situation," which describes the context in which the accumulation and the retrieval of knowledge occurs. Depending on the context, expert designers can create a variety of interpretations from the same set of requirements and scenarios. While working in conceptual design, experts interpret the design situation based on their subjective understanding of the problem and previous experience because they construct memories related to a current situation. Variations in the definition of the situation trigger unique trends of design evolution and opportunities. New interpretations of the situation based on the original understanding and representation can produce a new representation that provokes further novel interpretations (M.L. Maher, J. Poon, & S. Boulanger, 1996), and every new interpretation and related representation adds value to the design. Schön (1983) referred to this dynamic process *reflection-in-action* because new features, properties, and any other relevant aspect that was not intentionally put there could emerge, affecting the evolution of the design. In the same direction, he later pointed out that the designer sees, draws, sees again, and reflects (Schön, 1988).

Although Schön's statement is based on drawing as the medium of the representation, it is still valid for a computational environment since designers also can

see, recognize, detect, or discover relevant elements of a situation derived from their own operations that contribute to constructing re-interpretations (Gero & Kannengiesser, 2004). Such re-interpretations rely on reformulating the physical structure, behavior, or function (SBF) of the design (Goel & Stroulia, 1996); building analogies that recall information from previous experience or even other fields (Goel, 1997; JS Linsey, Laux, Clauss, Wood, & Markman, 2007; J Linsey, Murphy, Markman, Wood, & Kurtoglu, 2006); and seeking emergent features that may be of further use in the design process according to the theory of affordances (Gibson, 1977). These highly sophisticated actions influence the interpretation of the situation, and it remains unclear how to best support them with computers.

Reformulation

Reformulation means changing what the design is about. From Gero's perspective, it is based on the Goel's Structure, Behavior and Function framework, which intends structure to be the physical composition of the artifact, behavior as the way it responds to stimuli, and function as its intended purpose. Each of these three components of the paradigm can undergo a variety of changes that either require a structure as the starting point (Brown, 2003; Chandrasekaran & Milne, 1985) or adaptation of the Behavior and Function components (Goel & Bhatta, 2004). According to this approach, representations play an important role since they can highlight aspects of the structure that trigger reformulations. The first type of reformulation affects the structure and defines a new design space, intended as the set of possible variations of the current state. These kinds of changes are very interesting since they involve parametric and topological transformations, parametric in the sense that values driving changes in the geometry (e.g., the length, the width, the diameter) and topology in the sense that the configuration can change (e.g., the number of legs of a table, or stick versus unitized systems of building facades.) We will review this type of change in further sections since changes in the configuration of designs are very challenging. The

second type of reformulation derives from changes at the behavioral level, which means that the structure must vary to satisfy the requirements of a new set of possible behaviors. The last type is triggered by changes at the functional level that may provoke changes at the behavioral level and consequently at the structural level. Reformulation implies potential changes and a shifting direction that affect the very nature of the design.

Forming Analogies

Analogies are another mechanism for reinterpretation and creativity (Goel, 1997). They work by recalling information about previous experiences or other fields in an effort to address a current design situation. They are a likeness type of relations expressed in terms of A is to B as C is to D. Forming an analogy relies on the identification of commonalities in two inherently different design situations. An analogy is primarily an association that adds information to from an existing knowledge (Eastman, 2001b). Such knowledge from one situation can be applied and adapted to a different design situation based on commonalities (Goel & Bhatta, 2004) . The association between one situation and another relies on two elements referred to as the "source" and the "target." While the source is the information from which a relationship is built, the target is the situation in which the relationships is applied through induction. The relevant analogies in design are called "deep analogies." Unlike shallow analogies, these are high-level abstractions such as the function or the nature of a structure. An example of the utilization of deep analogies in design is the decomposition of engineering problems into requirements that define the desired functions that satisfy them, and the utilization of these functions and even sub-functions to find working principles based on analogies. A working principle is an abstraction of a mechanism that identifies the very essence of the logic of a phenomenon. Such mechanisms can be based on principles of thermodynamics, physics, chemistry, or nature in a more general sense. Every working principle (e.g., leverage, buoyance, rotation) is associated with

desired functions. The collection of working principles creates a working structure (Pahl, Beitz, & Wallace, 1996) that synthesizes and organizes the principles into a coherent pattern. The resulting working structure is a collection of diverse principles coming from various sources. The analogical reasoning behind the searching mechanism of potential working principles is a highly creative moment in an engineering design (Moreno et al., 2014).

Looking for Emergence

Emergence is strongly related to what Schön (1983) describes as reflection-in-action since it refers to features, properties, or any other relevant aspect of a design that was not intentionally added to the original design. Every time experts face a new design problem, they recall previous knowledge that links a large amount of information beyond the scope of the motivation of the original recall. Expert designers seem to identify and remember such information and affordances that may be useful later in the design process. Since the assigned or expected function of an object can also afford other unexpected functions, the reinterpretation of the design situation can provide all kinds of unexpected opportunities. Another reading of Schoen's approach is based on visual reasoning in design (Oxman, 2002), in which the representation plays an important role because it can trigger the recognition of emergent patterns, opportunities, and anomalies. Different from computers, which cannot deal with ambiguous representation, designers have the ability to recognize embedded patterns within representations even though they were not intended in the original design, indicating the presence of more than the obvious perception of the representation. These perceptions can be related to emergent shapes or sub-shapes (Stiny, 2001) or relationships and correlations. Any of these emergent interpretations can introduce new trends of development. Addressing ambiguity in computational representation and pattern recognition is still an open research area (Grasl & Economou, 2011, 2013).

The three presented actions (reformulating SBF, building analogies, and seeking emergence) play an integral role in the interpretation of a situation. In fact, a situation is not a static scenario. On the contrary, it is actively built by the designer in a very "fuzzy" way since all the above mechanisms are combined while operating over the representations, sometimes altering the direction of the design. Expert designers seem to feel comfortable applying all of these resources in a design situation. Computational representations provide the context that allow reinterpretations to occur.

2.3. Formulating Design Problems

This section introduces the difference between design problems and requirements. While requirements (Pahl et al., 1996) are a list of features and desired performance goals (Becker, 2008), design problems represent a formulation that organizes the relationships and tradeoffs among the requirements. In other words, requirements are the initial inputs that define design problems.

The notion of a design problem is strongly related to the notion of a design solution. Designers not only design a solution but also a problem. Creative expert designers usually define complex problems from initially ambiguous or incomplete requirements (Cross, 2004). This ability generates highly differentiated problem formulations among designers, since their preferences or skills determine what aspects they select to work with. While formulating a problem, they speculate with incomplete evidence and make conjectures about possible solutions without devoting a great deal of effort to understanding the original requirements of the problem. During this process, expert designers execute three important actions: framing the focus of interest within the design situation, building ill-defined problems, and co-evolving the problem with the solution (Dorst & Cross, 2001; Mary Lou Maher, Josiah Poon, & Sylvie Boulanger, 1996).

Framing Design Problems

Designers seem to select specific aspects of a problem to focus on, referred to as “framing.” Framing implies setting the boundaries of a design situation, selecting the focus of attention, and imposing coherence in the decisions. Framing not only occurs at the beginning of the design process but also periodically occurs along the entire design process. This shifting framing represents how the designer views the problem at a specific time. In the early stages of design, they focus on the formulation of a problem by attempting to identify or select key aspects from which they will approach the problem to produce tentative preliminary designs. By contrast, in late stages, they tend to devote their attention to technical aspects of the design. Early in the process, designers have a higher degree of freedom, explore various design alternatives, and perform rough analyses for decision-making by determining consistency between design intent and requirements. Unlike in the early stages, in later stages, designers solve problems regarding the technical aspects of the project within the boundaries defined by earlier decisions.

Although the design stages require diverse knowledge, expert designers make early decisions including implicit considerations of technical aspects from late stages. They can synthetize and balance design intent and inputs from technical considerations early on in the process because they try to be consistent with their earlier framing of the problem.

Building Ill-defined Problems

Designing is not a deductive activity, since there is not a unique way to link the needs, requirements, intentions with possible solutions. On the contrary, there are multiple ways to address the same problem. This openness is due to the description of the problem is never complete and the problem (needs, requirements and intentions) and the solution (the physical structure) belong to two different conceptual worlds. From this perspective design problems are neither fully defined nor fully open. Some

aspects of the problem are unavoidable and highly determined, other aspects are under-determined and require interpretation, and other aspects are un-determined and leave room for the preferences of the designer who defines the criteria to address them (Dorst, 2007). Experts deliberately treat the design problems as ill-defined and expend considerable amount of time defining them. They are constantly shifting the goals and constraints of the problem.

Co-evolving Problem-Solution

Problem and solution co-evolve. Designers use solution conjectures as a means to better understand the problem. They use tentative solutions as mechanisms to better understand the nature of the problem instead of analyzing the problem. Tentative solutions often expose hidden aspects and trigger the redefinition of the problem, which implies that the solution must be adapted to the new conditions. This dialog between problem and solution iterates variable number of times before to reach a matching problem-solution pair. Expert designers design the problem as well the solution. They explore the problem and solutions in parallel (Rowe, 2004) unlike experts in other fields. Along this process they use a generative approach rather than a deductive one, because it facilitates exploring and finding the problem and the solution matching pair. Finding the matching pair problem-solution (Cross, 2004) implies that the definitions of significant aspects of the solution are already contained in the problem formulation and the recognition of the problem is associated with possible solutions. Through this co-evolving process, expert designers produce reliable solutions early on, ether with no need of radical modifications or fluently modified if necessary. Such modification can be simple adjustments or shifting directions.

2.4. Recalling Patterns of Organization

Experts accumulate large number of references from their own work. Through years of professional experience they establish a repository of relationships problem-

solutions. Remembering these relationships is also is a way to speed up the process of solving recognizable problems, because referencing previous solutions implies that significant elements of the solution for the new problem are already known. From that perspective, new solutions are vernacular in expert designers (Lawson, 2004).

We can recognize at least three main encrypted logic of organization that expert designers recall to structure and give identity to their designs: *Chunk of Constraints* as well-known relationships and trade-offs among aspects of the problem (Gobet et al., 2001), *Conceptual Structures* as abstract architectural elements that locally organize the designs such as corridors or hallways , and *Design Schemas* or overall patterns of organization imposed to the problem (Lawson & Dorst, 2009)

Recalling Chunk of Constraints

Because of their experience, experts know many constraints of the problem in advance (Gobet et al., 2001) that help them to frame and focus the design space (Flager, Welle, Bansal, Soremekun, & Haymaker, 2009) and elaborate valid solutions, valid in the sense that they are consistent with the constraints of the problem. Many of these constraints are often implicit relations that describe important aspects of the problem form the simple ratio between the span and high of a beam to more complex relationships such as the dependency among geometry, fabrication limitations, cost and timing. Therefore, splitting the problem in chunks almost immediately builds and interpretation and provides clues about further steps. Experts do not see a general problem. They visualize deeper levels of elaboration of the problem. In fact, the problem is a coherent set of relationships across many constraints.

Recalling Conceptual Structures

A conceptual structure represents all the set of relationships that designers establish across systems and parts of a design. Often the physical components belong to more than one conceptual structure. For example corridor and rooms could share

the same wall which could be also part of the structural system. The complexity arises when the designer constantly and unconsciously switches from the perspective of one system to another during the design process. The purpose of this oscillation is the definition of the final physical components from multiple perspectives in a very synthetic manner. Although this is an important designers' skill, current design tools mainly offer interfaces from the perspective of the physical components.

Recalling *Design Schema*s

Through protocol studies and interviews, Lawson realized that designers use idiosyncratic terms such as *belvedere* and other metaphors such as *boulevard, rotunda, atrium,* or *node* among others to reference abstract patterns of organization and not necessarily a specific shapes (Lawson, 2004). These *Design Schema*s are high level conceptuality and at the same time materialized in low level auxiliary geometry. From project to project expert designers develop their own *Design Schema*s which represent their own way to solve recognizable problems. A *Design Schema* or schemata is an abstract structure even more abstract than a typology. It is a way to organize the space; it is a pattern of organization rather than a typology. It includes the geometry and all the related relationships as well. A *schemata* probably represents one of the most relevant aspects of the design knowledge since it is a mechanism to organize in a coherent manner diverse information about the design. This *Design Schema* captures implicit knowledge representing the preferences and guiding principles of a designer and brings coherence to the design as a whole.

2.5. Generating Design Solutions

Expert designers make decisions based on their experience balancing design intent and inputs from technical considerations in a very efficient and synthetic manner. This ability is the result of many hours of deliberated practice, and it is not necessary related to talent (Cross, 2004). On the contrary, design experts rely on the accumulation of

experience though the exposition of a large number of design situations. From their experience they distill guidelines, rules, priorities and preferences that are constantly refined from project to project. Along their careers, every expert designer develops a distinctive set of principles based on personal experience. These principles guiding the decision making process represent the expert's own way to design or expert's style. From the perspective of Bloom's taxonomy of learning domains (Bloom, 1956), expert designers are allocated in the evaluation category, the higher level, because they have not only knowledge about their field and know how to apply it in different situations, but they also make judgments and assessments of the effectiveness of their decisions. Their critical thinking allows them to compare alternatives, assess the technical viability of potential design solutions and include previous experiences external criteria in a very short time while producing potential solution.

Following Parallel Lines of Thoughts

Experts have the ability to conceptualize the design situations, identify the underlying principles behind the problem, redefine the problem and reuse their experience to rapidly generate possible matching solutions. During this process they seem to be able to handle *parallel lines of thought* and tend to generate a range of design options or a design space rather than one single solution. This particular ability allows expert keep their options open while design proceeds (Cross, 2004). Experts generate more than one possible solution and perform preliminary evaluations of their tentative designs before further design developments.

Evaluating Preliminary Solutions

Designer can integrate multiple information describing objects or spaces and mentally simulate the activities supported by them to evaluate un-built designs. This process of assembling relies on the use of mental imagery to integrate the information into an internal representation of shapes or spaces. The process of conformation of the image of the design is very fast and facilitates assessments such as floor plan layouts,

usability of objects, or space conflicts (Eastman, 2001). A wide range of these assessment methods can be formalized and described for reutilization within a design domain.

Integrating Knowledge

To produce solutions, experts can integrate knowledge across different domains while working in a problem by developing aesthetics and other technical aspects in parallel through iterations. Although expert designers frame the problem and define a focus of interest, they must address the problem from different perspectives to satisfy requirements from different sources and stakeholders. To balance and compensate different requirements the usually develop the different parts in parallel through several loops. They also have a memory of relationships problem-solutions that constitutes a network of relationships that they constantly invoke.

2.6. Developing the Design Domain

Becoming an expert implies acquiring skills and knowledge in a specific design field as area of concentration or Design Domain (DD). Although a domain defines common areas of specialization such as curtain walls, precast concrete, tensile structures and others, it can be also understood from the perspective of the specialization of the designer. From that point of view, given the same area of specialization and depending on the boundaries of the designer's expertise, a different DD can be built. The DD groups and organizes all the knowledge about entities and relationships that describe the universe of the area of concentration. These entities can be concepts, objects and related attributes, parameters and constraints, relationships and interaction among those objects, analysis and evaluation methods or typical design problems.

The characterization of a design domain has challenges regarding the declaration of typical design problems, the formalization of design rules derived from

heuristics or the declaration of evaluation methods to assess different aspects of the designs (Shea et al., 2005). The following subsections attempt to explain the nature of the main elements of the DD and the inherent challenges to formalize them.

Recognizing Problems

Experts learn from the solutions as well from the problems. Through their professional experience they distill the commonalities across design problems and build their repository of recognizable ones. Based on this catalog of problems experts can build up an image of the new problem and identify the problem type with partial or incomplete information. They reference the new problem on typical and recurrent recognizable problems instead of defining the problem from analysis.

Reusing Physical Parts

Designers describe what they know in terms of concepts rather than physical components. This conceptualization is what defines the semantic of the elements of their domain of knowledge (Venugopal, Eastman, Sacks, & Teizer, 2012).Concepts such as *center*, *perimeter*, *hallway* or *entrance* are abstract structures which are not literally one single physical component like a door or a window frame nether an assembly. They make the distinction between design concepts and physical components since physical components are organized under design concepts and they can also share physical components. The process of gaining experience and structuring their own knowledge seems to be driven by the development of such high level conceptuality that organizes the extensive list of parts, attributes and techniques to put them together.

Applying Design Rules

Rules represent the formalization of procedural knowledge derived or distilled from heuristics, norms, and guidelines. Rules are strongly linked to types since types define the context and range of validity and versatility of the interpretation and

application of design rules, and the rules define the range of adaptability and combination of different types. Rules can represent knowledge about selection of pattern of organization, ranges of adaptability and dimensioning of physical parts, selection of the components of assemblies among others.

Applying Evaluation Methods

Expert designers seem to run mental simulations (Dogan & Nersessian, 2010) of their designs to evaluate the overall functionality and feasibility. Although these mental simulations are based on tacit knowledge, many of the implicit calculations and estimations can be rationalized and formalized into explicit procedures. These formalizations and extensions of them represent proven best practices that are constantly reused. They are strongly connected with the ability of designer to divide the problem in chunks of constraints since the chunking mechanisms are based on repetitive trade-offs among requirements. Most of these methods can be found in handbooks and captures in computational tools.

2.7. Discussion: The Taxonomy of a Design Domain

The assumption of this research is that capturing design expertise to better support designers in action is a process of gradually making explicit tacit considerations involved in the procedural knowledge that the designers manipulate. Furthermore, that process demands at least three conditions: definition of the primitives units or basic knowledge modules, declaring what structures and provides integrity to the designs, and developing the repository that collects the knowledge through time.

Defining the Types of a Design Domain

In this regard, Schön (1988) points out: *What should be taken as the primitives or fundamental units of design knowledge*?. The identification of these primitives is the first step for further specialization and reutilization. Defining the taxonomy of a DD in architecture implies making explicit tacit assumptions and declaring types and attributes

of the entities of the domain that can further be extended though specialization or combination.

Design knowledge within a domain has multiple forms such as custom parts and assemblies, parametric relationships at the parts as well as assembly level, design rules and constraints, decision rules, evaluation functions, and abstract entities such as conceptual structures or *Design Schema*s. The specific challenge in terms of capturing is rooted in the diversity in nature, sources and related formats to store such range of knowledge. Parametric models of physical parts are totally different than evaluation functions. The first one is a 3D model, and the second is a concatenation of equations. Furthermore, while parametric models describe parts and assemblies, designers think in terms of conceptual structures. This diversity is linked with the limits of the elicitation of the tacit knowledge and the degree of detail (or abstraction) of the declaration of the types of a domain.

Declaring the Underlying Patterns of Organization

What does give coherence to a solution made of parts and rules from a design domain? A collection of physical parts and design rules seems to be not enough to bring coherence to designs. Finding the mechanisms of integration of the variety of components of a design domain is one of the main motivations of this research

Expert designers recall and reuse underlying patterns of organization, rather than a collection of physical parts. These patterns are typical design problems that structure the requirements, conceptual structures that organize the parts, and the *Design Schema*s that bring coherence to the design as a whole. All the combinable set of components of the domain models are driven by these immaterial entities. Unlike other design fields such as the discussed airplane design, expert designer in architectural domains do not rely on the repetition and combination of limited universe of components. On the contrary, rely on these patterns of organization to bring

coherence to the diversity of elements involved in a design since they synthetically represent the integration of the matching pair problem-solution.

The underlying patterns of organization embedded within *Design Schema*s contribute to this linkage since they are driven by the problem and at the same time they represent the fundamental features of a solution. Reusing previous resources implies that not only a design rules or physical parts have value, but also the way that they have been combined.

Modeling a Design Domain

How to model a repository that growth through time? Assuming the declaration of the taxonomy of elements of a domain and the identification of the mechanisms to provide integrity and coherence, the next challenge is building a dynamic process capturing design knowledge in an extensible repository. This process involves multiple sources since the expertise does not resides only in one single designer, On the contrary, it is distributed among collaborators. Such declaration should be based on design terminology independent from means of representation to facilitate the generation of solutions through any kind of tool by combining and adapting elements from such domain. Through the different projects ether the types can be vertically specialized in sub categories and the repertory can also be enriched and horizontally extended through time, similar that experts do. Modeling a domain also requires the development of generation methods of possible solution to support exploratory design tasks which will be discussed in the next Chapter.

CHAPTER 3

REUSING DESIGN EXPERTISE

Overview

This chapter focuses on the second part of the research problem, the reusability of design expertise to computationally support the generation of design alternatives. It reviews several approaches to address the challenge of embedding design expertise in computational systems. This review organizes them in four categories based on the purpose of the system: generation, evaluation, selection and integration of all of them. Their achievements and limitations are discussed and compared with the main designers' action distilled in chapter two. This comparison contributes to identify actions that mainly are *human-based* without major computational assistance; *computer-aided* with valuable feedback or trivial tasks automation; *computer-based* with fully automation of the process; or computer-augmented in which the computational tools extend the designer's compatibilities. Results shows open research areas still lacking of computational support and promising progress in systems integration that facilities communications across different platforms and the integration of diverse sources of design knowledge in shared repository. Finally, this chapter highlights different challenges regarding reusing design expertise in terms making explicit usually tacit consideration and recalling knowledge to generate new design configurations.

3.1. Embedding Design Expertise in Computational Tools

Designers rely on their experience, instincts and common sense to read design situations and make decisions. (Eastman, 2001b)also points out that it is important to make a difference between what knowledge is in the computer and what is in the designer's mind to facilitate their interaction. Computers facilitate how designers manipulate graphical information, perform evaluations and compute decision making models, but still they provide limited support to the core of the creative process maybe due to the integrated nature of design or the shifting definitions of problems. However, current technology is gradually expanding the scope of the embedded design knowledge into design tools to address the multiple trade-offs among all the aspects involved in the design process.

Despite limitations and sometime skepticism, important efforts have been undertaken to capture design expertise and embed it into computational systems for reusability (Table 3.1). The type of embedded expertise varies from one system to another. We can distinguish four main tendencies: generation oriented, evaluation oriented, selection oriented and attempts to integrate all of the above. The first group is focused on assisting the designer to find or generate solutions, the second one on evaluating the solution candidates and providing feedback to validate the design or make changes if necessary, the third one focus on the implementation of algorithm to select solution candidates and the last group tries to capture the interaction among generation, evaluation and selection.

The next sections describe prevalent approaches in the four groups and discuss their purposes, logic, benefits and limitations with respect to the design actions they provide support for.

Table 3.1 Current approaches for design knowledge re-usability

Related to	*Approaches*	*Captures*
Generation	*Parametric Modeling*	*Physical Parts and Assemblies*
	Expert Systems	*Decision Rules*
	Generative Design	*Requirements*
	Design Languages	*Physical Parts and Design*
	Agent-based	*Interactions*
	Case-based Reasoning	*Typical Problems*
Evaluation	*Performance Evaluation*	*Evaluation Methods*
	Rule Checking	*Constraints*
	Constraint-based	*Design Scenario*
Selection	*Utility*	*Value*
	Multi-attribute	*Trade-offs*
	Optimization	*fitness*
Integration	*Interoperability*	*Exchanges*
	MBSE	*Integration*
	Custom Integration	*Interactions*

2.1. Solution Generation

The following set of approaches corresponds to systems whose aim is to support designer to generate (or find) solutions. This review is based on the type of knowledge they are dealing with and the process to generate the solution candidate. The studied approaches are: Parametric Modeling (PM) that is one of the fundamental technologies behind Building Information Modeling (BIM), Expert Systems (ES) applied in automatic detailing, Generative Design (GD) that implements algorithms based iterations to generate solution candidates, Design Languages (DL) focused on the combination and adaptation of well-known physical parts in new assemblies, Agent-based Design (ABD) that captures interaction among discrete entities and Case-based Reasoning (CBR) that reuse previous solution for new problems.

Although these approaches are classified by their main characteristic, some specific tools and methods combine them into the same workflow in different ways. For

example, while Expert Systems emphasize the utilization of design rules and Design Languages the generation of different configurations, both are based on Parametric Models. Besides this overlapping, the focus of interest is having a clearer understanding of the level of support they provide the action that designers perform, their limitations and achievements.

Parametric Modeling

Capturing design knowledge into parametric libraries is the simplest way to embed design knowledge into systems and facilitate *reusing physical parts.* Standard objects (e.g. furniture, mechanical parts, or building components) shared across different projects are described as adaptable objects. This standard design knowledge is captured in parametric assemblies, sub-assemblies or single parts, through parameters, constraints, and properties. All these variables can be edited by the user during the insertion of the object into the model since the relationships are organized in hierarchical binary tree structures that are updated as changes occur (Kalay, 1989; A. Requicha, 1980).The creation of those instances of the object can be manual or automatic depending on the implementation of additional rules. These objects are for general purposes and facilitate repetitive tasks. BIM tools have a wide range of these libraries embedding standard knowledge from different domains. An emergent set of tools and services are providing building products and assemblies to multiple platforms offering compatibility with different authoring BIM tools. These Building Element Models, or BEMs, are going beyond simple standardization. They are developing parametric libraries according to the specification of manufactures to facilitate the access to commercial products.

Different than the standard parametric libraries, custom parametric families embed the expertise derived from design practice. They embed successful original designs representing the best practices of a firm or designer (Bernal & Eastman, 2011). The specificity of the embedded knowledge is captured within parameters, constraints,

conditional, attributes, and, frequently, in more complex parametric functions to derive parameters based on external inputs during instantiation (Eastman, Sacks, & Lee, 2004). Design expertise can be embedded at the low part level as well as the assembly level. At the single object level, input parameters as well as derived relationships drive the behavior (G. Lee, Sacks, & Eastman, 2006) or the degree of variation of the object as an individual unit. At the assembly level, the shared driving parameters drive the relationships and constraints (Nassar, Thabet, & Beliveau, 2003) among the parts. Since an assembly is an object, and an object can be an assembly, multiple nested relationships such as an assembly containing a sub-assembly of single objects co-exist in large models. They facilitate design exploration based on dimensional or geometric variation of a configuration. Although they have limitations in terms of topological variation beyond their preconceived scope at the single part level, prematurely reducing the range of options to explore, multiple assemblies of parts can be created to explore *parallel lines of thoughts.*

Expert Systems

Expert systems address the questions about how to apply existing knowledge in similar situations. They capture specific domain knowledge for detailing and specification (e.g. Structure, MEP, or HVAC). The scope of the knowledge ranges from the description of objects to the rules to adapt them to variable range of conditions. The design expertise is captured in parameters, constraints, attributes of the objects. Although they take advantages from parametric technologies, they are beyond parametric libraries since they do not only define the objects, but also the decision rules, problem solving functions and tolerances to generate custom instances according various conditions (Eastman, Sacks, & Lee, 2003).

The available systems can automate engineering detailing usually by using building massing studies as master models representing design intent as input information (Glymph, Shelden, Ceccato, Mussel, & Schober, 2004). Any change in the

parametric input geometry triggers the re-arrangement the detailing of parts. The tools provide user interface to define parametric custom object and graphical interfaces to define rules and relationships across objects facilitating the usability from the final user perspective. The embedded expertise represents best practices in the field which can be re-used to speed up project developments by automating repetitive tasks of adaptation of already known solutions.

Generative Systems

Generative systems are based on an iterative cycle of application of rules and evaluation of the outcome. This approach creates multiple combinations of inputs values for a model or function. Every combination brings the opportunity for *looking for emergence* of new properties or affordances from the resulting composition. In fact, the unexpected outcomes of the generative process can be considered as apparently creative (Lawson, 2004). The driving algorithms produce large numbers of iterations in a short period of time generating several variations based on the iterative re-adjustment of the parameters and rules (J. McCormack, Dorin, & Innocent, 2004). Examples of this approach can be found in Genetic Algorithms (Frazer, Tang, & Sun, 1999) and Shape Grammars (Stiny, 1980).

Genetic algorithm is a term coined by John Frazer in the 90's. In the genetic algorithm approach design rules are applied to generate a design variant. After evaluating the resulting shape the driving parameters are adjusted and the rules are reapplied to create a new generation of variants. Large number of iterations produce multiple variations and, eventually, unexpected results that can be considered as apparently creative. The iteration mechanism is an approximation to *co-evolving problem-solution* dialog between problem and solution that characterizes expert designers.

The second approach is shape grammars that promotes the idea of creating design variants based on geometric rules and constraints. The grammar essentially

describes a universe of possible combination of geometric entities and in some cases facilitate *looking for emergent* features or geometry derived from the original primitives (Stiny, 1994).. The scope of the combination rules ranges from simple subdivision or linear transformations to more complex interaction with external inputs. Shape grammars are a powerful mean to capture design knowledge through the explicit formalization of design rules driving geometrical transformations. Shape grammars are more flexible than parametric models to capture design knowledge regarding geometric composition since they do not have to deal with hierarchical parametric structures. Some examples of implementation of shape grammars based graphs have demonstrated that emergent features can be captured by creating new connections among nodes of the original ones (Grasl & Economou, 2011, 2013). However, terms of implementation shape grammars are more demanding and labor intensive from the design perspective, because large number of rules and their cross relationships must be explicitly declared (Benros & Duarte, 2009), since implicit knowledge cannot be manipulated. The outcome of the utilization of shape grammars in design is a set of local relationships rather than a top down approach like in parametric modeling. An example of application of shape grammar in design is the project of mass customization of houses based on the grammar of Alvaro Siza's Malagueira houses (Duarte, 2005).The project entails two set of grammars, a discursive grammar that captures the needs and requirements of the final users, and a design grammar that captures rules from the original case study in order to reuse them in the generation of customized houses. An interactive computational model links both grammars to generate the design solutions. What is relevant in this project is that the grammar capturing the requirements seems to be a mechanism to preliminary *framing design problems* that interacts with the grammar of the solution.

Design Languages

This approach allows the production of variety of design configurations by *reusing physical parts*, *applying design rules* and evaluation methods. The physical parts represent the static aspect of the language since they are the primitive elements. The rules and the evaluation method represent the dynamic aspects since they provide the main driving parameters and embed the procedural knowledge regarding how to assemble the variety of parts. These parts correspond to a set of objects from which multiple instances can be created according to the rules. Each object is an explicit description of a physical part, its geometry, parameters, sub parts, constraints and any other relevant attribute that capture what is known about the part (Shah, Paredis, Burkhart, & Schaefer, 2012). This approach is called a language because it is a vocabulary of combinable units. This approach is valid in very restricted design domains that have a limited set of elements and well-defined relationships among them.

Examples of the development of design languages can be found in the car industry (J. P. McCormack, Cagan, & Vogel, 2004). Based on a shape grammars, fundamental features of the front part of the cars and design rules were declared to enable the creation of variety of combinations. These features are the middle and outer hood, the fender, the grill and emblem. Multiple specializations of them were also declared such as rounded or square front grills. An entire catalog of these parts and specializations classified by periods of time was developed and different combinations of parts were identified. Each feature has relationships with each other defining associations such as hierarchical dependencies or adjacencies. What provide flexibility to the grammar is the fact that every individual feature can parametrically and topologically change as long it preserves the integrity of the overall composition facilitating the exploration of *parallel lines of thoughts*.

Another set of examples of this approach that also allow the exploration of *parallel lines of thoughts* can be found in the aerospace industry for the design of

airplanes (Bohnke, Reichwein, & Rudolph, 2009; La Rocca, 2011). The static elements of the vocabulary such as fuselage, wing, flap, nacelle and connector wing-fuselage are described through the implementation of classes of objects. The dynamic aspects or design rules are another set of classes that create instances of the airplane components. A third group of classes represent the static templates of the low level elements of the airplane components such as key points of the geometry, profiles and extrusion guides built from points, resulting surfaces, and finally volumes describing the high level components of the assembly of an airplane. Some of these low level classes are involved in the generation of more than one component. For example the profile class participates in the creation of fuselages, nacelles and different types of wings depending on the number and coordinates of the points defining the profile. During the generation process multiple instances of the static components are created and assigned to different configurations to explore parallel design concepts. The resulting assemblies can also be further refined though optimization. To avoid dependency of any CAD tool during the generation of a design configuration, the syntax of the language is based on Unified Modeling Language (UML) that is the main repository of the parts and rules in terms of objects with attributes. Translators to different tools are implemented to create the actual geometric models form the UML declaration. The benefit of being tool independent is that all the definitions can be related to the specific design domain semantic that capture design conceptuality in very abstract terms.

Agent-based Design

The Agent-based Design (ABD) approach relies on the iterative interaction among active and autonomous entities with discrete information and instructions. ABD is rooted on the notions of the agent and the neighborhood populated by other agents. Agents, usually represented as objects, are entities that function continuously, have appropriate rationale for every step they executes and coexist with other agents and parallel processes (Modi, Tiwari, Lin, & Zhang, 2011). Their behavior can be driven by

two different approaches: pro-activity and reactivity. While pro-activity promotes the interaction and cooperation among agents with a common goal, the reactivity promotes dynamic responses to changes in the environment. Those autonomous reactions trigger the emergence of an intelligence without central reasoning control (Bento & Feijó, 1997). The resulting outcome of all these discrete interactions through time eventually can produce unexpected remarkable results or easily fall into constraint circularity and non-sense loops. Avoiding those issues is very difficult for most of the implementations since it requires to predict future possible scenarios many steps ahead. Another approach is the implementation of learning mechanisms that improve the responses of the agents while they get more experience (Shen & Norrie, 1999). For example, Bento & Feijó propose a long term memory implementation to support this leaning mechanism. That declarative memory remembers facts provided by the user or collectected along the interactions that are used by the procedural component of the agent to make decisions or even make corrections to the procedural knowledge thei manipulate.

ABD have been mainly applied for simulation, form finding and optimization since these processes essentially are searching mechanisms and do not necessarily anticipate or predict the results. Agent-based simulation have been implemented to provide feedback to spatial layouts under design under the assumption that spatial configuration affects the movement pattern. For example, Penn and Turner (2001) have implemented agents representing occupants that have access to pre-computed graphs that represent the space in terms of nodes that records what is visible from any point on the floor plan layout. This allows to simulate space visibility and its impact on occupant's behavior. Different set of rules driving the agents facilitate the exploration of potential patterns of movement through the space providing a sort of *preliminary evaluation*s of the layouts.

Baharlou and Menges (2013) have explored ABD in form finding methods embedding material, fabrication and geometric constraints into agents that subdivide double curvature surfaces according fabrication purposes. In the same way, Cui and Turan (2010) has been also applied ABD to solve multi-objective optimization problems in ship design by capturing different the trade-offs across technical constraints. Although the reutilization of these chuck of well-known constraints is not an actual recalling mechanism, it provides alternatives to the hull design because of the interactions of agents that represent those constraints. In summary, ABD seems to contribute to reuse the knowledge about different trade-offs among constraints and also, if successful, it improve the ability of designers of *looking for emergence* of new features or behaviors along the process.

Case-based Design

Case-based Design (CBD) is a subset of Case-based Reasoning (CBR) that uses the mechanism of *recognizing problems* to understand and solve new problems based on previous solutions (Goel & Chandrasekaran, 1992; Pearce et al., 1992). It can simply reuse an existing solution, suggest adapting or combining solutions, prevent potential failures or interpret the situation (Kolodner, 1992). This mechanism also save time finding a solution by allowing the rapid generation of plausible ones, even though they need further validation. According to Kolodner the effectiveness of a case-based reasoner depends on the accumulated experience or cases, the ability to connect new situations with previous ones based on similarities, the ability to adapt or combine old solutions to new problems and the ability to adapt the evaluation of the outcomes in order to avoid making the same mistakes indexing the situation by failures in the past.

CBR has two main approaches: Problem-solving and Interpretative. While problem-solving attempts to literally reuse and adapt old solutions, the interpretative provide a classification of the situation, an evaluation of a solution, an argument or in some cases an actual solution. Both approaches depend on remembering a learning

mechanisms since they become more efficient by saving time adapting old solutions and competent by expanding their repository of experiences providing more suitable solutions.

In terms of implementation the problem is defined as a set of constraints that are satisfied in order of importance. The system records the solution and use it as starting point for a similar problems by adapting the solution, combining solutions or relaxing the constraints of the problem. CBD is the particular application of these techniques of CBR developed by Computer Science and Artificial Intelligence to address design problems. This approach attempt to adapt or combine previous designs to new environments or functional avoiding the reproduction of solutions from scratch. It has also proven to restrict the search space for solutions by restricting the exponential growth of design alternatives derived from combinatorial of parts or features (Schmitt, 1993). It also preserves the implicit trade-off among requirements that are assumed to be solved in the invoked base solution.

CBD systems developed in mid-90's created complex design based on small case repositories, preserved the implicit solutions from different trade-offs, recorded the history of the designs, and indexed new cases. Although the restriction to the creativity is arguable because of the endogamic combination, one of the most salient characteristic of CBD is that it is a process that facilitates reusability and continuous growth of the *Design Domain Repository*. However, they also demonstrated limitation in terms of learning and problems synthesizing a large number of constraints (Mary Lou Maher, Balachandran, & Zhang, 1995; Mary L Maher & Pu, 1997). Heylighen and Neuckermans (2001) provided a detailed assessment of the CBD tools for architecture under the underlying framework of cognition: structure and organization of knowledge, reasoning process and learning. The scope of the applications of the systems they evaluated include retrieval of relevant design cases and precedents, preliminary building designs, complex building installations, layout designs and support for analysis

and evaluation for conceptual design. According to the first criteria, knowledge is structured mostly in specific, general and adaptive and they are mainly organized by indices, hierarchies, functions and databases. The reasoning criteria is based on retrieval, adaptation and merging and contributed to progress in the actually implementation of *recalling* mechanisms. However, the third and the most characteristic mechanism of learning has still very limited implementations and relies mostly of leading the systems with external cases to enrich their knowledge repository.

2.2. Solution Evaluation

Solution evaluation approaches are focused on performing evaluations and analysis of solution candidates, providing valuable feedback to support decision making. The notion of computer as design critic (Lawson, 2004) efers to all kind of estimations, analyses and performance evaluation that can be executed. We can recognize multiple tools to evaluate variety of aspects. Some of them ask to the user to provide input variables for the evaluations. Others derive the information to perform evaluations automatically from the model that also facilitates the execution of parallel analyses. Although both approaches have differences, they have in common the need of a design draft.

The challenge is to perform the evaluation early on along the design process, because most of the current evaluation came too late in the design process to effectively participate in the design itself. The approaches reviewed in this subsection range from the need of a design draft to perform evaluations to parallel development of the rules and the solution. They are Performance Evaluation, Rule Checking and Constraint-based Design.

Performance Evaluation

The Performance Based Design (PBD) paradigm characterizes how a product executes a given function under stress (Becker, 2008). In other words, it is focused on

what the product should achieve instead of how it should be built (Kalay, 1999). The most common performance evaluations are energy consumption, natural lighting or structural performance. However, multiple analyses can be performed in other technical areas. Although capturing the knowledge to execute performance analyses implies large amount of information and complex calculations, it entails mostly explicit declarative and procedural knowledge usually found in handbooks. For example, Szokolay (2008) documented a complete review of several methods to assess thermal comfort, energy consumption, HVAC systems, natural and artificial lighting, noise control and acoustic design and even water consumption. The aim of these evaluations is obtaining feedback for design decision-making from early to late design stages.

Two main approaches support *integrating knowledge* from other domains in the design and evaluation process: Embedding knowledge into commercial tools and into custom functions. The commercial tools, in turn, are divided also into two: Those that are specialized in specific analyses that need a geometric model as inputs provided by other modeling tools, and those that having a geometric modeling engine integrate analysis packages in the same environment without need for data exchange with external tools. Although these tools facilitate and in some cases automate *applying evaluation methods* from the design domain knowledge repository, they have two important limitations derived from interoperability issues and the uncertainty derived from the evaluation results. The first one is due to the exchange process design and evaluation tools that affects the natural iterative process between both in the designer's mind. The second one, is due to the fact that most of the evaluation tools are literally black boxes that hinder the criteria, assumptions, relationships, and calculations involved in the process (Bernal, 2011). It implies that different tools can provide variations in the results depending of their engines that do not necessarily implement the same methods affecting the reliability if the outcomes.

In terms of embedding technical knowledge to perform evaluations in custom functions or applications, most design and engineering computational tools allow custom implementation through their application programming interfaces (API). Such capabilities facilitate deriving values form the models, feeding the customs functions and re-computing the results without any external data exchange rather than using the results to manually modify the designs. This customization of the tools can capture best practices and provide real-time feedback to support decision making.

Rule Checking

The aims of rule checking systems are automating the evaluation of the fulfillment of rules, providing feedback regarding conflicts and linking them with the broken rules especially in large projects where it is difficult to track all the rules by human reviewers. Rule checking systems entail four phases: the rule interpretation, model preparation, rule execution and reporting the results. The first step is capturing the essence of the rule, considering that the rules can represent critical requirements, design guidelines, or norms (Seebohm & Wallace, 1998) . The rules usually come in textual form and must be translated to machines. Parsing techniques facilitate such translation. The translation can be at least in two forms: predicated which will be evaluated true or false, or ontologies of names and property which first identify if the condition exists (e.g. emergency exit) and then if its properties fulfill the requirements (e.g. exit width).

The GSA design rule checking system (Design Assessment Tool, DAT, developed by Georgia Tech) provides an example of rule circulation path checking. Best design practices have been collected in a design guideline by the GSA which has been translated into computable parametric rules. (Eastman, Lee, Jeong, & Lee, 2009). The second step is the generation of a graphs circulation model based on a BIM model of the buildings, the first model represent the connectivity between spaces, and the second the walking distances across the building (J.-k. Lee, Eastman, Lee, Kannala, &

Jeong, 2010). The rule execution evaluates if the circulation paths satisfy the security requirements. Finally, the system reports the conflict areas through visualizations and by providing a reference to the textual version of the violated rule. Such feedback is later used by designer for modifications. The limitation of this approach is that it provides reports of the conflicts but they do not participate directly in the design process, since the designer most interpret the reports and provide the solutions. The challenge is how to actively integrate rules in the design itself.

Constraint-based Design

Generation of solutions based on constraints provides a possible way for integrating and applying design rules in the design process and implement techniques for recalling chunk of constraints that designers collect to build design solutions. Design knowledge is captured in constraints and rules representing requirements that must be satisfied by the design. The scope of knowledge embedded in such constraints could represent standard or custom conventions as well norms. These rules or constraints are assigned to objects that will be combined later on. Some systems provide feedback during the design process if any rule is violated, whereas others search for a combination of constraints that satisfy the design problem.

Examples of real-time feedback can be found in tools for floor plan layouts. Different restrictions in terms of adjacency or dimensioning can be assigned to room types. During the instantiation and layout design if any of these rules is broken immediate feedback is provided. Example of commercial product to design preliminary floor plan layouts interacting with rules defined by the user can be found. Graphic interfaces facilitate the definition of the parameters and properties of the spaces and the relationships among them. Users can introduce the rules by assigning constraints or programming custom functions. The elements of the layout as well the rules can be edited all the time. Such flexibility created a scenario with the conditions for *co-evolving*

problem-solution. This approach is limited for scenarios with a small number of variables and loses flexibility as the complexity of the rules and constraints arise.

Regarding the utilization of constraint based approach to automatically generate design configurations, Examples can be found in the design of complex mechanical parts (Yvars, 2010) and building systems (Medjdoub, 2009) that entail large number of trade-offs among constraints. Research efforts are focused on the creation of models representing the constraints that are combined by searching algorithms that operate under the constraint satisfaction problem (CSP) approach. The searching addresses the problem of finding the proper configuration and dimensioning of parts that satisfy the constraints for give problem. This particular approaches make progress in the creation of arrangements of chunks of well-known constraints that represent best practices and parts of the solution as well.

2.3. Solution Selection

Decision making is the third component in the cycle generation, evaluation and selection (Mela, Tiainen, & Heinisuo, 2012). Research efforts have been mostly devoted to computational implementation of generation and evaluation procedures. However, there is limited research on the post-optimization decision making area (Mourshed, Shikder, & Price, 2011). Most of the methods attempt to reflect values and preferences through quantitative indicators complimented with relative wrights of importance. If these weight change, the selection of best candidates will also change. Current evaluation and optimization tools gradually provide more reliable performance indicators. Nevertheless, the indicators do not necessarily represent the actual process of decision making, which is far more complex than adding or multiplying the values derived from the weighted combination of indicators. Actually, the quantitative methods only operate over the declared indicator and weights. Therefore, the reliability of the decision is depending on the completeness of the model and the human ability to

interpret results. This section will review computational attempts to implement three approaches to support for selection and decision making: Multi-criteria Decision Making, Optimization and Utility.

Multi-criteria Decision Making

Multi-criteria based approach for selection of solution candidates establishes metrics to evaluate how the design fulfills given objectives. Qualitative requirements are translated into quantitative values, and acceptable limits of fulfillment of them. This approach aggregates data to quantify performance, which may also be normalized to establish a level of comparison of different aspects (Augenbroe & Park, 2005). It has three essential steps: identifying User Needs (UNs), transforming UNs into Performance Requirements (PRs) and quantifying the level of satisfaction of such criteria.

UNs correspond to the formalization of requirements organized into groups such as functionality, safety, health, or sustainability. The definition of the user expectations can be achieved through two main methods: target oriented (TO) to express specific achievements, and failure preventive (FP) to prevent risks and interferences with others building functionality aspects (Becker, 2008). Indeed, UNs are usually defined by combination of both.

PRs correspond to the expression to describe the more general UNs desegregated into performance criteria, the relationship between quantitative values of relevant physical factors used as performance indicators. These factors are used to predict outcomes during the design process, and the evaluation of a given solution is based on quantitative levels of satisfaction of such performance criteria. Finally, the results must be ranked and prioritized to select a solution candidate. This step is particularly important, since multiple trade-offs exists among aspects of the project (Keeney & Raiffa, 1993). For example, desired hours of natural lighting versus avoiding green-house effect. In this regard, different users could have different interpretations

based on the same indicator, since they can assign different weight or importance to those indicator depending on their individual goals and priorities

The challenge to this approach is how to balance the need for explicit declaration to execute the evaluations with the ambiguity of the design definition in early design stages. Multi-criteria approaches have flexibility to measure fulfillment of multiple requirements by solution candidates. However, because the comparison is based on quantitative values, it is difficult to apply such methods in early design stages when precise information usually is not available. Further, these approaches do not explicitly handle uncertainty in information or the preferences of decision-makers. A custom example of this approach can be found in Hopfe, Augenbroe, and Hensen (2013). They use a case study that illustrates the process of making choices between two different HVAC systems that also include uncertainty information about the designs, by ranking different performance indicator based on stakeholders' preferences.

Multi-objective Optimization

Optimization methods, originally developed in Mathematics and Operations Research, enable designers to define and search through large spaces of design variations. The generation of those spaces essentially rely on iterations that produces the volume of possible solutions. This exploration requires parameterizing the geometry according to the target evaluations representing the variety of objectives and implementing algorithms for continuous variation until reaching the solution candidate (Kasik, Buxton, & Ferguson, 2005). The resulting solution space grows exponentially with linearly increasing input-output parameters and the number objectives to be satisfied (Koch, Simpson, Allen, & Mistree, 1999). To address this growing complexity, researchers have generated a wide array of methods including decomposing design problems into sub-problems, screening significant variables, reducing design space based on heuristics, mapping, and visualization (Kleijnen, 1997; Shan & Wang, 2010).

Currently, all these methods take advantage of parallel computing and increasing computer power.

The type of knowledge embedded in these parametric models and algorithm captures best practices to address multi-criteria design problems. The users can play different roles in the knowledge capturing process and re-utilization such as providing the design expertise, implementing the algorithms representing the expertise, and generating and evaluating design alternatives. An example of this kind of systems is the Performance-Based Generative Approach developed by the Engineering Design Centre of the University of Cambridge and Bentley Systems (Shea et al., 2003). The approach combines Custom Objects (CO), a predecessor of Bentley Generative Components parametric tool, and eiFrom a Generative tool, exchanging information through XML models. CO is an associative and object-oriented-based parametric modeler to support design, and eiForm in a generative environment to perform performance evaluation, and optimization. While CO captures design intent trough all set of parametric relationships, eiForm performs structural analyses with multiple objectives, search for optimized variations of the original input geometry provided by CO and send back an improved version of the original input geometry. The iteration of this process facilitates *integrating knowledge* during the design process. Another example is the application of Genetic Algorithms for the optimization of building envelopes and the design of consistent HVAC systems (Caldas & Norford, 2003).

While the iteration process facilitates some negotiation between design and engineering expertise during the design process and allows the visualization of the trade spaces (Flager et al., 2009), standard optimization approaches do not explicitly deal with uncertainty of information and preferences of decision-makers, and they require a high level design specifications before design decisions can be made.

Utility

Utility methods attempt to formally describe the preferences of the decision maker. Choosing by advantages (CBA) and Collaborative weight, rate and calculate (WRC) are two distinctive methods for such purpose.

CBA makes the relevant attributes and advantages of the solution candidates explicit. It also sort them in order of preferences by ranging from features that the design must have to those that are desirables. (Parrish & Tommelein, 2009; Suhr, 1999). WRC methods model the varied preferences amongst stakeholders (Haymaker, Chachere, & Senescu, 2011). However, CBA, and WRC have been criticized for the way they capture stakeholder utilities, and for their inability to consider uncertainties. Normative decision theory provides a rigorous foundation for how one should go about making decisions if one wishes to be rational, i.e. consistent, with one's elicited preferences (Keeney & Raiffa, 1993; Pratt, Raiffa, & Schlaifer, 1964; Von Neumann & Morgenstern, 1945), although how to address multiple criteria and stakeholders, and how to accurately capture a decision makers preferences, remain active areas of research.

Limited examples of computational implementations can be found. Nicknam, Bernal, and Haymaker (2013) used a web-based tool to explicitly declare de advantages and disadvantages of several solution candidates for the same design problem. The interface allow interactively design the profile of the desired solution as a rudimentary *co-evolving dialog problem-solution*. However, the major limitation is the ambiguity of some criteria such as those related to aesthetics that ae highly subjective or difficult to elucidate, the incompleteness of the attribute models, and the lack of responsiveness of the parametric models that require manual editing for major changes.

2.4. Integration of Generation, Evaluation and Selection

Although solution generation, evaluation and selection correspond to different aspects of the design tasks, the current tendency is gradually integrate them into common environments to facilitate their interaction and better support *integrating knowledge* from different domains. Three main approaches addressing this dispersion in different ways will be presented and discussed: interoperability based on data exchange, system integration based on platforms for interaction of systems for complementary purposes and custom integrations system-to-system.

Interoperability

Interoperability rely on data exchanges among systems based on industry standard. It allows passing information from one tool to another avoiding or minimizing the errors derived from human reinterpretation of paper based information (Tarandi & Froese, 2002). Interoperability supports the process of integrating knowledge regarding design, performance analyses, cost estimations, fabrication, scheduling, mechanical systems or rule checking among other sources by facilitating the transit of information across different purpose tools that add value to the design along the process.

The Industry Foundation Class (IFC) standard based on the STEP standard for engineering design satisfies the needs for exchanges in the Architecture, Engineering and Construction (AEC) Industry (Eastman, Teicholz, Sacks, & Liston, 2011), and it is one of the fundamental technologies supporting Building Information Modeling (BIM) paradigm. The IFC inherits the object-oriented data model from STEP that enables modeling building parts, their geometry and material properties, scheduling and other relevant attributes for different kind of analyses. The IFC provides a neutral file format that enable sharing information among computer applications. The IFC captures industry standards and it is constantly extending the schema to include new objects. For this purposes different groups of interest develop their standards extending the scope the IFC schema. In this regard, the exchanges among different disciplines

involved in the AEC industry are mapped and explicitly declared along the entire life cycle of the project identifying exchange requirements that are mandatory, optional or not needed. Examples can be found in the industries of precast concrete (Eastman et al., 2003) and structural steel (Danso-Amoako, O'Brien, & Issa, 2004; Eastman, Jeong, Sacks, & Kaner, 2010; Lipman, 2009).

Although the exchanges are focused on the descriptions of physical parts, their attributes and metadata, recent research efforts filter groups of components and attributes that belong to the same system or view of the model (Venugopal et al., 2012), similar to the notion of *conceptual structures* in design. Those Semantic Exchange Modules (SEMs) represent aspects that are distributed across parts. Beside the progress in capturing these abstract structures, current exchanges are mainly sequential and have limitations to represent the simultaneous interaction of different sources of knowledge while designing.

Model-based System Integration

This is an interdisciplinary approach to enable the realization of products and systems (Friedenthal, Moore, & Steiner, 2011). Although we can find platforms that use different wrappers to make interact complementary tools for design and evaluation (Flager et al., 2009), we will focus on the model-based approach that use meta-models, intended as the model of the attributes of the actual model, to capture domain knowledge and share it with different systems.

Model Based System Engineering (MBSE) in engineering design is focused on the development of meta-models as the main design domain knowledge repository. The meta-model captures the design knowledge through multiple kinds of representation: class definitions, associations, sequences of operation, description of activities, typical use cases and parametric relationships. MBSE is an approach that supports the formalization of requirements, the development of designs, and analysis for verification. It starts in the preliminary design stage, continues through design

development and often supports monitoring the life cycle of engineering products. The fundamental component of this approach is the meta-model that represents aspects of the product such as its physical structure or expected behavior. From the meta-model, variable number of configurations with different topologies can be specified depending on the design requirements. The meta-model became the means of communication across disciplines (Reichwein & Paredis, 2011). It is the central repository of diversity of knowledge and is the source of information for multiple sub-systems such as analysis or CAD tools. For that purposes it remains as abstract as possible preserving the integrity of the design high level conceptuality. For example, from the design domain perspective the declaration of airplane wing and its related attributes can be strictly based on design terminology. The representation of the wing for an engineering analysis or CAD modeling could vary adding more detailed specification. While different expert designer can agree in what a wing is, different analysis or CAD tools have different way to represent it and all these internals details should not be registered in the general domain declaration. The domain model does not need to know about the internals of any representation, and the representation does not need to know the internals of another representation. These levels of abstraction allow distilling the fundamental design knowledge at the model level by separating it from complexity of the geometric representation.

The motivation behind this approach is capturing all the domain specific semantics (Eck & Schaefer, 2011) in very abstract terms that can be reused in different designs and refined and extended from one project to another. The supporting languages are derived from the software engineering field; specifically UML and the derived System Modeling Language (SysML) both object oriented languages (Kifer, Lausen, & Wu, 1995) for systems developments.

Custom Integration

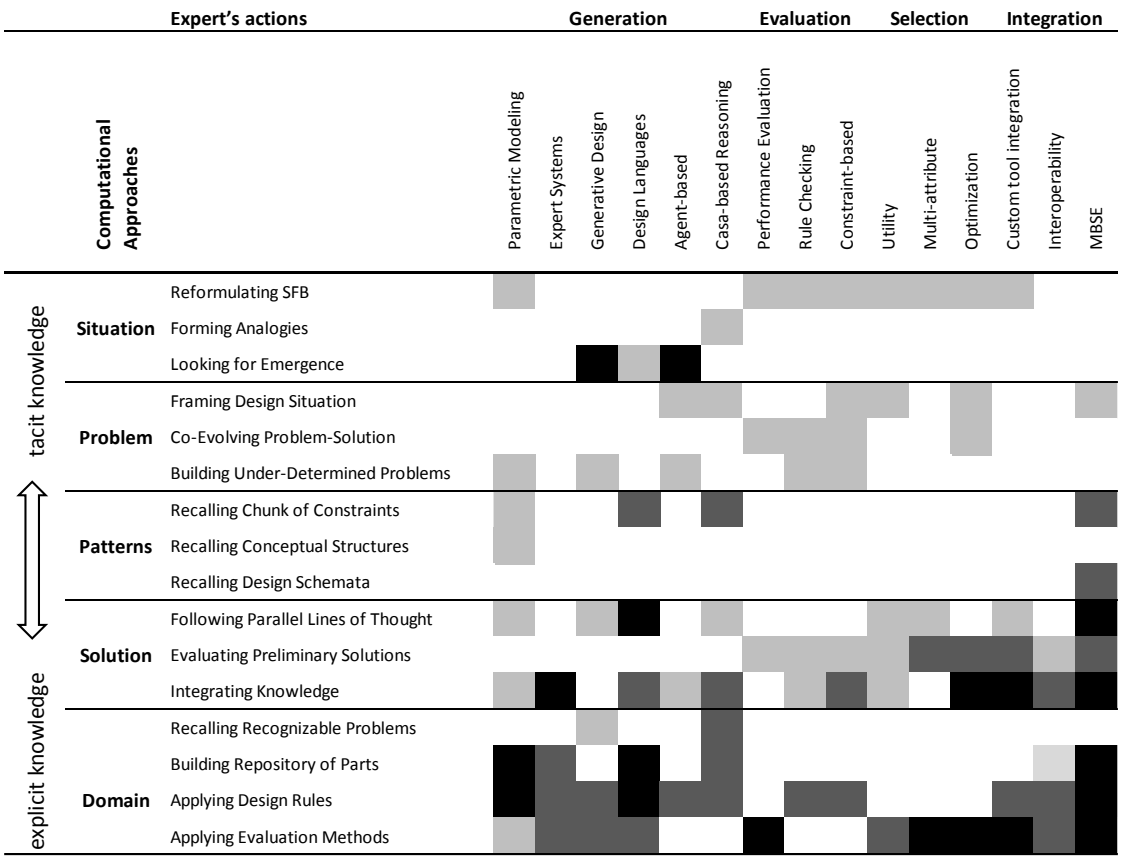
Currently, we can fins custom implementation coupling performance assessment with parametric modeling. An academic example can be found in Sanguinetti, Bernal, El-Khaldi, and Erwin (2010), in which a parametric model of a retrofit project was linked with a performance calculator implemented in spread sheets for energy consumption, natural lighting, glare index, and payback period. The performance calculator received inputs from the user such as material specifications and directly from the parametric model (e.g. dimensions, volumes, orientation, shading areas, or opaque and glassing surfaces). After every design modification of the parametric model the four performance indicators are updated in real time, providing valuable feedback for decision making. These custom implementations will drive the emergence of a new generation of design tools embedded design and engineering constraints within parametric systems.

A commercial effort is the case of Vasari by Autodesk (Autodesk website, 2012), which is a low resolution tool to evaluate aspects such as energy performance. The analysis package is directly linked with a massing model of the building, eliminating exchanges and improving the interactivity between design and evaluation in early design stages. Although these two examples do not modify the design, they are constantly monitoring the design process to provide feedback to the designer.

2.5. The Scope of Computational Support for Designers in Action

Table 3.2 characterizes the role of computational approaches described above in relationship with the actions performed by designers (white human-based, light grey computer-aided, dark grey computer-based and black computer-augmented) Based on the literature review, It identifies actions that currently are mainly human-based -- lacking of assistance; computer-aided -- providing valuable feedback or facilitating tasks; computer-based -- automating processes; or computer-augmented -- extending the designer's compatibilities to potentially improvement of design quality. This classification and discussion does not imply an order of preference, or attempt an exhaustive analysis of all possible combinations. Rather the matrix attempts to provide a panoramic view of relationships that will help to visualize tendencies and formulate research questions.

Table 3.2. Classification of designer's actions based on computational support (from Bernal et al. (2015))



Human-based actions

Although most designer actions have some kind of tool support, those related to *Interpreting Design Situations* and *Recalling Patterns of Organization* are mainly human-based with minimal assistance to the designer's cognitive needs (Goel, Vattam, Wiltgen, & Helms, 2012). Even though some research effort exists (Davies, Goel, & Nersessian, 2009), design situations related to *Reformulation* and *Forming Analogies* lack of support due to the difficulty of computing unexpected relationships and inferring in shifting directions. Computer programs have not yet demonstrated human-level ability to link apparently non-related things and build a logic interpretation (Dreyfus, 1992; Goldscmidt, 1988; Lawson & Dorst, 2009). *Design Languages* and *Case-based Design* capture some of the geometric relationships useful for recalling and adapting previous solutions, however, the human designer is still needed for the actual assessment and selection of these patterns.

Computer-aided actions

The reviewed research efforts demonstrate that computers partially support *Interpreting Design Situation*, *Formulating Problem* and *Generating Solution*, but they have several limitation to support designers' abilities for synthesis and critical thinking that typically drives these actions (Cross, 1990). *Generation* oriented computational approaches incorporate parameters and constraints according to gradual definition of the situation frame, handle some degrees of under-determination of the problem in terms of under or un-constrained relationships, and facilitate the production of models of parallel alternatives. The main contribution of the *Evaluation* and *Selection*-oriented approaches is providing multiple performance assessments and normalized comparisons of features of alternatives that inform decision-making.

Computer-based actions

Computers can follow algorithmic processes that evaluate and search through design spaces, or perform heuristic search based on external knowledge provided by

experts, to narrow down the search space to only feasible solutions. Although taking advantage of these computational approaches to automate creative tasks is a desirable target, computer-based automation in design mostly consolidates explicit domain knowledge into standard procedures. Automation is concentrated in actions that capture and reuse procedural knowledge related to *Design Solutions* and *Design Domains* rather than addressing more challenging issues such as *Forming Analogies* or *Co-evolving Problem-Solution*. Generation-oriented approaches mainly automate the well-defined domain knowledge in terms of Application Design Rules, building repositories of *Reusing Physical Parts* and *Applying Evaluations Methods* during the actual generation process. Automated design is appropriate for applying well defined knowledge, especially for detailing. Similarly, *Evaluation*-oriented approaches automate standard evaluation procedures that support *Integrating Knowledge* from different fields in the sense that make them interact triggering a cross criticism among project aspects.

Computer-augmented actions

Beside some moderate success in the *Interpreting Design Situation* category, specifically regarding augmenting the potential of *Looking for Emergence* of new design features derived from the use of iterative algorithms, most of the augmentations in the literature are related to the *Generation of Design Solutions*, *Developing the Design Domain*. In terms of solutions the main achievements are *Following Parallel Lines* of thoughts and *Integrating Knowledge*. *Design Languages* and *Model-based System Integration* can expand the design space from the boundaries of parametric variations to a family of topologically different configurations based on combination of parts that can effectively support the exploration of parallel alternatives. In addition, optimization approaches, custom integration, and *MBSE* actively addresses the negotiation among criteria derived from multiple aspects of the design. Recent progress focuses on capturing and reusing explicit knowledge of design domains. *Parametric Modeling*, *Design Languages* and *MBSE* facilitate building repositories of parts and support the

reutilization of already known solutions representing best practices. Finally, the approaches that allow the application in real time of explicit design rules and evaluation methods augment the reliability of designers' choices and decision-making.

2.6. Discussion: The Problem Reusing Design Expertise

The problem reusing design expertise to support the designers' action have been addressed from different perspectives by computational means for generation, evaluation, selection and integration of all of them. The reviewed approaches show different degrees of success in the manipulation of design knowledge and also leave open research areas. Computation research has made little headway in assisting the manipulation of tacit knowledge related to qualitative aspects of the design task, due to difficulty representing these implicit assumptions in design decisions. These limitations are concentrated in the actions that frame the focus of interest, trigger unpredictable evolutions and provide coherence to the design. On the contrary, progress can be found in the automation and augmentation of actions to manipulate explicit knowledge from design domains to produce solutions.

The hinge between the tacit and explicit knowledge seems to be the patterns of organization that represent the underlying relationships that designers establish across parts and sub-systems. While physical parts are driven by parameters and constraints, conceptual structures organize arrangements of these parts and *Design Schema*s drive the overall integrity of the design. Although the ability to recall and adapt all these abstract entities is an important skill of designers, current computational tools offer limited support to represent and manipulate them. Addressing the manipulation of these patterns is a key to facilitate the reusability of design knowledge. In this regard we can find some progress in the manipulation of parameters and constraints, but conceptual structures and *Design Schema*s are still open challenges.

Despite the limitations to provide support to the most sophisticated actions that designers execute, integration approaches offer opportunities to progress assisting designers in actions, since they integrate many of the features of diverse computational systems. Besides the custom initiatives system-to-system, the two major approaches based on interoperability and system integration facilitate the interaction sharing information declared in different formats for multiple purposes. Interoperability supports design development by allowing variety of file exchanges for evaluations that effectively inform the evolution of the design. However, it has lower impact on conceptual design when the information is not well structured or not captured in the standard schemas such as IFC. On the other hand, Systems Integration provides methods to augment most of the explicit designer actions, and it is perhaps most suitable to address the conditions identified to capture design expertise: definition of the primitives, declaration of the patterns of organization and building the repository of design knowledge. The systems integration approach based on meta-models allows modeling the design domain capturing design knowledge, implement mechanisms to generate design alternatives and also perform preliminary evaluations based on the available attributes.

Meta-models of Design Domains

Meta-models as the main knowledge repository can satisfy the needs for capturing structural and behavioral aspects of the system of interest through multiple kinds of representation such as definitions of objects, associations, declaration of types, sequences of operation, description of activities, typical use cases and parametric relationships. The use of non-system specific languages, such as the well-known Unified Modeling Language (UML) or the System Modeling Language (SysML) that is an extension of UML for system engineering purposes (Friedenthal et al., 2011) allow independency from any computer program. From this independent knowledge repository multiple design configuration or arrangements of parts or assemblies can be specified. It is important to establish the distinction between specification and

generation. While generation implies the production of the final geometric model of the design specification only implies the definition of the features of the model. The hypothesis is that the resulting specification for the *Design Schema* as well the propagation of physical parts can be generated in domains specific terms for further interpretation and geometric representation of the auxiliary geometry as well as the solid models of parts and assemblies.

Generation of Design Alternatives with Different Configurations

Expert designers produce small number of possible solutions in early design stages representing parallel lines of thoughts. These alternatives are compared with the initial problem formulation. The dialog problem-solution facilitates the definition of the problem and the selection of the path of the design development. This mechanism demands rapid generation of design alternatives. However, current design tools have limited capacity to support variations beyond the scope of parametric models, limiting the generation of variety of possible solution candidates. Every variation that implies changes in the configuration implies that a new model must be manually created. Furthermore, designers tend to use sketches rather than CAD or BIM models for early design exploration because in the same environment they can efficiently explore and reflect about parallel trajectories of development, compare options and make decisions. Achieving such flexibility to produce topological changes of the resulting configurations challenges the hierarchical structure of parametric models in terms of their capability to recombine parts within an assembly to create new configurations according to the needs of the problem. The hypothesis is that the separation between the configuration specification and the geometric representation extends the flexibility to explore design candidates with different topologies. Producing the configuration specification outside of a CAD or BIM tool avoid the complexity of parametric structures which easily run into inconsistency, under or over constrained situations during design modifications that Eastman et al. (2011) well describe while manipulating parametric models.

Performing Preliminary Evaluations

Another important characteristic of expert designers is that they integrate knowledge across fields and perform rough evaluations of different aspects of the possible solutions. They develop aesthetics and technical aspects simultaneously. Most of the preliminary evaluations of the designs are driven by heuristics derived from previous experience, which are mostly low resolution estimations based on simple methods. These heuristics range from simple rules for dimensioning physical components to rough estimation of performance. Coupling generation and preliminary evaluations implies addressing the problem of partial definitions or incomplete information of the early design drafts. Currently, design and evaluation tools are not fully integrated to facilitate such tasks. On the contrary, most of the evaluation tools need geometric models as inputs to perform the analyses. It implies that the design must be defined in advance. However, not every evaluation requires information form geometric models. Some evaluations can be performed using the driving and simple derived parameters of a model. In addition, the heuristics varies from expert to expert and most of the time correspond to a set of custom methods to make decisions. For example, in the specific field of custom façade systems deciding between stick or unitized systems, allocation of the water barrier, dimension of the grid of the façade according to the module of the building structure, definition of the interval of the brackets, preliminary dimension of the section of linear elements according to the spanning between floors among others, are based on rules and assumptions derived from previous experience.

The speculation is that preliminary estimations of the resulting configurations as well as decision making could be performed based on driving and derived parameters of the solution configuration. The metamodeling process facilitates the access to the main parameters of the resulting configuration. Ether evaluations or dimensioning can be performed before any geometric representation take place.

CHAPTER 4

DISTILLING DEISGN KNOWLEDGE

Overview

Three case studies from the expert field of custom façade systems constitute the empirical context of this research on capturing and reusing expert design knowledge for the generation of design alternatives. The case studies were provided by Marc Simmons, Ventulett III Distinguished Chair in Architectural Design at Georgia Tech, who is renowned for the implementation of high-end building skins at Front Inc., (http://www.frontinc.com/), an international façade consulting firm. The projects are examined from the perspective of various aspects of this design domain to identify objects and look for evidence of design actions that have been performed. The aim is to capture the structural and behavioral components of expertise in this particular field along with the design requirements. The structure represents physical or abstract entities of the domain, and the behaviors represents actions that the designer executes to assemble a coherent arrangement of objects that satisfy design requirements. The design knowledge, actions that manipulate such knowledge, and a series of problem requirements were distilled from transcriptions of the explanation of each case study by the expert designer. The distilling process (Figure 4.1) was based on techniques of the verbal analysis complemented by digital documentation. It is intended to identify objects that embeds knowledge and will constitute the meta-model of the Domain. The characterization of the case studies poses specific challenges regarding the implementation of methods that enhance computational support of designers in action. The following sections will introduce the field of custom façade systems with a set of selected problems within the domain, present the methodological approach for distilling design knowledge, and review the challenges of modeling the domain, including

physical parts, underlying design patterns and requirements shared across different façade systems

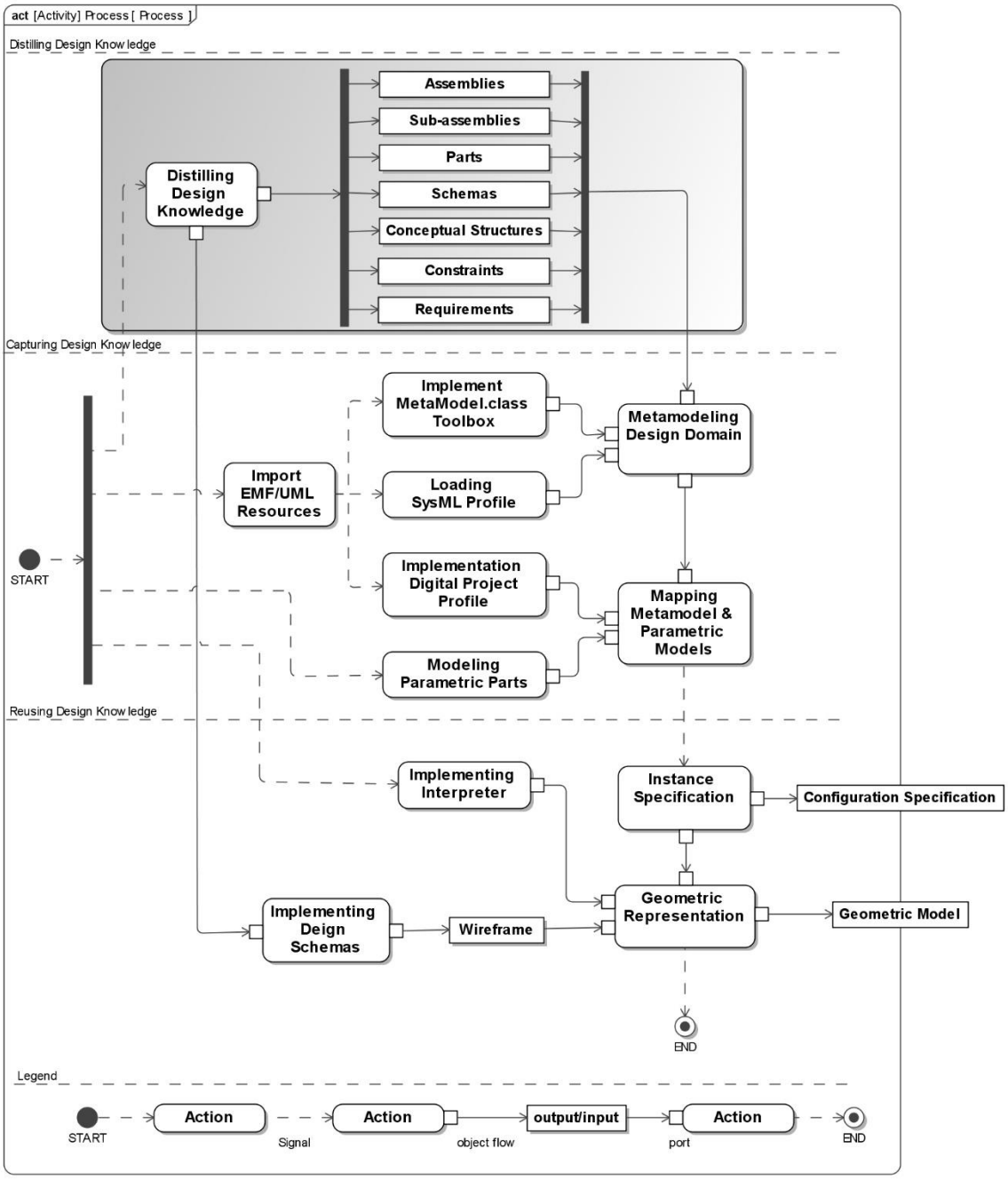


Figure 4.1. Distilling design expertise in the context of the overall process

4.1. Design Domain Knowledge of Custom Façade Systems

While chapter two discussed design expertise from a theoretical perspective, this chapter addresses the same problem from an empirical perspective. For such a purpose, three case studies in the field of custom façade systems have been selected to search for, identify, capture, and represent design knowledge. The selection of this particular domain is based on a limited number of components combined in the production of solutions, similar challenges that systems have to address, and the possibility of building a comprehensive integrated model by collecting and organizing knowledge from distinct façade systems through time.

The building façade system domain is typically classified in two main groups from both fabrication and installation perspectives: stick walls, which are assembled on site, and unitized systems, assembled off site and lifted on site for installation. From these two categories, multiple types of façades can be designed for various purposes, such as green walls, *brise-soleils*, rain screens, or kinetic walls. These systems are assemblies of physical components made of a variety of materials such as steel, aluminum, wood, concrete, glass, rubber, and plastic, among others, in all combinations. These material systems also support several aspects satisfying functional requirements.

The complexity of façade systems starts with the partitioning of the regions of the building envelope. Diverse inputs such as the sizing of parts, the modularity of the structure of the building, mechanical systems allocated above the ceiling, mechanical floors, or variations in the floor-to-floor height affect the subdivision of the façade. In addition to partitioning, the façade must transfer loads to the main structure of the building, address water proofing issues, define the dimensions of the components and the sequence of installation, or balance heat gains with natural lighting and glare. All of these aspects or perspectives of the same façade system persist across systems and depend on single parts and attributes distributed across the entire facade. These

aspects can be addressed by many common strategies. Such a dichotomy of diversity and commonalities is the main motivation for declaring and capturing knowledge defining the arrangement of physical components that simultaneously supports several functional aspects. Furthermore, most of these aspects are strongly interrelated though all a number of cross relationships that determine the nature of the components and assemblies of the components from multiple perspectives.

The selected case studies provide context from which one can distill design knowledge and better understand how design schemas drive the propagation of physical parts to produce different configuration of prefab panels, and also how to generate and control such a pattern of organization. While this chapter focuses on distilling design knowledge from these three case studies, the next chapter explores the integration of knowledge in a comprehensive model.

Case Study 1: Seattle Central Public Library

Conceptually, the Seattle Central Public Library, originally designed by OMA and LMN Architects, is a series of suspended open boxes with interstitial spaces between them. A diagonal grid envelops the entire composition (Figure 4.2). The pattern of the grid, the focus of interest of this case study, determines the structural approach, the arrangement of the parts, and the overall aesthetics of the building. The boxes are steel structures supported by columns that bears the vertical loads. They are consolidated by massive steel diagonals that brace the vertical surfaces. A steel structure also spans the interstitial spaces between the boxes and supports all of the lateral loads of the building. The diagonal grid façade has two main criteria that differentiate the regions of the envelope. First, the vertical regions are only self-supporting and not structural; however, the slope planes play a structural role in the building. Second, depending on the orientation of the regions, myriad glass types control the heat gain and natural lighting. The combination of both determines the specifications of the physical components of the façade in all areas of the building.

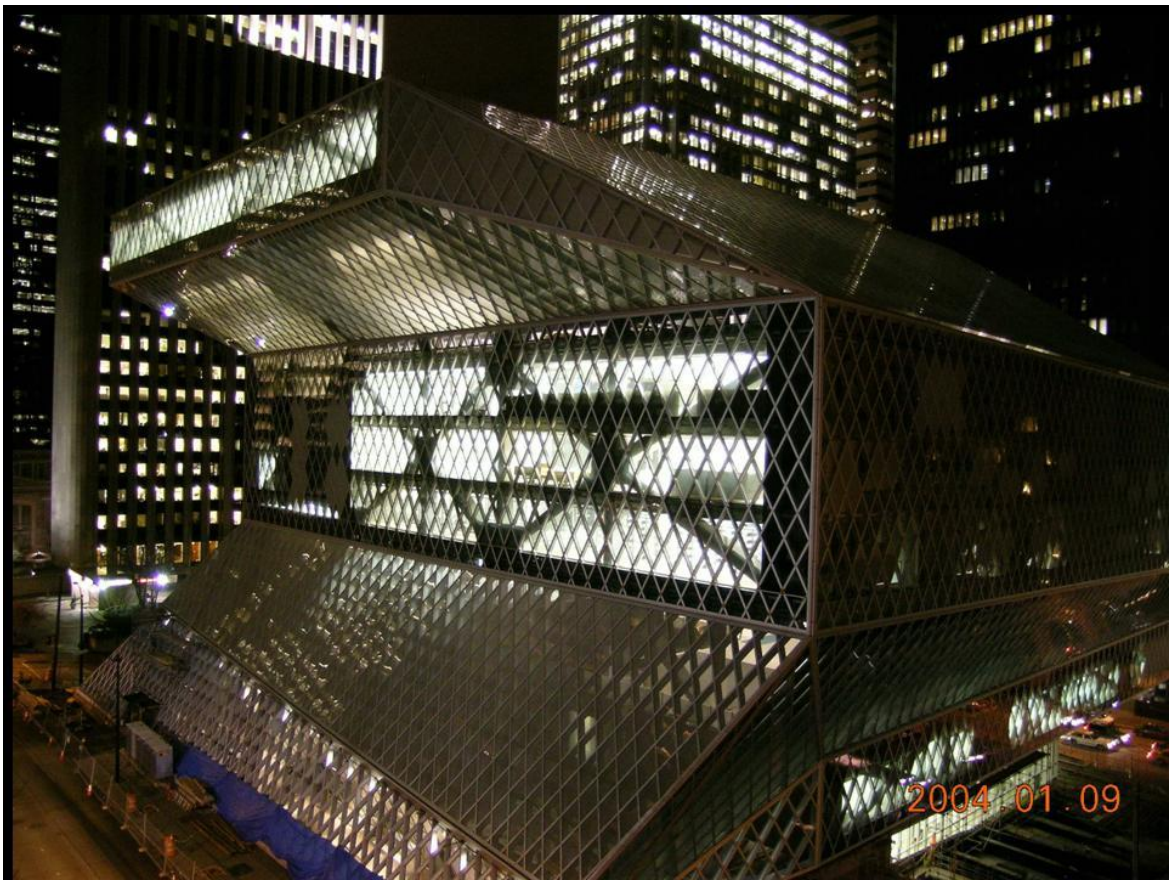


Figure 4.2. Façade regions sharing the same diagrid pattern (Courtesy of © Marc Simmons, 2015)

In addition to this main approach, several other aspects such as fabrication, installation, waterproofing, natural lighting, preventing snow accumulation, and even maintenance are integral to the design specifications. Appendix A provides detailed explanations of these aspects by the expert designer and the original description of the physical components. It also discusses the decisions that were made during the design process.

Case Study 2: Via Verde Residential Building

The design approach of the façade of the Via Verde Residential Building, designed by Grimshaw Architects in New York City, is essentially a mega prefabricated panel (Figure 4.3) lifted on site for installation. This panel also includes the *brise-soleils* and balconies installed on site before lifting.

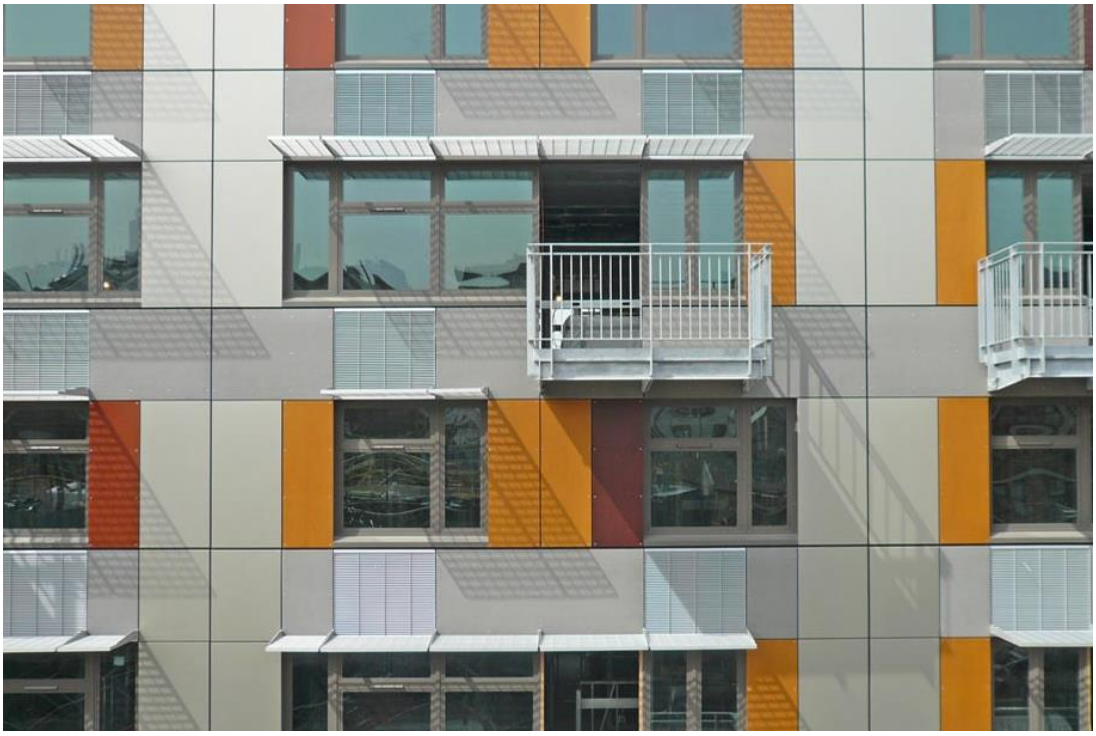


Figure 4.3. Via Verde mega panels (Courtesy of © Marc Simmons, 2015)

These mega panels are a composition of layers that address challenges derived mainly from their structural conception, installation process, thermal expansion and insulation. The layering is composed of an outer rain screen and water barrier that protect the mineral wool insulation, followed by a structure made of galvanized studs that leave a great deal of space for the electrical system and the interior finishing. Since the construction of this project, such layering of prefab panels has become a standard solution in New York City.

The structure of the mega panel is based on the Vierendeel approach. The apertures of the rectangular metal grid of the structure leave space for installing windows and balcony doors, and it provides anchor points for overhangs and balconies. Bracing is installed behind the opaque areas of the panel. The installation sequence begins with one panel hanging from the slab, and the following panel, placed on the edge with interlocking joints, is attached to the slab to continue the process.

Because of this installation sequence, the balcony door is in the middle of the stack joint, which is halfway between the slabs. Because of the size of the panel, it is subject to thermal expansion that depends on the differences between internal and external temperatures from season to season. Such dimensional variation determines the tolerances and the sealing of the joints between panels and the design of the connections to the slab, the purpose of which is to prevent shear forces.

The focus of interest of this case study is the production of variations according to manipulation of the Design Schema. Appendix B provides a detailed description of the panels.

Case Study 3: 100 at 11th Residential Building

This case study focuses on the random generation of the patterns of subdivisions of the mega façade panels (Figure 4.4) of the 100 at 11th Residential Building, which were conceptually conceived by Atelier Jean Nouvel and materialized by Front, Inc. The design is based on the specular reflection of the sun on the water, defining the aesthetics of the surfaces of the façade.

The façade is subdivided inti rectangular mega panels, similar to those of case Study 2, lifted on site for installation. The regularity of the major subdivision allows the vertical transfer of loads though the edges and control of the seals between them. The mega panels are internally randomly subdivided according to the following guidelines: The largest piece of glass should be close to the living room for a better view. Ten percent of the façade area needs to consist of operable windows in the residential areas of New York City, and they must be close to the kitchen and not at the floor level for security reasons. The location of these two fundamental glass panels having been defined, some edges are vertically extended to create internal mullions, and additional horizontal subdivisions create the remaining glass panels.

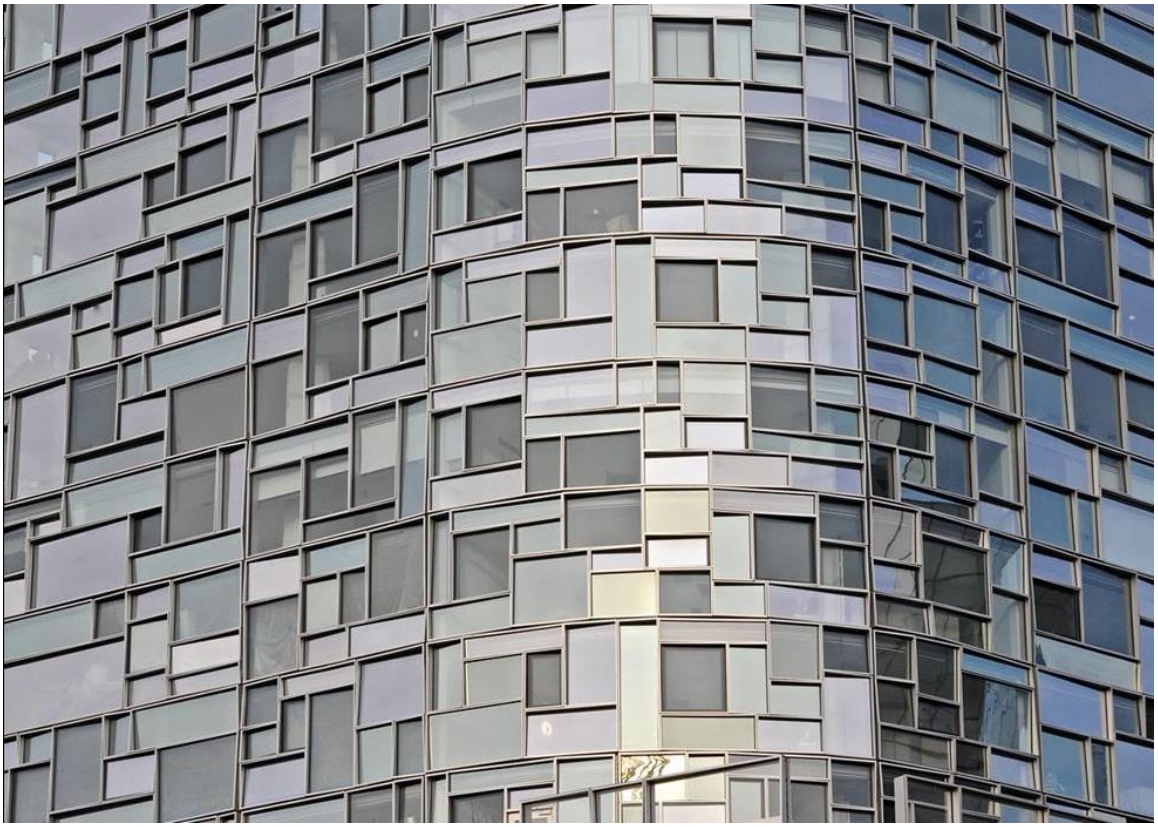


Figure 4.4. Steel mega panel with random pattern of glass distribution (Courtesy of © Marc Simmons, 2015)

Every glass panel has three main attributes. One is its size, determined by the rationalization of the mega panel subdivision, a tilted angle in four directions (up, down, left, and right) from one to five degrees, and glass type. All these details are explained in Appendix C by the expert.

The inner steel structure of the mega panel is composed of a horizontal steel tube on which the bracket joints to the slab edge are, vertical mullions, and the lower edge. Externally, everything is glass and aluminum, which negotiate the variations derived from the random geometry and orientation of the glass panels.

4.2. Distilling Methodology for Capturing Design Knowledge

The methodology for distilling design knowledge is mainly based on a verbal analysis of the transcriptions of the three case studies presented by the expert designer. The transcriptions have been complemented by digital documentation of the projects. The purpose of the analyses is to search for evidence that supports the hypothesis stated in chapter two regarding the definition of the taxonomy of primitive physical components, the declaration of the underlying patterns of organization driving the assembly of components, and the notion of a repository collecting and organizing all that knowledge as the main condition for capturing design expertise. In this regard, several analyses of qualitative data contribute to identifying the main aspects involved in the design, fabrication, and installation of façade systems; their requirements and tradeoffs; the structure of physical components, and evidence of actions executed by the designer during the processes of all three case studies.

The Verbal Analysis Method

Verbal analysis is a method of qualitative research involving classifying, sorting, and quantifying *messy data* (Chi, 1997), which is derived from complex activities that produce large amounts of non-structured verbal data. Integrating quantitative elements with qualitative studies, this method reduces subjectivity. It differs from think-aloud protocol analysis, which distills information while solving a problem rather than presenting results; however, it does not necessarily explain the complexity of what a subject is doing. Although Ericsson and Simon (1984) identified explanations, descriptions, justifications, and rationalizations in their think-aloud protocols, the protocols still differ from the explanatory type of verbalization. Furthermore, while think-aloud protocol analyses focus on the process, verbal analyses are on what the subject knows. This particular study is based on the verbal analyses of a single subject that eliminates any distinction among subjects with regard to eloquence. In this study, the same expert designer presents and explains the three case studies. Furthermore, the

expert is a professional lecturer, a skill that facilitates the communication of content. Multiple observers or repeated observations can assure the reliability of the verbal analysis (Ittelson, Rivlin, & Prositanskt, 1970, p. 644). In this particular study, a single observer iteratively searches for major categories of coding throughout the three case studies.

The definition of a protocol for the verbal analysis, based on theoretical questions regarding capturing expertise, can be approached in three different ways according to the widely used guide for verbal analysis elaborated by Chi (1997): the *interpretation* approach, which uses qualitative data to interpret quantitative data; the *complement* approach, which uses quantitative data to confirm qualitative data and vice-versa; and the *two-step* approach, which uses qualitative analysis as the background for generating hypotheses that are tested later. In the following chapter, this study principally adheres to the last approach, for it integrates distilled knowledge to explore the extent to which it can be later reused. In addition, unlike the top-down approach, which is driven by theoretical questions that seek confirmation, the bottom up approach is driven by feedback from analyses that can trigger new hypotheses generated from data.

Successive verbal analyses are driven by inquiries that structure the entire process in a sequence of definitions of the research problems and the formulation of specific questions and analyses. The procedure is structured in several steps (Figure 4.5) that can be iteratively readjusted and thus refine the questions based on feedback about the analyses.

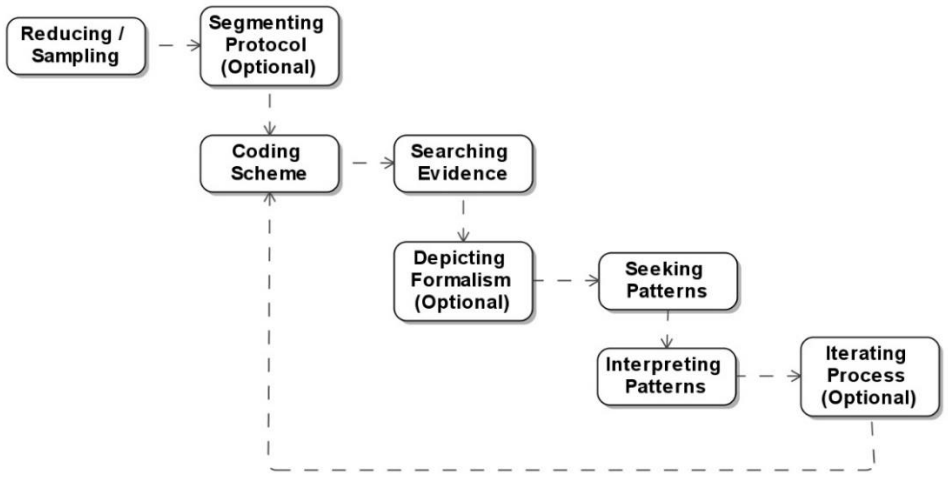


Figure 4.5. Steps for verbal analysis according to Chi (1997)

Searching Versus Segmenting Approach

After a reduction of the extension of transcriptions by skipping the non-content text, the first step is to identify the unit of analysis. Multiple grain sizes can be defined. Considering the characteristics of the data, grain size could be a key word along with sentence, idea, reasoning chain, or episode. Grain size and inference, however, pose a tradeoff. We can have macro or micro inferences depending on the unit of analysis. However, avoiding segmentation it is also acceptable in searches for the occurrence of specific targets. According to the aim of this study, the searching approach has been adopted, and the process entails going through a sequence of sentences describing and explaining four categories of design knowledge: the structure of physical components and related abstract entities, the actions of designers, the requirements, and the aspects or perspectives involved in the definition of the design. Analyses of the transcription of this study are based on the semantics of sentences (Purcell, Gero, Edwards, & McNeill, 1996, p. 226), and searches for utterances and specializations of these four major categories in the transcriptions. Discrete utterances confirm general categories, and specializations imply sub-categories. Narratives of the transcriptions also provide the context for establishing cross relationships among the categories and analyzing their correlations.

General Coding Schemes for Searching for Evidence

The coding formalism of this study corresponds to the need to search, identify, and distill the taxonomy of primitive types of a domain, which enables building a design knowledge repository for further reutilization. The schemes are organized into several categories (Table 4.1): Structural knowledge, which searches for descriptions of objects corresponding to physical components; R*equirements* (or Problem Knowledge), which determine what a physical structure must achieve or avoid; and *Behavioral Knowledge, which* identifies episodes that exhibit evidence of all actions executed during the design process. In addition, an overall fourth category of *Design Aspects* is a collection of the spectrum of aspects that participate in the definition of a design from multiple perspectives. The risk of having too many categories obscures the relationships across the content by excessively atomizing the data. To prevent such atomization, the four major categories have a maximum of one level of generalization of recurrent observations (Ittelson et al., 1970) and one level of specialization. The following sections will provide the context and specific definition of each final category.

Table 4.1. Overview of coding scheme categories

Coding Categories	Generalization	Definition
Structural Knowledge (Objects)	Physical components	Parts, assemblies, and related attributes
	Pattern of organization	abstract strategies that organize parts according to design intent
Problem Knowledge (Requirements)	Target-oriented	Goals to achieve
	Failure-preventive	Risks to avoid
Behavioral Knowledge (Actions)	Situation	Interpreting the context of the problem
	Problem	Formulating the design problem
	Patterns	Defining the strategy of organization
	Solutions	Generating design solutions
	Domain	Reusing knowledge
Design Aspects (Perspectives)	Structural	Issues directly affecting the physical structure of the design
	Performance	Quantification of the satisfaction of functional requirements
	Procedural	Procedures or processes that affect the design decisions

4.3. Searching for Design Knowledge

The formulation of specific coding schemes for the proposed major category are an attempt to build a clearer understanding of the internal structure and relationships across various types of knowledge. *Structural Knowledge* has two main categories: *Physical Component, which* contains sub-categories describing the physical composition of a design; and *Patterns of Organization, which* contains sub-categories of *Requirements* are classified into two main general categories based on a definition of "requirements" by Becker (2008) that states that requirements are target-oriented, indicating that they delineate a specific goal that a design must achieve,; and that they are failure-preventive, which indicates that a design should avoid critical conditions. Behavioral knowledge contains five sub-categories consisting of actions that designers execute related to *Design Situations, Problems, Patterns of Organization, Solutions,* and *Domains* of Knowledge. Structural and behavioral knowledge as well as the key characteristics of Requirements are understood as inputs for the problem formulation, which have been already been discussed in chapter two.

Design Aspect knowledge, the final category, emerges from the multiple restrictions and perspectives driving the design decisions. *Design Aspects* are organized into three general sub-categories: *Structural*, *Performance*, and *Procedural*. Their identification operates in a direction opposite to that of the previous categories because many aspects emerge along the coding process.

Searching for Structural Knowledge

The fundamental questions driving this coding scheme are related to the identification of the primitive types that constitute the physical structure of a design, and the provision of coherence to the arrangement of *parts (*Figure 4.6*).* Transcriptions of explanations from the case studies reveal at least three specializations of physical components (Table 4.2): the final indivisible *Parts,* such as the single mullions; the *Sub-assembly* of parts with a specific role or function, such as the internal steel structure of a

prefab façade panel; and the overall *Assembly* of components of the design, such as the resulting curtain wall. The literature reviewed in Chapter Two, however, identified three mechanisms or abstract entities that contribute to a coherent design of parts: *Chunk of Constraints,* which, according to Gobet et al. (2001), represent already known restrictions among parts; *Conceptual Structures* (Lawson & Dorst, 2009), which represent abstract spatial entities; and *Design Schema* or Schemata, which, in the words of Lawson (2004), represent the overall pattern driving the design organization.

Examples of the above specializations of the three case studies can be found in Table 4.3. Explanatory sentences contain the description of the entities, their attributes, and cross relationships that will be further integrated in a comprehensive model.

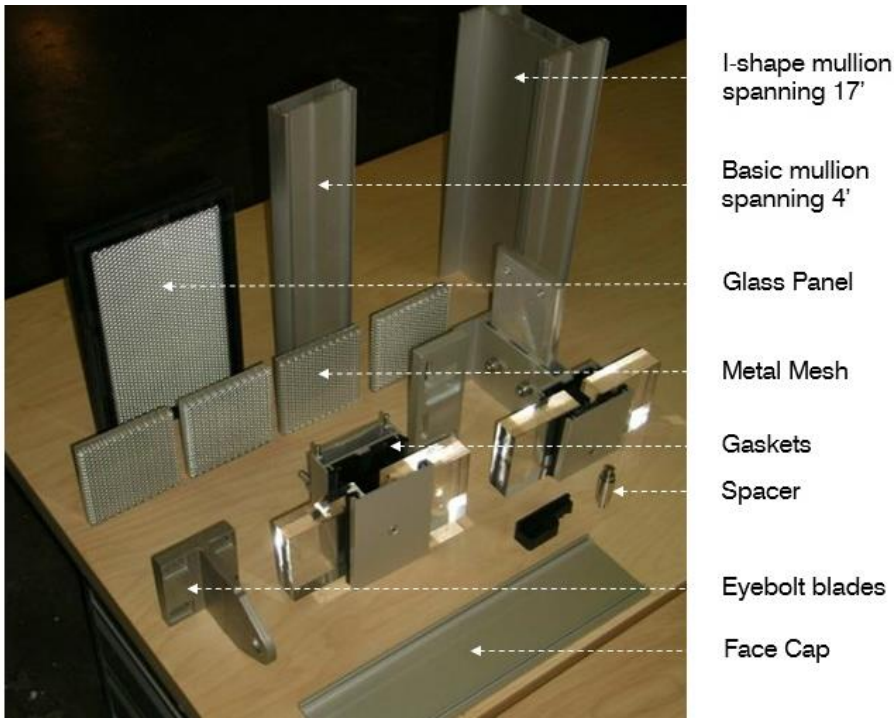


Figure 4.6. Case study one: curtain wall systems of parts (Courtesy of © Marc Simmons, 2015)

Table 4.2. Structural knowledge coding scheme

Generalization	Specialization	Definition
Physical Components	Assembly	Overall assembly of the product or project
	Sub-assembly	Component with specific sub-function made of parts
	Part	Single part
Patterns of organization	Design Schema	Overall pattern of organization of the design, also called “design parti”
	Conceptual Structure	Abstract architectural elements that locally organize the design such as corridors or hallways
	Constrains	Implicit or explicit limits that define the domain of values for given attributes

Searching for Behavioral Knowledge or Design Actions

The hypothesis of this study is that the designer’s actions or the behavioral component of the design activity triggers, in some way, the reutilization of the structural knowledge. Explanations for the three case studies allow us to retrospectively reconstruct the reasoning behind the decision-making process while designing. The coding scheme (Table 4.4) is directly derived from Chapter Two, in which we have searched for episodes with evidence of the already documented design actions related to interpreting the *Situation*, building *Problems,* recalling *Patterns of organization,* generating *Solutions* and modeling a *Domain* of knowledge. The purpose is to visualize and quantify their impact on the definition of a design. Examples of such episodes can be found in Table 4.5.

Table 4.3. Samples of structural knowledge from the case studies

Generalization	Specialization	Case	Transcription Samples
Physical Components	Assemblies	C1	Diagonal structure only exist in the interstitial zones.
		C2	Everything is a single mega panel, and brise-soleils are bolted in, and the balconies are integrated at the site.
		C3	The mega-panel dimensions vary from 11' x 18', x 20', x 37' and affect the dimensions and number of component sub-panels.
	Sub-assemblies	C1	(The aluminum mesh) is encapsulated within the glass.
		C2	The entire structure of the panel is a vierendeel beam with two hanging portions to the right and left leaving space in the center for the door.
		C3	Everything outboard of that (the reference plane) is going to be gaskets, aluminum, glass, water proofing, everything inboard of that, steel (structure).
	Parts	C1	We wanted certain meshes to be very tight and other ones to be quite open. We were, actually, modulating it through micro folding
		C2	…that panel was made on galvanized plate cold form sheet metal.
		C3	And then you have the vertical steel mullions, the glass, and exterior metal trim.
Patterns of organization	Design Schema	C1	The idea that the exterior diagonal grid then comes into the picture as a diagonal shear grid that would serve as a lateral system to the building.
		C2	In a mega panel, the door is half the way in one panel and half the way in the other panel. The door was actually installed after the fact. That was a kind of macro decision that had to be made.
		C3	So, the whole building becomes rationalized into this mega grid, floor by floor and seven panels per floor plate. After that, everything has some variability in a kind of a crazy grid inside the panels.
	Conceptual Structure	C1	Diagonal structure only exists in the interstitial zones.
		C2	So, you start to see the joint perimeter. These joints are slightly larger. They are larger because they need to handle thermal expansion.
		C3	So, there is a continuous line across the entire façade, which is a reference plane.
	Chunk of Constrains	C1	Because it has a relatively lower thermal mass, it has high thermal expansion, but this material is so white it is also rejecting a huge amount of heat. So it is actually not expanding very much.
		C2	These joints are slightly larger. They are larger because they need to handle thermal expansion…. What panels do, because they are so large, they do expand.
		C3	…for the room you satisfy the requirement of allocating the window from inside where you most appreciate the view.

Table 4.4. Behavioral knowledge coding scheme

Generalization	Specialization	Definition
Design Situation	Reformulating SBF	Changing the physical structure, its function or the way it executes such function
	Forming Analogies	applying knowledge from different fields based on commonalities with the current design situation
	Looking for Emergence	Identifying affordances that may be useful later in the design process
Design Problem	Framing	Selecting specific aspects of the problem to focus on
	Building Ill-define Problems	Building problems with under-determined aspects that require interpretation
	Co-Evolving Problem-Solution	Exploring the problem and solutions in parallel
Pattern of Organization	Recalling Chunk of Constraints	Knowing general constraints of the problem in advance
	Recalling Conceptual Structures	Using well-known architectural spatial entities to organize parts
	Recalling Design Schemas	Using an already known “design parti” to organize the design
Design Solution	Following Parallel Lines of Thought	Exploring parallel alternative solutions
	Evaluating Preliminary Solutions	Using heuristics and rough estimations to assess the viability of solution candidates
	Integrating Knowledge	Visualizing the trade-space among different domains of knowledge
Design Domain	Recognizing Problems	Referencing the new problem on typical and recurrent problems
	Reusing Physical Parts	Reutilization of components based on professional experience
	Applying Design Rules	Using procedural knowledge derived or distilled from heuristics, norms, and guidelines
	Applying Evaluation Methods	Reutilization of formalized and validated evaluation methods derived form best practices

Table 4.5. Samples of behavioral knowledge from the case studies

Generalization	Specialization	Case	Transcription Samples
Design Situation	Reformulating SBF	C1	The idea that the exterior diagonal grid then comes into the picture as a diagonal shear grid that would serve as lateral system to the building, while there were still column grid system in the building, was actually a very late development.
	Forming Analogies	C3	The collage is based on the specular reflection of the sun on the water… The question was, can we imagine a façade which has such reflection?
	Looking for Emergence	C2	One thing is also very good is there is nothing between studs. Electricians… have impunity; there is no worries about cutting. They don't have to wait for the carpenters to come in to finish the water proofing.
Design Problem	Framing	C3	So, the whole building becomes rationalized into this mega grid, floor by floor and seven panels per floor plate. After that, everything has some variability in a kind of a crazy grid inside the panels.
	Building Ill-define Problems	C2	If you add all these tolerances together is unreasonable, basically there is a subjective artful judgment…If you are too conservative, people will say that you are not serious. If my consultant is saying that I have to add 2" to all the joints everywhere, it doesn't help.
	Co-Evolving Problem-Solution	C2	The fact that the door runs across the stack joint, which is half of the room, was actually a tradeoff. We had to say that as a technical detail we had to figure out in the macro picture of the project.
Pattern of Organization	Recalling Chunk of Constraints	C2	So, you start to see the joint perimeter. These joints are slightly larger. They are larger because they need to handle thermal expansion…. What panels do, because they are so large, they do expand.
	Recalling Conceptual Structures	C1	The idea that the exterior diagonal grid then comes into the picture as a diagonal shear grid that would serve as lateral system to the building, while there were still column grid system in the building.
	Recalling Design Schemas	C3	Their first design intent was a collage-like organization of panels, but more importantly, every piece of glass was intended to be different from the adjacent panels.
Design Solution	Following Parallel Lines of Thought	C1	…during schematic design, the entire skin of this building was a tension system. For about six weeks it was an obsessively done tension system... It took a little while to work up that scheme and get the pre-stress on the cable to support fabric over the 134' spans, and you still have snow loads in Seattle.
	Evaluating Preliminary Solutions	C3	If you are going to do nonlinear load paths with aluminum box mullions, you need to reinforce them with sheet metal aluminum; (it) requires connections detailed as moment connections with large fasteners and exposed bolts…. It is, actually, very difficult.
	Integrating Knowledge	C1	Industrial stretched metal, we visited the factory… we asked them if could they adjust the rate of the holes. So once you punch it, the degree of pull governs the degree of aperture in the mesh.

Table 4.5 (continued)

Design Domain	Recognizing Problems	C2	In thermal expansion it is more critical than curtain walls, because you have up to 15' horizontal mega panels.
	Reusing Physical Parts	C1	This (the mullion) could have been a box, it could have been a "T." Obviously, an "I" shape was chosen because it is conceptually similar to the steel.
	Applying Design Rules	C3	The face dimension of every piece of steel on the inside is going to be limited and harmonized to 3," meaning that now we can play with the depth. And the depth was 3," 4," 5," and 6." Every piece of steel except for the tube on top is either 3 by 3, 3 by 4, 3 by 5, or 3 by 6. "The bottom member is 6."
	Applying Evaluation Methods	C2	Several options are evaluated (simulated) with different combinations of material thicknesses to satisfy the requirements

Searching for Requirements

The requirements are the primitive elements of a design problem that can also be defined as a coherent arrangement of tradeoffs among requirements. This list of target-oriented or failure-preventive features can also be understood from the perspective of how prescriptive they are. By extending the description of Dorst (2007) regarding the qualitative differences among aspects of the design problem also discussed in Chapter Two, both general types of requirements can be classified into the same three main sub-categories : determined, which cannot be avoided; under-determined, which require interpretation; and un-determined, which leave room for the preferences of the designer who defines the criteria for addressing them (Table 4.6). The challenge is inferencing the requirements from the transcriptions and distilling a clearer definition of them. While some requirements are clearly expressed in quantitative terms, others are implicit in the description of parts. Examples of the descriptions for inferring the requirements can be found in Table 4.7.

Table 4.6. Problem requirements coding scheme

Generalization	Specialization	Definition
Target-oriented	Determined	Prescriptive and unavoidable requirements of the problem
	Under-determined	Requirements that need interpretation by the designer
Failure-preventive	Un-determined	Criteria defined by the designer's preferences

Emerging Design Aspects or Perspectives

The aspects category attempts to capture the perspective from which the design is defined. Aspects represent either what the views of several stakeholders are (Jobe, Johnson, & Paredis, 2008) or how the design performs a given function or task (Augenbroe & Park, 2005). The hypothesis is that the definition of the design is constantly changing the perspective. The proposed categories are Physical aspects, which directly affect the configuration, Performance aspects, which refers to how an artifact addresses a given task or function, and Procedural aspects, which include the influence of the processes along the life cycle of a building façade in this particular domain (Table 4.8). Examples of specialization of these categories from the case studies can be found in Table 4.9.

The Physical Aspects of the façade design directly impact the physical definition of the design, such as structure, material, or geometry. For example, besides multiple levels of subdivisions that drive the propagation of physical components such as mullions, frames, or glass panels, the geometrical aspect also determines the allocation of the points for load transfer between the façade and the main structure of the building where all joints and brackets are placed to address wind, snow, dead, or seismic loads that depend on the geographic location, the soil, or the type of building structure. In addition, multiple virtual reference planes define the end of the building structure, the edge of the slabs, the allocation of the axial lines of the self-supporting structure of the facade, the positioning of the insulation and water barriers, and the finishing plane and tolerances between contiguous components. All of these auxiliary geometrical elements can determine the range of adaptability of physical systems to diverse conditions.

The Performance Aspects in design not only quantify the execution of a function but also allow one to track the set of attributes and components that are integrated in such tasks. The most common performance aspect of building façades are related to energy, lighting, and acoustic insulation. For example, the first two entail an important

Table 4.7. Samples of requirements or problem knowledge from the case studies

Generalization	Specialization	Case	Transcription of Requirements
Target-oriented	Determined	C1	Every single hole in the extrusion (mullion) has to absolutely match with the drilled hole on this extrusion (cap)
		C2	We put the water requirement to 12pounds/sqft pressure of infiltration, not 50 pounds.
		C3	There is another rule that says that the first 60' from the bottom of the building needs to maintain the street wall on the sidewalk.
	Under-determined	C1	They (the upper curtain wall brackets) have to handle the vertical translation…
		C2	The fact that the door runs across the stack joint, which is half of the room, was actually a tradeoff. We had to say that as a technical detail we had to figure out in the macro picture of the project
		C3	…the intention of every single piece of glass was different from the intention of the adjacent panels.
	Un-determined	C1	They (OMA and LMN Architects) just wanted to be that, no columns, all clear span interiors.
		C2	…
		C3	The question was, can we imagine a façade that has such reflection (based on the specular reflection of the sun on the water…)?
Failure-preventive	Determined	C1	Every penetration of the cap must be water proofed.
		C2	We don't care about what the deflections are during peak loads of a category five hurricane. All we care about is its structural integrity, staying on the building and doesn't fall.
		C3	If it will have an operable window, you don't want it at floor level, especially in a high rise.
	Under-determined	C1	If you look at some other buildings that are a little bit cheaper, you often see those kinds of lines, which don't have perfect lines of reflectivity.
		C2	So we specify the joints so that the panels never fail structurally and don't induce in-plane load so strong.
		C3	It will reduce slightly the floor plate, and stretching the building right tangent to the site will result in a thinner building that is not really deep enough to accommodate the depth of the units.
	Un-determined	C1	…
		C2	The city is trying to mitigate plumbs impacts.
		C3	Originally, they didn't want a regular grid on the façade….

tradeoff. While natural lighting is a valuable feature, it could also could provoke the greenhouse effect and increase glare in interior spaces. These paradoxes are important inputs to the definition and the selection of material specifications. Although a number of analytical tools support the analyses of these aspects, expert designers typically apply simple methods based on rough estimations during preliminary design stages. Based on their experience, they have the ability to evaluate the designs early on in the design process, when they calculate many of the heuristics regarding linear dimensions, gross areas, volume of materials, or average number of elements per item. These simple calculations also help them to define preliminary quantifications and costs. At later stages in the process, designers refine the design with physical mockups, an important source of information, along with the anticipation of fabrication and installation issues and visual evaluation in which they test the performance aspects of the design, such as waterproofing.

Procedural Aspects correspond to all of the processes that take place during the fabrication, installation, and operation of the building. Their limitations determine the context and the boundaries of what can feasibly be designed. The modulation and the sizing of the components is a combination of the original size of the technical or cost limitations of fabrication, the tracking capacity, and the weight and on-site maneuverability. These are interrelated processes that contribute several inputs and constraint to the design. For example, some installation processes in buildings with complex geometry start from the corners to prevent conflicts at the end of the sequence. Besides modularity, the range of predefined tolerances is another important aspect since assembling components from different suppliers needs to be anticipated. Simple consideration to factors such as replacement of components or easy access to cleaning and maintenance can also contribute to defining the size of façade components or the development of complementary systems for maintenance.

Table 4.8. Design aspects coding scheme

Generalization	Specialization	Definition
Structural	Aesthetic	Aesthetical considerations based on preferences
	Geometry	Geometrical issues represented in auxiliary geometry
	Structure	Structural requirements and limitations
	Material	Material properties and limitations
	Tolerances	Tolerances for installation, thermal expansion, and manufacturing
	Code	Restrictions from the building code
Performance	Energy	Energy consumption related issues
	Lighting	Natural lighting issues
	Acoustic	Accosting considerations
	Water proofing	Water infiltration prevention
	Fire protection	Fire protection restrictions and requirements
	Snow accumulation	Issues derived from the snow accumulation
	View	Quality of the view
	Cost	Quantification and cost estimation issues
Procedural	Fabrication	Fabrication limitations and capabilities
	Transportation	Restrictions that influence weight, sizing, and maneuverability
	Installation	Installation procedures
	Maintenance	Accessibility for cleaning and repair

Table 4.9. Samples of design aspects from the case studies

Generalization	Specialization	Case	Transcription Samples
Structural	Aesthetic	C1	The "I" shape box mullion that has been engineered to span 17' on the clad, this could have been a box, it could have been a "T." Obviously, it was an "I" shape because it is conceptually similar to the steel.
	Geometry	C2	…then your thermal stress is coming, the whole thing expands, and you get a little problem. We map though all those instances…. It comes down to geometrical analyses; you really have to draw it.
	Structure	C1	Putting the bracing structure and using it structurally for the lateral stability of the building puts tons of steel inside of the rest of the structure. Then, the façade is important, but actually it is for free.
	Material	C1	Stainless steel has one-third of the thermal conductivity of aluminum. It is better thermally, but it is not as good as plastic spacers, but the plastic spacers are subject to long-term deterioration. So, stainless steel spacers are the most reliable long-term solution.
	Tolerances	C3	The curtain wall anchors basically have two compress channels (to adjust it horizontally), four anchor bolts inside, and a dead load seat. The bolt on the bracket pulls on as full lateral and dead load restraints. The panel lays out gently and then it is adjusted.
	Code	C3	Then we need 10% of an operable window in residential areas in New York City. Then we say, we have 400 sqft space and 40sqft of operable windows.
Performance	Energy	C2	…the insulation is absolutely parametric, depending entirely on the needs.
	Lighting	C1	The fact that they (the blinds) are not there is attributable to the density of the steel.
	Acoustic	C3	…and the gaskets to provide place for thermal and acoustical seals.
	Water proofing	C1	It has three lines of defense inside of the 2" curtain wall.
	Fire protection	C3	…a horizontal metal closure for stopping fires.
	Snow accumulation	C1	There is snow accumulation to a certain point. The snow changes the U-value of the assembly because the snow is insulating.
	View	C3	And there is a door (in the bedroom) that slides back that can be open. So, you can actually see the entire panorama (from the bedroom), and the whole façade which is (in the corner) 40-45' long.
	Cost	C1	Could the design team be authorized to raise funding? If you didn't have all that certainty, somebody could say "Can we look other solution?" But because it was so well documented, the team was allowed to do that.
Procedural	Fabrication	C3	In fabrication, each mega panel would be pre-assembled and the beam connected to the panel on site."
	Transportation	C2	There are a limited number of panels that can be stuck in a truck.
	Installation	C2	What you really need is your first panel (hanging from the slab) and the next one seated into that with interlocking pins, and you lock it back in (the upper slab), and you are good to go.
	Maintenance	C1	Cylindrical tie up point on the roof (provide access to maintenance)

4.4. Depicting and Interpreting Design Knowledge

The already described coding schemes capture episodes of the three case studies that contain explanations, descriptions, assumptions, inferences, and justifications, among other forms of support for design decisions (Table 4.3). While some of them, such as the description of physical parts, are very explicit, others contain more encrypted implicit information that requires interpretation. Depictions and further analyses of such pieces of information partially reveal how the designs are structured, the type of design actions involved in the definition of such a structure, the rotation of perspectives while the structure of the design is defined, and distributions of requirements within the structure of the design.

The depictions of coded transcriptions are node-based mapping that attempt to rebuild the hierarchical structures of designs, and two- dimensional matrices that evaluate the cross-references and the influence of designer actions, aspects, and requirements that determine the structure. The node-like mapping of the three case studies captures not only the structure of *Parts, Sub-assemblies,* and the overall *Assembly* of *Physical Components* but also the nodes or groups of nodes that can be understood as *Conceptual Structures* or *Design Schemas*. In addition, this node-based representation is the context within which one maps the relationship of *Constraints* and *Requirements* to particular nodes or abstract immaterial patterns. The second type of depictions, two-dimensional matrices, quantifies cross references between the design structure and other categories to visualize, examine, and measure the influence of these other forms of design knowledge that conform to the definition of the design.

The Design Structure

The aim is to build a representation of the *Structural Knowledge* of each case study that identifies the participating objects and to visualize the relationship between *Physical Components* and *Patterns of Organization* with the purpose of building a comprehensive model of the domain. While the material elements are represented as

nodes, the immaterial elements are dashed lines. The components are stratified in three levels—single *Parts, Sub-assemblies,* and overall *Assembly,* representing the hierarchical decomposition of the structure. Patterns are linked to the corresponding nodes or groups of nodes, where they influence. Originally, the representation was conceived as a tree structure. However, because of reutilization by different sub-assemblies of the same primitive parts, the resulting representation is a graph-like structure of nodes viewed from the perspective of the overall assembly. Although the segments of the transcription explaining the parts and assemblies include references to their attributes, such as material specifications or dimensions, these attributes are not represented in the graphs, but they will be included in the implementation of the meta-model that will merge the three case studies into one comprehensive model that will be discussed in the following chapter.

The first case study, the *Seattle Library* (Figure 4.7), is structured in at least four levels: The upper level is the overall façade, also called the *Wrapping Diagonal Grid,* from the perspective of the *Design Schema*, which splits into two major conceptual structures in the next levels: *Vertical Surfaces* and the *Interstitial Zones between* them, which are reference points for most of the decisions and parts specifications. These two abstract entities include two sub-assemblies, one playing a structural role and the other the corresponding curtain-wall for each case (Figure 4.8). The third level is the sub-assembly of the glass panel shared by the two curtain walls, similar to many other parts allocated at the bottom level. Also, in the third level, sub-assemblies split the diagonal structure into *steel frames* modules because of transportation and installation restrictions.

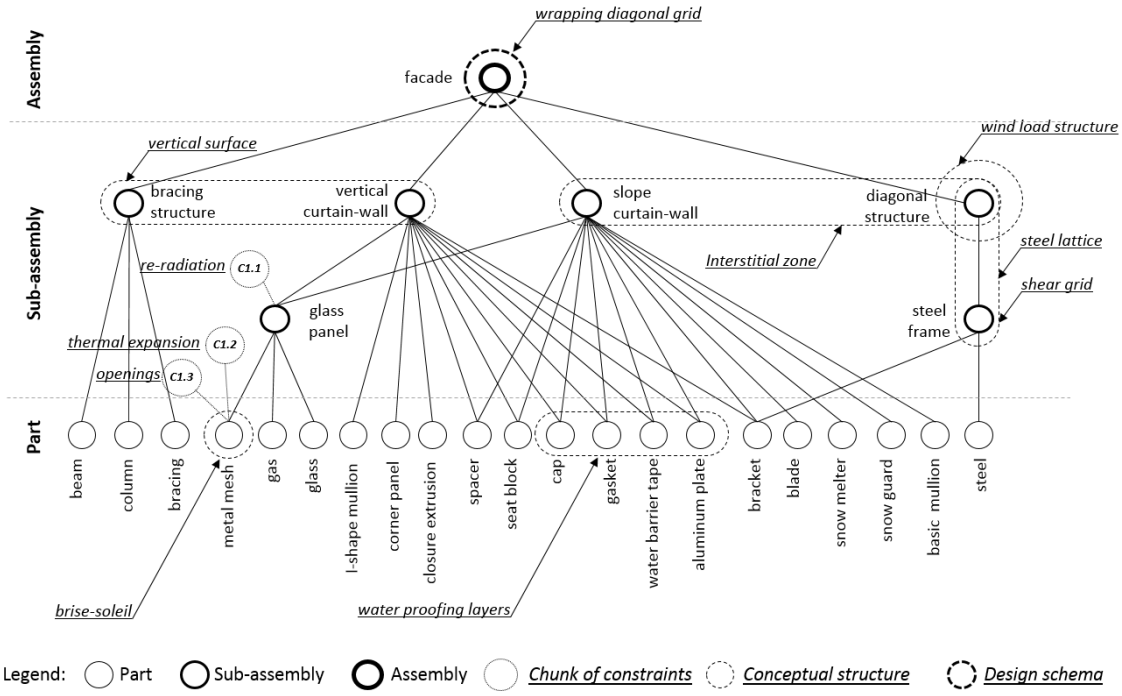


Figure 4.7. Physical structure of case study 1, Seattle Library

Figure 4.8. Case study one, sub-assemblies of the diagrid façade (Courtesy of © Marc Simmons, 2015)

At the bottom level, we can also find two key conceptual structures: the *Brise-soleil,* which directly matches a primitive metal mesh corresponding node, and the *Water Proofing Layers*, which, unlike the *Brise-soleil*, is a subset of parts of the curtain walls. The remarkable feature of the building is the flexibility of the notion of the conceptual structure since it relies on a single part, a subset of parts, or an assembly of them. In addition, depending on the perspective, the same sub-assembly can embody myriad conceptual structures such as the denominations of the *Wind Load Structure,* the *Steel Lattice*, or the *Shear Grid* of the same diagonal steel structure supporting the curtain wall in the *interstitial zones*.

The declared *Chunk of Constraints* seems to be focused on the knowledge included in the definition of the glass panel and the specifications of its individual parts (Table 4.10). They are a collection of already known characteristics that contribute to the performance of the sub-assembly of the glass panel, which is constructed of several layers. No other recall is made beyond the framing of the problem in terms of solar radiation and natural lighting control.

Table 4.10. Chunk of constraints of case study 1, Seattle Library

Constraints	Transcriptions
C1.1 Re-radiation	The low-E coating in the interior plus the gas basically mitigate the radiation, and it pushes the radiation coming-in back out again.
C1.2 Thermal expansion	Because it has a relatively lower thermal mass, it has high thermal expansion, but this material is so white it is also rejecting a huge amount of heat. So it is actually not expanding very much. What we are getting is a 35% cut from the benchmark immediately….
C1.3 Openings	but given the geometry of the mesh curving, when it moves slightly, it becomes completely opaque, and when it moves lower in the sky, of course, it becomes more open.

The second case study, the ***Via Verde*** *Residential Building,* it is also organized in four levels (Figure 4.9). The top of the hierarchy contains the overall assembly of the pre-fab panel, also referred to as the *mega-panel* by the architect from the *Design Schema* perspective. On the second level, this panel has a frame structure that is decomposed into third level sub-assemblies that are finally decomposed into metal parts at the bottom, similar to the stack joint that is decomposed in a series of aluminum profiles, gaskets, and other single parts. However, the pre-fab panel also includes several sub-assemblies on the second level with no further decomposition (Figure 4.10).

Although balconies, *brise-soleils*, windows, and other prefabricated sub-assemblies are included none of them reveal explicit sub-parts describing their internal structure, as the right end of Figure 4.9 shows. This lack of information is probably the result of knowing that the internal structure is not required for its integration in the overall assembly of the prefab panel. In other words, it seems that only information required for establishing relationships with other entities is explicitly declared. An example of such information is the rain screen joint that connects the screen with the structural frame of the pre-fab panel made of studs.

The *Conceptual Structure* of the *mega-panel* is linked with particular key nodes that execute particular tasks: The *join perimeter* refers to the node that groups all the parts that make up the connection and sealing between panels; the *panel edge* is the very last aluminum extrusion in the border of the panel that also belongs to the stack joint system; and the *hanger rod* is a specific vertical metal piece that takes the dead loads of a subsection of the frame metal structure.

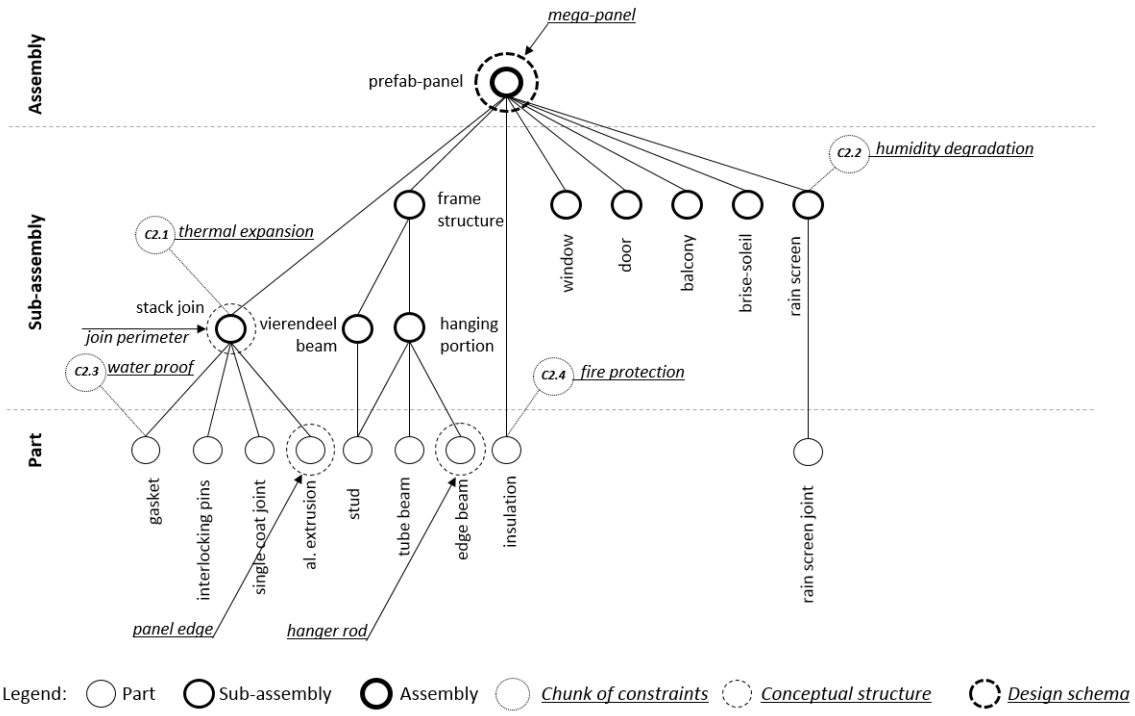


Figure 4.9. Physical structure of case study 2, Via Verde

Figure 4.10. Case study 2, sub-assemblies of the mega panel (Courtesy of © Marc Simmons, 2015)

Constraints, all of which refer to addressing typical recognizable challenges (Table 4.11), are associated with nodes with very well-defined roles such as addressing *thermal expansion* derived from the size of the panel assigned to the stack joint sub-assembly, preventing *humidity degradation* of the insulation assigned to the rain screen *Sub-assembly*; assuring *waterproofing,* also affected by the dimensional variations assigned to the gaskets; and providing *fire protection*, required by code, that is assigned to the specifications of the insulation.

Table 4.11. Chunk of constraints case study 2, Via Verde

Constraints	Transcription
C2.1 Thermal expansion	These joints are slightly larger. They are larger because they need to handle thermal expansion…. what panels do, because they are so large, they expand.
C2.2 Humidity degradation	The only issue with mineral wool is that if it (gets wet), its U-value decreases. And the idea is you have a rain screen on the front of it, so you don't have a problem with humidity degradation.
C2.3 Water proofing	For most buildings in New York, we could say 12 pounds sqft pressure under water pressure. (We can also say 15 pounds,) but those are big storms. Then, could (the compressible gasket) satisfy 8 to 10 pounds of pressure differential? It could.
C2.4 Fire protection	Then, if you put the insulation outside, it must be class zero insulation, which means mineral wool, because if you put polystyrene foam outside of the building, it just burns. You can't do it.

The third case study, the ***100 & 10th Avenue Residential Building*****,** has a *collage Design Schema* that governs the entire decision after subdividing the façade into mega panels (Figure 4.11). The panels are decomposed into a steel structure that supports the aluminum glass frames. Both *Sub-assemblies* together are conceptually interpreted by the designer as *metal mesh.* Two other *Sub-assemblies*, operable windows and glass panels, complete the system, which is decomposed into a large list of single parts (Figure 4.12). Many of these parts are similar to those in the other two case studies, at least with regard to their general primitive types, such as brackets, gaskets, or mullions. Also similar to that of the previous cases, the decomposition of the sub-assemblies only describes relevant parts that characterize the component but omits its complementary parts.

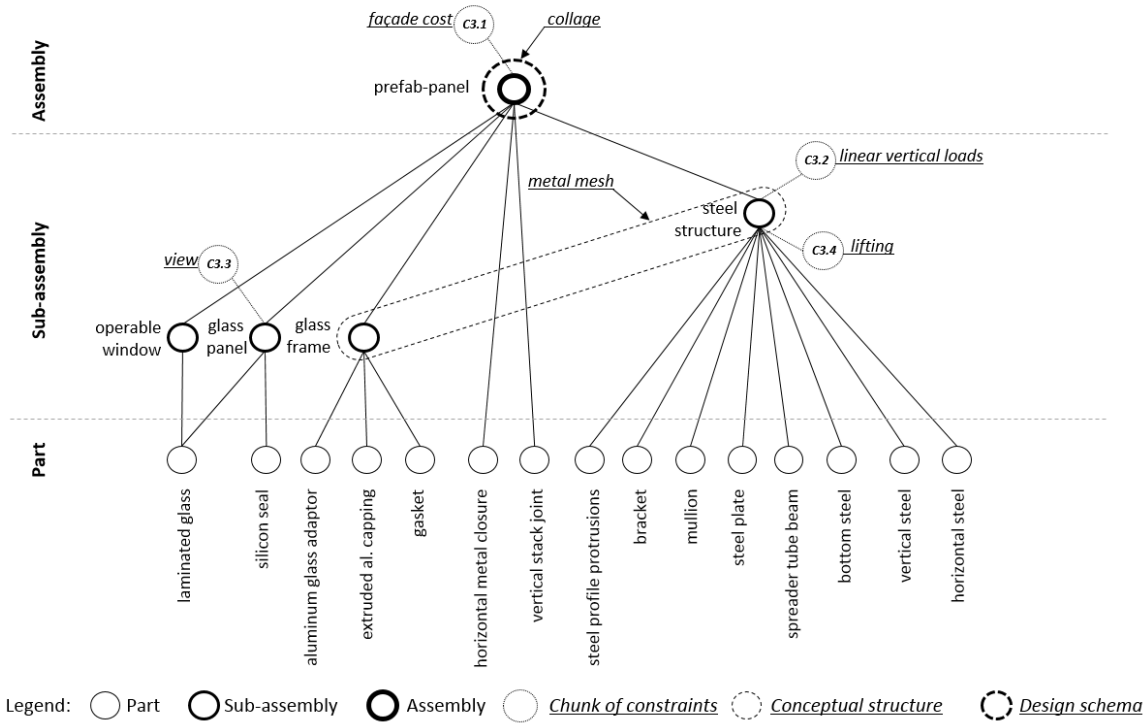


Figure 4.11. Physical structure of case study 3, 100 & 10th Avenue

Figure 4.12. Case study 3, sub-assemblies of the prefab panel (Courtesy of © Marc Simmons, 2015)

The *Constraints* are linked to nodes that are supposed to address the related functional requirements (Table 4.12). The overall panel acknowledges cost limitations;

the steel structure assumes loads derived from the lifting procedure on-site and the vertical dead loads along the entire façade; and by varying the positioning and the size of the glass panel, the designer addressed the view requirements. The constraints are declared as conditional statements by the designer, who seems to anticipate the implications of design decisions within a much broader scope and also knows the base line from which to establish comparisons and preliminary estimations.

Table 4.12. Chunk of constraints case study, 100 & 10th Avenue

Constraints	Transcription
C3.1 Cost	Because the façade represents 40% of the surface of the building, it became 25% (of the total cost). The typical cost is around 12% - 15%.
C3.2 Linear vertical loads	If you are going to do nonlinear load paths with aluminum box mullions, you need to reinforce them
C3.3 View	…you satisfy the requirement of allocating a window from the inside, where you most appreciate the view.
C3.4 Lifting	To hang precast, you hang it in two points. You don't want three points because you can't guarantee that the three points will carry the loads properly.

Structure Driven by Design Actions

While shaping the physical structure of a design, designers execute several actions throughout the process. Mapping and quantifying the cross references of these actions with the definition of the design structure allows us to better understand the impact and the role they play in facilitating the reusability of design knowledge. The cross references between designer's actions and the physical structure are presented in tabular form for each case study and then aggregated as a whole (Table 4.13). The designer's actions are listed in columns and classified in those related to interpreting the design situation, followed by formulating problems, recalling patterns, generating solutions, and building a design domain. The tables register events when the two coding schemes coincide during the same episode. The following statement exemplifies this situation from the third case study:

Table 4.13. Incidence of design actions over the design structure

Structure vs. Actions		Situation			Problem			Pattern			Solution				Domain					
		reformulating SFB	forming analogies	looking for emergence	framing	building ill-defined problem	co-evolving problem-solution	recalling chunk of constriants	recalling design schema	recalling conceptual structure	following parallel lines of thought	evaluating preliminary solution	integrating knowledge	rocognizing problems	reusing physical parts	applying design rules	applying evaluation methods	num of actions	total references
Case Study 1, Seattle Library																			
Patterns	design schema	1							1									2	2
Patterns	conceptual structure					1		1										2	2
Patterns	constraints							2				2						2	4
Components	assembly			1				3								1		3	5
Components	sub-assembly							3									1	2	4
Components	part		1					11					2		1	4		5	19
	num of structures	1	1	1		1		5		1		1	1		1	2	1		
	total references	1	1	1		1		20		1		2	2		1	5	1		
Case Study 2, Via Verde																			
Patterns	design schema				1													1	1
Patterns	conceptual structure							2				1						2	3
Patterns	constraints							5	1									2	6
Components	assembly	1			1					1	1	1	1			1		7	7
Components	sub-assembly			1				3	1				1		2	1		6	9
Components	part	1			1			5			1		1		1		1	7	11
	num of structures	2		1	3			4	2	1	2	2	3		2	2	1		
	total references	2		1	3			15	2	1	2	2	3		3	2	1		
Case Study 3, 100 & 10th Avenue																			
Patterns	design schema	1			2	2			1							4	1	6	11
Patterns	conceptual structure		1	4	1		1											4	7
Patterns	constraints	1						1				2			1	2		5	7
Components	assembly			1				1								2		3	4
Components	sub-assembly	1			1		2			1		1		1	1			7	8
Components	part		1		1	1		2	1			2		1	3	6	1	10	19
	num of structures	3	2	2	4	2	2	3	2	1		3		2	3	4	2		
	total references	3	2	5	5	3	3	4	2	1		5		2	5	14	2		
Aggregated Case Studies																			
Patterns	design schema	2			3	2			1	1						4	1	7	14
Patterns	conceptual structure		1	4	1	1	1	3				1						7	12
Patterns	constraints	1						8	1			4			1	2		6	17
Components	assembly	1		2	1			4		1	1	1	1			4		9	16
Components	sub-assembly	1		1	1		2	6	1	1		1	1	1	3	1	1	13	21
Components	part	1	2		2	1		18	1		1	2	3	1	5	1	2	13	40
	num of structures	5	2	3	5	3	2	5	4	3	2	5	3	2	3	5	3		
	total references	6	3	7	8	4	3	39	4	3	2	9	5	2	9	12	4		

"The collage is based on the specular reflection of the sun on the water… The question was, can we imagine a façade which has such a reflection?"

It is a clear explanation of the general *Design Schema* (structural knowledge), based on an inference derived from an analogy (behavioral knowledge). Such intersections of structural and behavioral knowledge are registered. Light gray codes represent a low number of references, and dark gray the opposite. In addition, the total number of participating structural components and cross references are counted.

Each case studies emphasizes the influence of design actions in unique ways. The two-dimensional matrices map the occurrence of the multiple cross references between the actions and the structure. The analyses focus on the most salient cross-reference that show a tendency or a singularity.

The ***Seattle Library*** case study from the perspective of the patterns of organization shows that the *reformulating SBF* influences the selection of the overall *Design Schema*:

"*The idea that the exterior diagonal grid then comes into the picture as a diagonal shear grid that would serve as a lateral system to the building."*

This episode shows evidence of changing the function of the façade from a regular curtain wall to the structure of the building. With regard to the *Conceptual Structures*, they are only invoked when a high-level reference is needed. For example, the *open boxes* or the *steel lattice* are called for *formulating the ill-defined problem* of having open spaces free of structure and the dimensioning of pre-assembled sections *,* respectively. *Recalling chunk of constraints* also has a direct impact on the final applied constraints of the design.

From the perspective of the *Physical Components*, the results reveal the tendency of the final assembly and sub-assemblies to cross reference with *recalling*

chunk of constraints as well, with three incidences each. The assembly is also affected by *looking for emergence* and *applying design rules.* It shows that right after the designer visualizes an opportunity to use the façade for the lateral stability of the building, many constraints and rules rationalize such an idea. Results also show a concentration of actions affecting the specification of single parts with five participating actions and nineteen total references. Most recall an already known *chunk of constrains* followed by *applying design rules.* Nevertheless, large blank areas are in the *framing* and *recognizing* problems in both categories, *Patterns* and *Components*.

The explanation of this first case study seems to emphasize the development of the design rather than the formulation of the general problem. It provides evidence for the role of the already known *chunk of constraints* and *design rules* in the specifications of their *Parts* and *Assemblies* by reusing knowledge about how they work, what their properties and their imitations are, and how they go together. An example of an episode explaining the singularity of the steel structure follows:

> *"…under dead and wind loads, we needed double steel depth in these areas to 24" … basically we used that I-shape to add double depth steel to the back of the existing grid."*

The designer already knows that the constraints derived from the span exceed the capacity of the current steel section and then immediately creates and applies a rule by doubling the structure where it is required. This episode shows that chunking mechanisms encapsulate knowledge about not only the parts but also their assembly similar the what .Gobet et al. (2001) and Dabbeeru and Mukerjee (2008) point out.

Regarding the patterns of organization for the ***Via Verde*** case study, *framing* the problem determines the conditions for the overall *Design Schema*. The following statement clearly defines the conditions for the design and on-site assembly once the *parti* is imposed:

"Everything is a single mega panel, and brise-soleils are bolted in, and the balconies are integrated at the site."

Along with the development of the *Schema*, several other restrictions are applied to conceptual structures by *recalling chunk of constraints.* The two following statements are examples:

"So, you start to see the joint perimeter. These joints are slightly larger. They are larger because they need to handle thermal expansion."

"The only issue with mineral wool is that if it gets wet, its U-value decreases. And the idea is you have a rain screen on the front of it, so you don't have the problem of humidity degradation."

The *Conceptual Structures* joint perimeter and rain screen are receptors of the recalled constraints regarding thermal expansion and humidity degradation, respectively. In both cases, the conceptual structure is used to generalize the area of influence of the recalled constraints. Furthermore, several recalled constraints are literally integrated to the actual constraints of the design.

With regard to the *Physical Components* category of the coding, case study 2 shows, in the lower section of Table 4.13, a homogeneous distribution of actions affecting them. Several actions intervene in the *Assembly*. The following episodes are a sequence of major interventions in the overall assembly

"…brise-soleils are bolted in, and the balconies are integrated at the site" (framing)

"If the panel stops before the slab, you can wreck the panel" (evaluating preliminary solutions, reformulating SBF)

"Several options are evaluated with different combinations of material thicknesses to satisfy the requirements" (following parallel lines of thought, integrating knowledge)

"…it is more logical to say that it will be a systemic tolerance and everything will be uniformly less 1/8" rather than randomly distributed" (applying design rules)

Sub-assemblies and *Parts* are also affected by several actions. The quantification of the influence on the design structure by the main actions of designers indicate again the tendency of relying on an already known chunk of constraints to define these sub-assemblies and parts limiting the design space early on. This tendency, similar to that in case study 1, significantly impacts *recalling chunk constraints,* which determines many of the features based on previous experience. Most of these recalls are executed under the logic of evaluative statements (e.g., *if you have…, when you have…. if you want…*), followed by actual recall (e.g., *they have a detail which is…., the insulation is….,* or *the pressure is….*), and finally the adapting of knowledge (e.g., *you are going to weld*…, *the insulation depends on*…, *or could the compressible gasket satisfy*…?). These steps form a sequence of recall: condition, association, and adaptation.

Unlike the previous two case studies, the third case study, ***100 & 10th Avenue***, does not show concentration in the recalling chuck of constraints column. On the contrary, the upper section of its table seems to emphasize cross-referencing *Design Schema* with *applying design rules* and *Conceptual Structures with looking for emergence*. Less intense are the cross references of constraints with *evaluating preliminary solutions* and *applying design rules*. The collage *Design Schema* is driven by several rules that produce its apparent randomness. Rules that control the positioning of the glass panels, determine the distribution of the vertical mullions, and select the glass type and the tilted angles shaping the collage. Although these rules define the actual geometry of several single parts, they are defined at a high level of the schema (See Appendix C, the Mega-panel section). *Sub-assembly, instead of being* driven by actions, appears to be interpreted by them, since the action *looking for emergence* identifies and qualifies the resulting features of *Sub-assemblies*. For example, in the next episodes, the conceptual description is preceded or followed by a resulting feature:

"…obliquely it collapses into the metal. You see only this insane mesh of metal at certain angles."

"…you got this glass-protected wind screen that is sealing your apartment and terrace."

"The landscape designers started to specify those trees boxes in this tri-dimensional lattice."

The last element of the pattern category, *constraints*, shows some influence of evaluating preliminary solutions, represented by the following episodes that are then followed by actual evaluations.

"If you are going to do nonlinear load paths with aluminum box mullions, you need to reinforce them.

"Because the façade represents 40% of the surface of the building, it became 25% of the total cost. The typical cost is around 12% - 15%."

Regarding the cross references between the *Physical Components* and actions, the general *Assembly* is driven also by the *applying design rules* because of the similarity to the S*chema*, which refer to the same *Assembly* from an aesthetical perspective. Although several actions are involved, the most important ones from the perspective of the evolution of the design are *reformulating SBF* and co-*evolving the problem-solution* at the beginning of the design process through key questions that lead to different solutions:

"Can the mullions actually all be tapered, tilted, and offset from each other and structurally be one? No."

"On top of that, the nonlinear load paths floor to floor added an extra complexity since it then needed to go around the frames…"

Framing also affects the definition of the strategy for proceeding with the design process. After all, it is not a solution, but a determination of the guidelines of the design task:

"Everything outboard of that (the reference plane) is going to be gaskets, aluminum, glass, water proofing; everything inboard of that is steel."

The definition of *Parts* is mostly driven by *applying design rules* that control many attributes of the primitive components according to the collage *Schema*. To maintain consistency, parts also recall several *chunk of constraints* and *reuse physical parts.* Some examples of the rules follow:

"They said that we want our largest piece of glass close to the living room."

"Then we need 10% of operable window in residential areas in New York City."

"If it will have an operable window, you don't want it at the floor level, especially in a high rise."

In summary, the aggregated case study shows evidence of the strong influence of recalling fragments of already known constraints on design decisions. It also provides clues about the sequence of reusing such knowledge by identifying the condition, building the association, and executing the adaptation. Such mechanisms operate on the abstract *Design Schema* level and on the lower level of the attributes of a single part. In addition, the various sequences of design actions, not necessarily in a particular order, take place during the evolution of the design.

Switching Perspectives While Defining the Structure

This analysis attempts to visualize and quantify the impact of design aspects on the definition of the structure to provide a more comprehensive understanding of the factors that enter into design decisions. For such a purpose, the aspects identified by the coding scheme are compared to the coding scheme of Structural Knowledge using a two-dimensional matrix that keeps track of the cross references in the same episode of its two main categories (Table 4.14). While the left column represents the *Pattern of Organization* and *Physical Components* coding schemes, the top row represents the *Physical, Performance, and Procedural aspects* schemes. The table quantifies the cross-referencing episodes for every case study and an aggregation of all of them. The darker the squares, the higher the number of cross references.

Table 4.14. Impact of design aspects over the structure

Structural vs. Aspects		Physical						Performance								Procedural				num of aspects	total references
		aesthetic	geometry	structure	material	tolerances	code	energy	lighting	acoustic	water proofing	fire protection	snow accumulation	view	cost	fabrication	transportation	installation	maintenance		
Case Study 1, Seattle Library																					
Patterns	design schema			1																1	1
	conceptual structure			3								1								2	4
	constraints							3												1	3
Components	assembly		1	3								1								3	5
	sub-assembly		1	3				1	1							1		1		6	8
	part	3	1	7	1	1					1		1			3		9	2	10	29
	num of objects	1	3	5	1	1		2	1		1	2	1			2		2	1		
	total references	3	3	17	1	1		4	1		1	2	1			4		10	2		
Case Study 2, Via Verde																					
Patterns	design schema																	1		1	1
	conceptual structure				1	1		1			1							2		5	6
	constraints					3		1			2	1								4	7
Components	assembly					3	1	1										3		4	8
	sub-assembly	1		2				1			1				1	1		2		7	9
	part			1	1		1	4			3	3				3		3		8	19
	num of objects	1		2	2	3	2	5			4	2			1	2		5			
	total references	1		3	2	7	2	8			7	4			1	4		11			
Case Study 3, 100 & 10th Avenue																					
Patterns	design schema	3	4											1						3	8
	conceptual structure	2	1											1	1					4	5
	constraints		1	1										1	1			1		5	5
Components	assembly		2											1		1	1			4	5
	sub-assembly				2						1					3		1		4	7
	part	5	3		1	1	1	1		1	1	1			2	2		2		12	21
	num of objects	3	5	1	2	1	1	1		1	2	1		4	3	3	1	3			
	total references	10	11	1	3	1	1	1		1	2	1		4	4	6	1	4			
Aggregated Case Studies																					
Patterns	design schema	2	4	1										1				1		5	9
	conceptual structure	3	1	3	1	1		1			1	1		1	1			2		11	16
	constraints		1	1		3		4			2	1		1	1			1		9	15
Components	assembly		3	3		3	1	1				1		1		1	1	3		10	18
	sub-assembly	1	1	5	2			2	1		2				1	5		4		10	24
	part	8	4	8	3	2	2	5		1	5	4	1		2	8		14	2	15	69
	num of objects	4	6	6	3	4	2	5	1	1	4	4	1	4	4	3	1	6	1		
	total references	14	14	21	6	9	3	13	1	1	10	7	1	4	5	14	1	25	2		

The first case study, the ***Seattle Library***, in the subsection *Patterns of Organization,* shows some influence of the *Physical* aspects on the *Design Schema,* specifically, the *structure* aspect. This aspect determines the challenge of the diagonal wrapping grid, which maintains the lateral stability of the building:

"…(the diagonal wrapping grid) would serve as a lateral system to the building."

Conceptual Structures, also cross-referenced with the *structure* aspect as well, mainly define the role of groups or systems of parts in the entire system. It seems to be a mechanism that assigns tasks as we can verify in the next two episodes:

"Vertical surfaces are conventional column and bean structures."

"Diagonal structures only exist in the interstitial zones."

The *Performance* aspects, specifically *fire-protection,* affects the *conceptual diagonal structure*. Since the steel must be fire protected if it is the primary structure, designers add concrete columns to the building that assume such a main role, while the diagonal structure assumes a complementary role as a lateral system.

In terms of *Constraints,* the three registered episodes are also from *Performance* aspects, specifically, the *energy* aspects, which determine the conditions for the glass panel with the embedded metal mesh, as in the following example:

"Because it (metal mesh) has a relatively lower thermal mass, it has high thermal expansion, but this material is so white it is also rejecting a huge amount of heat. So it is actually not expanding very much."

The *Assembly* and *Sub-assemblies* also have cross-references from the perspective of the *Structure*. Those references define the challenges that these two entities need to address, and the impact their specifications For example:

"We got three diamonds long among supporting brace steel. The reason is this surface is not contributing to the lateral stability of the building. Certain surfaces are laterally stable and when they are not, we take as much steel as we need."

In the *Physical Components* section of the table, the *Parts* have multiple cross references. Many features of *Parts* are defined by the *installation* aspect, which belongs to *Procedural* aspects. Its purpose is to speed up the assembly sequence and ensure precision and adjustments. The following episodes show considerations that come from professional experience—when the designer visualizes steps ahead in the installation process and anticipates them by adding features that enable the *Parts* to address them, as the following episodes reveal. While the first episode focuses on the sequence of installation, the second focuses on the precision of the installation. In the third example, the designer recognizes the complexity of the process and the need for flexibility in adjustments. See the following examples from the transcription:

"The gaskets are actually going on to the extrusions."

"A piece of hard plastic is perfectly indexed to the distance between the aluminum and the glass."

"Here is the bracket. It is basically a block of aluminum. It has linear slider holes plus minus ¾" adjustments..."

The *fabrication* aspect, from the *Procedural* category*,* also defines restrictions derived from the capabilities of the process and material. However, it appear to have less influence on the design than *installation*. The following example demonstrates the direct relationship between the process of folding and the size of the opening of the metal mesh, which determines how much light enters:

"We wanted certain meshes (within the glass panel) to be very tight and other ones to be quite open. We were, actually, modulating it through micro folding it."

With regard to the *Physical* aspects, specifically, the *structure* aspect, it implicitly contributes to framing the requirements of the *Parts* in the entire system, defining some degree of specification, and usually providing a supporting argument. The next examples illustrate how considerations from the *structure* aspects contribute to defining the parts according to a type of reasoning chain created by these three elements, which

define what the part is doing, and how it works. The first episode provides an argument for a requirement, and the second describes the role of the part and a related requirement. Unlike the first two with implicit content, the third explains the entire reasoning chain: the role, the requirement, and the argument.

> *But because we don't have diagonal steel behind our vertical curtain wall, we need to span floor to floor*
>
> *"(mullions) are spanning on the diagonal. Diagonal span from floor to floor is 17', which is quite long."*
>
> *"These brackets only provide perpendicular wind load resistance, but they are laterally flexible and vertically flexible. … It is a rigid connection that just moves up and down with the thermal expansion of the curtain wall."*

The *aesthetic* aspect, the same *Physical* aspect category, influences non-restrictive decisions that are subject to interpretation, thus, they are more difficult to rationalize even though they seem to be obvious despite the lack of quantitative arguments. See the following example:

> *"Obviously, an "I" shape was chosen (for the mullion) because it is conceptually similar to the steel."*

The second case study, the ***Via Verde***, the main focus of the mega-panel Schema is how to frame the problem from the perspective of the *installation* aspect, which belongs to the *Procedural* category. It is also determined by *fabrication* preconceptions even though it exhibits no clear cross referencing with the second aspect.

With regard to *Conceptual Structures,* they are cross-referenced at least once as a sort of mechanism that assigns tasks by coupling, for example, the edge of the panels with *material* or the joint perimeter with *tolerance,* both of which belong to the *Physical* category. Regarding the *Performance* aspects category, the rain screen is coupled with *energy* and *waterproofing*. Lastly, the balconies and other elements are coupled with

installation from the *Procedural* aspects. These associations immediately build a problem and distribute the tasks that will be precisely assigned to assemblies or parts. The next example episode illustrates a high-level association in which the *material* aspect and *Conceptual Structure* are explicit and the tolerance aspect is *implicit*:

"The aluminum extrusions are defining the true edge of the panel…"

The *Constraint* row indicates that they are mainly under the influence of the *tolerance* and *waterproofing* aspects from the *Physical* and *Performance* categories, respectively. The oversized panel is subject to dimensional variation because of thermal expansion and requires the *tolerance* aspect in the design. Such a phenomenon imposes a series of constraints on the edges of the panel and the connection to the slab of the building. The *waterproofing* aspect intervenes in the entire performance of the panel since insulation is very sensitive to humidity and affects the entire layering design approach of the panel. The following episodes illustrate each:

"These joints are slightly larger. They are larger because they need to handle thermal expansion…"

"The only issue with mineral wool is that if it gets wet, its U-value decreases."

The lower subsection of the Table 4.14 of this second case study reveals the influence of aspects over the Physical Components. The cross references of the *Assembly* and S*ub-assemblies* tend to determine actions that one must take to address general challenges rather than specifications. At the *Assembly* level, the Pre-fab panel is coupled with compensating for *tolerances,* evaluating insulation options according to regulations and energy codes*, or* pre-assembling for on-site *installation*, combining *Physical*, *Performance* and *Procedural* aspects. At the *Sub-assemblies* level, the insulation type is coupled with *cost* estimation from the *Performance* category; and the panel frame made of studs with welding as a *fabrication* method and electric system *installation,* both *Procedural*. While parts seem to address the resolution of the challenges, assemblies seem to match the strategy. For example,

"Everything is a single mega panel, and brise-soleils are bolted in, and the balconies are integrated at the site."

For the same case study 2, *Parts* are highly determined by the entire scope of the *Physical*, *Performance* and *Procedural* aspects that impact the actual specifications of the attributes of parts. Cross referencing establishes a direct link between attributes and aspects without necessarily specifically declaring the requirements. Matches such as the R-values of insulation and *energy*, gasket type, and *waterproofing*, insulation type, and *fire protection*, material type and *fabrication*, and interlocking system and *installation*. For example:

"From the energy standpoint, you get a building 60% opaque, and 3" gives you R13, 4" 1/2 R per inch. If you want R16, just add one inch to the insulation."

The third case study, the ***100 & 10th Avenue***, exhibits a concentration of *aesthetic* and *geometric* aspects influencing the *Patterns of Organization*. The *aesthetic* aspect attempts to generate a collage-like *schema* driven by an analogy while the *geometrical* aspect focuses on the rationalization of the subdivision of the mega-panels to achieve such an effect on the façade. The next episodes link these two aspects with the schema in a sort of encryption of the general design approach:

The collage is based on the specular reflection of the sun on the water… The question was, can we imagine a façade which has such reflection?

Their first design intent was a collage-like organization of panels, but more importantly, the intention of every single piece of glass was different from the intention of the adjacent panels.

At the *Conceptual Structure* level, cross referencing with the aspects shows a tendency of defining strategic approaches to developing the design by coupling conceptual abstract structures with strategies to address the challenges derived from the aspects: the aesthetic aspect with the concept of mesh of metal; g*eometry* with a reference plane to differentiate a structure from glass panels; *view* with adding terraces

to the lower floors, *cost* with the metal-glass façade estimation. The next episode is an example of using the conceptual structure of reference plane to define a design strategy:

> *"…you need to add some rational management to this problem. So, there is a continuous line across the entire façade, which is a reference plane."*

At the *Constraints* level, constraints define conditions such as the location of windows because of the *view,* which also determines the *geometric* pseudo-random internal subdivision, the facade modulation according to the *structure*, or lifting restrictions because of on-site *installation.* The following episode explains a key condition of the design:

> *"… you satisfy the requirement of allocating the window from inside, where you most appreciate the view."*

If we examine the section of *Physical Components* of the last case study, the overall *Assembly* shows some intensification of the influence of the *geometric* aspect. It is consistent with the influence of the same aspect over the *Design Schema* in terms of dimensioning of the collage. The remaining aspects also determine high-level challenges stemming from the decision of having a mega-panel, such as allowing a panoramic *view,* pre-assembling off-site to facilitate *installation*, and the recognition of trucking limitations for *transportation.*

The *Sub-assemblies* in cross-referencing, similar to those in the Via Verde case study, demonstrate associations with strategies for addressing general challenges such as using auxiliary *geometry* as a reference plane for steel structures, focusing the *waterproofing* between glass panels, defining the *fabrication* method of the pieces that interface the steel structure with the glass panels, or defining the technique for connecting the steel structure of the mega-panels to the slab during *installation,* as can be seen in the following episode:

"That is the edge beam that has a series of horizontal brackets that attach to the top of the edge beam of the slab."

The *Parts* section seems to be the more intense area of influence of the *Physical*, *Performance* and *Procedural* aspects of the façade domain, with 12 aspects invoked and 21 total cross references between parts and aspects. The *aesthetic* aspect shows a larger number of cross-references, followed by *geometry*, *cost, fabrication, installation,* and the others. Thus, this case study indicates that these references have a tendency to impact the values of the attributes and features of the parts by coupling with some kind of specification. For example, the *aesthetic* of the collage is coupled with the specification of the glass type, *geometry* with the tilted angles of the glass, *cost* with glass panel types*, fabrication* with the welding technique, *installation* with the lifting points, *material* with the differentiation between the structure of the panel and the glass frames, *tolerance* with adjustment details, *code* with operable window size, *energy* and *acoustic* with gasket type for sealing, *waterproofing* with the specification of the stack joint, and *fire protection* with the closure element between floors. The following episode demonstrates consistence between the specifications of the geometry of the glass while pursuing the collage *Design Schema*:

"Atelier Jean Nouvel provided a breakdown of the façade system as a composition of glass panels with four directions of rotation: tilting up, down, left and right; four glass variations; and angles of rotation varying through 0,2,3,4, and 5 degrees of vertical."

In summary, the three case studies show several tendencies. Case study 1 shows a tendency to associate a high-level approach to the diagonal wrapping grid *Schema,* which in some way defines the role of the *Conceptual Structure,* and *Constraints define* the conditions. All of these strategic decisions consistently affect the specification of the lower level *Parts*. Case Studies 2 and 3 demonstrate similar tendencies to associate the *Schema* with the main frame of the problem by splitting the façade into mega-panels. The conceptuality also shows a link with general strategies,

and like those in case study 1, *Constraints* also define conditions. Their Physical Components demonstrate a tendency to highlight key challenges from various aspects related to the overall assembly, strategies embedded in the sub-assemblies, and specification of the parts.

The final section of Table 4.14 in the beginning of this section shows the aggregated impact of the design aspects of the three case studies, intended as the perspective from which decisions-making occurs. Despite focusing on the areas of interest of each case study, the expert designer highlighted several other aspects that exhibited tendencies. The table shows a tendency of higher impacts in the specification of the Part according to the contexts defined in the Pattern of Organization category, and the Assembly and Sub-assembly from the Physical Components. Installation, aesthetic, structure, and fabrication together with lower intensity energy and waterproofing affect the features and attributes of the Parts, which constantly changes the perspective of the specifications.

Requirement Association

After executing the searching procedure based on the already described coding scheme for requirements, the distilled requirements were attached to the Physical Component or Pattern of Organization for every case study they are related to. These associations are represented as rounded squares linked with the corresponding nodes on the graph of the design structure. The association requirement node facilitates the visualization of the distribution of the Target-oriented (TO) and Failure-preventive (FP) requirements and identifies their impact and the level of determination they are imposing over the structure. In addition, the distilled requirements of all of the case studies are organized and classified into tables.

From a top-down perspective, the first case study, the ***Seattle Library,*** shows a tendency to link the most general but also *determined TO* requirements such as waterproofing, fire protection, and solar heat gains with either the overall *Assembly* or key *Sub-assemblies* from the *Physical Component* perspective. We verify that only one requirement—the precision of the drilling—is linked to a lower level part, shown in Figure 4.13, which is complementary to Table 4.15. With regard to *under-determined*, we can also determine that two-thirds of the *TO* requirements are distributed at the lower level of *Parts*, and the other third is linked to either *Sub-assemblies* or the more abstract *Conceptual Structures* from the perspective of *Patterns.* For example, the requirements define tasks as a structural role, but they do not specify how to address such tasks or their desired levels of satisfaction. The only *un-determined* requirement, achieving clear span interiors, is linked to the overall *Design Schema* of the wrapping diagonal grid.

The *FP* section of this case study is considerably smaller, but it more precisely determines the two key *Parts* that should assume the requirements to avoid water penetration under very specific conditions, the glass breakage from floor to floor and also defines the under-determined conditions to avoid an irregular finish of the façade.

Even though the most undetermined requirements are linked to the general *Design Schema*, this logic of distribution seems to allocate more determined requirements or goals to the higher levels of the hierarchy of the design structure and leave more room for interpretation at the lower levels. In addition, not all of the nodes have an association with the requirements along the transcriptions. An explanation for this incompleteness may be that the expert explicitly states what is relevant to a more complete understanding of the design but fails to mention or simply implies the relevance of other requirements.

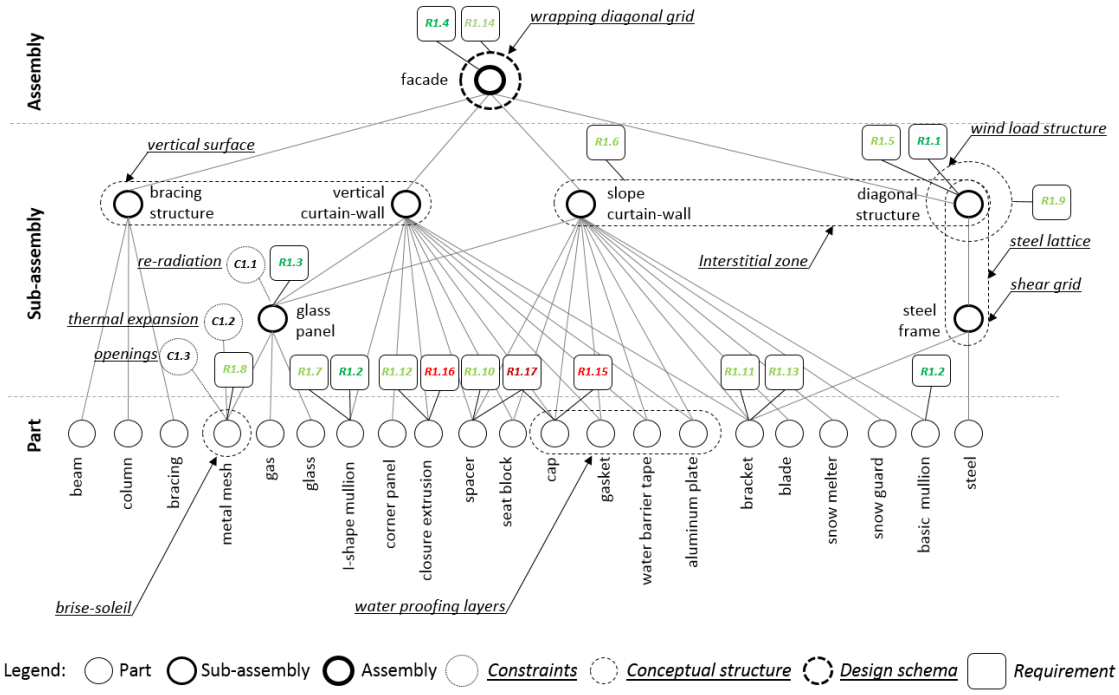


Figure 4.13. Map of requirements of case study 1, Seattle Library

Table 4.15. Requirement list of case study 1, Seattle Library

Generalized	Specialized		Requirement episode
Target-oriented	Determined	R1.1	… structure must be fire-protected in the connections with the slabs
		R1.2	Every single hole in the extrusion (mullion) has to absolutely match…
		R1.3	0.17 was targeted to the areas with larger solar heat gain
		R1.4	theoretical requirement to test 100% (waterproofing) of the building
	Under-determined	R1.5	[diagonal grid] will have a primary structural function
		R1.6	the skin elements between the open boxes to be the structure
		R1.7	[I-Shape mullions] need to span floor to floor
		R1.8	certain meshes to be very tight and other ones to be quite open
		R1.9	secondary steel structure assumable for wind load
		R1.10	… (index) the distance between the aluminum and the glass.
		R1.11	connection that just moves up and down with thermal expansion
		R1.12	provide floor closure and a partition
		R1.13	allow them (steel-bracing) to be adjusted to each other
	Un-determined	R1.14	…all clear span interiors
Failure-preventive	Determined	R1.15	Every penetration of the cap must be waterproofed
		R1.16	we don't actually break the glass (for partition)
	Under-determined	R1.17	… see those kinds of lines which don't have perfect lines of reflectivity.

The second case study, the ***Via Verde,*** exhibits a better balance between *TO* and *FP* requirements (Figure 4.14 & Table 4.16). The *determined TO* are evenly distributed across the design structure. General installation, waterproofing, and insulation requirements are linked to the overall *Assembly* of the prefab panel. The stack joint *Sub-assembly* and the more *conceptual* version of the joint perimeter assume more specific water infiltration and construction requirements. At the bottom, very specific responsibilities with explicit targets are assigned to some *Parts* according to the requirements in association with the top node of the *Assembly* as a sort of specification of the same aspect. For example, while the *Assembly* must achieve at least R13 in terms of thermal resistance, the actual *TO* requirements related to installation aspects are distributed. One is attached to the joint perimeter *Conceptual Structure* node and other to the door *Sub-assembly,* both key nodes in the installation.

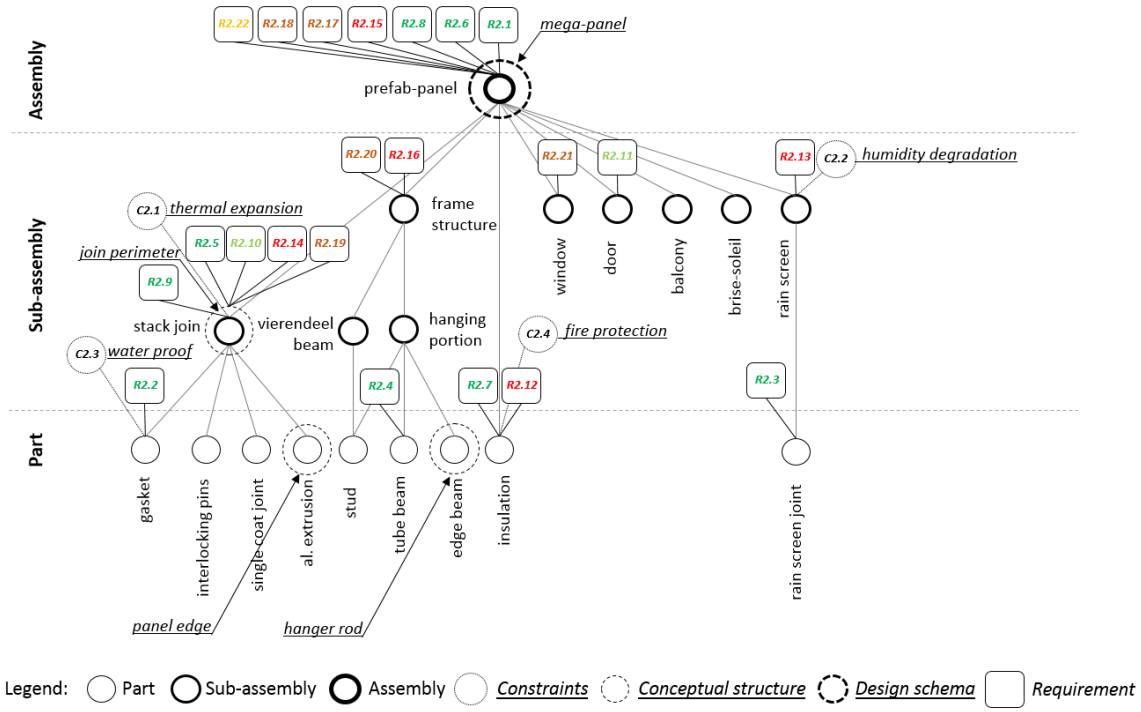


Figure 4.14. Map of requirements of case study 2, Via Verde

The *FP* section of the table places the *determined* requirements related to risk from thermal expansion at the *Assembly* of the prefab-panel. Requirements preventing fire propagation, humidity degradation, water infiltration, and structural integrity are linked to intermedia *Sub-assemblies* or *Conceptual Structure* in the case of water infiltration related to the joint perimeter node. *Under-determined* relate to tolerances linked to the general *Assembly,* while those related to cost of the structure and dust accumulation are linked to the frame and window *Sub-assembly* intermedia nodes since they represent more specific needs, which is similar to avoiding the structural failures of the joint perimeter. Finally, the *undetermined* requirement, mitigate plumbing impacts on insulation, is associated with the conception of the overall *Assembly*.

This case study shows a heterogeneous distribution of the requirements of both types *TO* and *FP* and presents a combination of highly determined requirements of the top node, combined with those that require interpretation. In this regard, the prefab-panel Assembly node is linked to three determined *TOs*, one determined *FP*, two under-determined *TOs,* and one undetermined *FP*.

Table 4.16. Requirements of case study 2, Via Verde

Generalization	Specialization		Requirement
Target-oriented	Determined	R2.1	first panel (hanging from the slab) and the next one seated into that with interlocking pins
		R2.2	could (the gasket) satisfy 8 -10 pounds of pressure differential
		R2.3	the stand out is engineered to handle the moment generated by the weight of the rain screen relative to the 5-6" distance
		R2.4	We are designing to, say, ¾" deflection over the mullion high
		R2.5	We put the water requirement to 12 ps/sqft pressure of infiltration.
		R2.6	…most buildings are designed for 50 years (water- proofing)
		R2.7	Now the 38.5°F (dew point) … is in the middle of the insulation, where we want it to be.
		R2.8	Code is R 13 for a wall. This building has R24 for walls.
		R2.9	We had to accept this configuration because the contractor insisted in putting the stack joint close to the metal frame.
	Under-determined	R2.10	… they (the joints) need to handle thermal expansion….
		R2.11	We had to say that (the door across the stack joint) is a technical detail we had to figure out in the macro picture of the project.
Failure -preventive	Determined	R2.12	…if you put polystyrene foam outside of the building, it just burns. You can't do it.
		R2.13	you have a rain screen on the front of it (insulation), so you don't have the problem of humidity. degradation
		R2.14	… in New York, we could say 12 pounds sqft pressure under water pressure.
		R2.15	The real risk is when you have thermal expansion at the point where a panel actually touches the other panel….
		R2.16	All we care is about its structural integrity, staying on the building and doesn't fall
	Under-determined	R2.17	…there is a subjective artful judgment about which one (tolerance) to cut it off
		R2.18	You have to agree that the fabricator will do things to compensate where he needs to compensate.
		R2.19	So we specify the joints so that the panels never fail structurally, and don't induce in-plane load so strong.
		R2.20	The steel stays crude to keep it cheap.
		R2.21	It prevents dirt from accumulating on the front of it, so the water can get absorbed by the sponge
	Un-determined	R2.22	The city is trying to mitigate plumbing impacts.

*B*esides several requirements related to the shape of the building, which will be omitted because they are not related to the prefab-panel, the last case study, ***100 & 10th Avenue,*** seems to follow the same tendency as those of cases 1 and 2. It also heterogeneously distributes requirements at different levels of determination along the design structure (Figure 4.15 & Table 4.17). The *determined TO* requirements define specific and clear features. While the panel *Assembly* node is linked to prefabrication requirements, the operable window and the steel structure sub-assembly nodes are linked to tilted installation angles and range of section dimensioning, respectively. The *under-determined TO* requirements are mostly linked to *sub-assemblies* that define general features subject to interpretation, such as conditions for the location of the main glass panels or differentiation considerations to preserve the collage aesthetics. The *undetermined* requirement of this section is based on an analogy suggesting that the reproduction of the aesthetics of the specular reflection of the sun on the water is directly linked to the same prefab panel that is highly modulated in terms of structure to achieve the same apparently random organization.

To prevent problems, the PF section assigns the *determined* requirements to *Sub-assemblies* while lifting the panel on site; avoid accidents derived from the positioning of the operable window close to the floor level, and prevent fire propagation from floor to floor. Again, the most un-determined requirement that requires interpretation and leaves room for creativity is linked to the overall *Assembly* and characterized the entire design:

"Originally, they (the client) didn't want a regular grid on the façade…. "

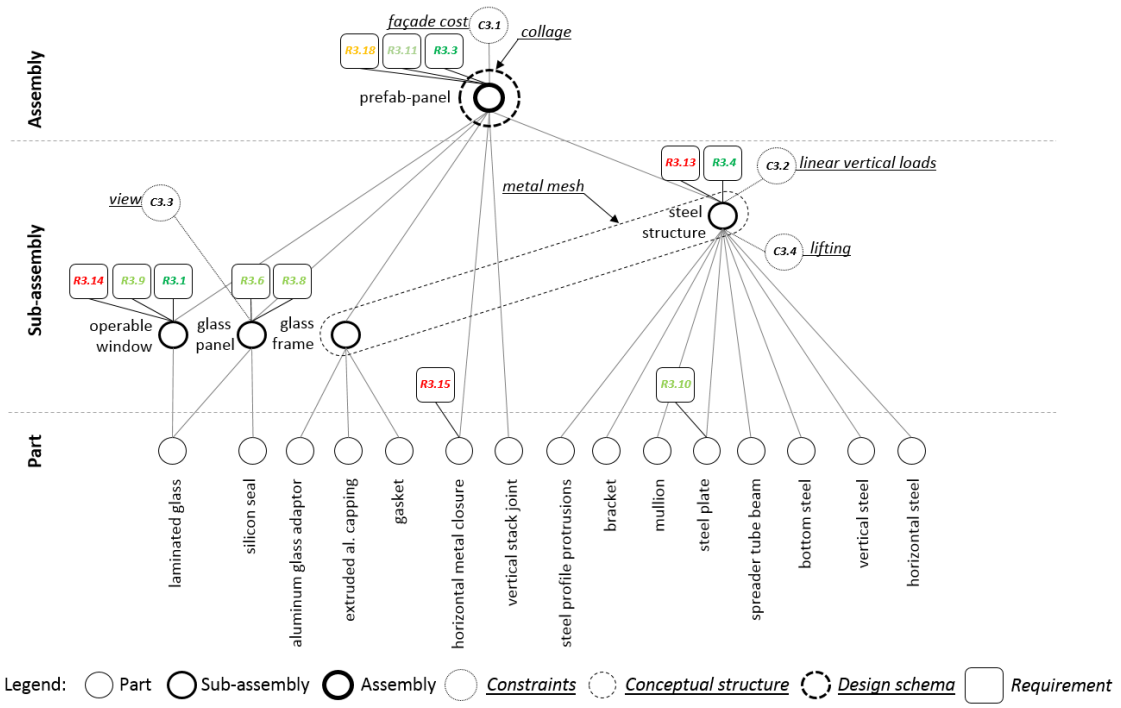


Figure 4.15. Map of requirements of case study 3, 100 & 10th Avenue

Table 4.17. Requirements of case study 3, 100 & 10th Avenue

Generalization	Specialization		Requirement
Target-oriented	Determined	R3.1	The operable windows should be installed at tilted angles
		R3.2	…the first 60' from the bottom of the building needs to maintain the street wall.
		R3.3	…each mega panel would be pre-assembled
		R3.4	Every single piece of steel except for the tube on top is ether 3 by 3, 3 by 4, 3 by 5, or 3 by 6.
	Under-determined	R3.5	We maximized the perimeter and organized the building with the core on the back.
		R3.6	… the intention of every single piece of glass was different from the intention of the adjacent panels.
		R3.7	…the steel has to support this street façade.
		R3.8	…we want our largest piece of glass close to the living room.
		R3.9	… you put your window close to the kitchen and satisfy your fresh air requirement.
		R3.10	The extrusions are welded and sanded smooth to maintain visual continuity between the mullions
	Un-determined	R3.11	(The collage is based on) the specular reflection of the sun on the water… The question was, can we imagine a façade which has such
		R3.12	So the idea was how to add some interest, architectural, to those (lower) floors.

Table 4.18. (continued)

Failure-preventive	Determined	R3.13	To hang precast you hang it in two points.
		R3.14	If it will have an operable window, you don't want it at the floor level.
		R3.15	There is waterproofing and insulation in the vertical stack joint, a horizontal metal closure for stopping fires.
	Under-determined	R3.16	…a thinner building that is not really deep enough to accommodate the depth of the units.
		R3.17	If you push your building back, you won't have this continuous façade.
	Un-determined	R3.18	Originally, they didn't want a regular grid on the façade….

4.5. The Problem of Implementation in Computational Environments

This chapter focuses on fostering a more complete understanding of the practical implications of the theoretical question about reusing design knowledge to better support designers in action while producing new design configurations. After searching for and distilling actual structural and behavioral knowledge and identifying aspects and requirements of the design problems, this study identified several challenges to the effective implementation in the computational environments of reutilization mechanisms. Challenges such as the incompleteness of the external representation of the designer's mental models, the use of abstract conceptuality to define problems, shifting perspectives during the design process, and the role of requirements that define not only restrictions but also alternative interpretations and avenues for creativity.

Incompleteness of the External Model of Design Knowledge

If we examine Figure 4.7, Figure 4.9 and Figure 4.11 they reveal the incompleteness of the description of the structures of the three case studies. Many of their intermedia sub-assembly nodes, such as balconies or brise-soleils, have no further decomposition in the transcriptions. Although they are obviously complex structures

from the perspective of the overall assembly, they act as single entities with a hidden internal structure. This hidden mechanism raises the following question: What is determining this level of abstraction as assembly, sub-assembly, part or even attribute? It seems that the chosen level of abstraction depends on the level of hierarchy of the insertion of the component into the assembly. This incompleteness is partially due to the process of capturing and distilling knowledge based on the transcription of the explanation by the designer. Despite detailed explanations by the expert, only a fraction of the mental model is delivered, and it depends on the focus of interest of the explanation. The focus of the *Seattle Library* tends to be the glass panel, that of the *Via Verde* the thermal expansion, and that of 100 & 10th Avenue the pattern of subdivision. Thus, the designer extends the explanation of these key issues while only implying all other general aspects or not mentioning them at all.

Another important characteristic of expert designers is that they integrate knowledge across fields and perform rough evaluations of the various aspects of possible solutions. They develop aesthetics and technical aspects by constantly shifting their perspectives. Most of the preliminary evaluations of the designs are driven by heuristics from previous experience, which are mostly low resolution estimations based on simple methods. These heuristics range from simple rules for dimensioning physical components to rough estimations of performance. Nevertheless, achieving preliminary evaluations also implies addressing the problem of partial definitions or incomplete information about the external model.

Forming the Design Problem

Based on cross referencing design structure with actions, we can infer that when facing a design situation, expert designers already know fragments of the design problem because of their previous experience. Our expert shows a clear tendency to recall many chunks of constraints when defining mainly parts and their assemblies and when applying constraints at the part and assembly levels as well. Case study 2 exhibits

some evidence of the sequence of adaptation of such recalled knowledge when the expert identified a condition when recalling was relevant, created the association of the recalled constraint and the new situation, and finally adapted the constraint. Table 4.13 also shows that the expert constantly applied design rules as a type of mechanism to rationalize a decision as the design proceeded. These two actions play an important role in forming the design problem. While the first defines what the restrictions are, the second determines how one addresses them. The challenge is how to consistently recall and assign constraints to the parts and assemblies in such a way that they create an organized arrangement of restrictions and rules that shape the problem.

Tradeoffs Among Requirements

The three case studies in the context of the incompleteness of the model, show similar logic of the distribution of requirements: from highly determined requirements at the top of hierarchy that define mandatory features for the subordinated nodes to complementary ambiguous and un-determined ones that are subject to interpretation. Such a combination demands a set of guidelines for further specification and specialization of the requirements in association with particular nodes, but it also leaves room for creativity and preferences. The tendency is that the accumulation of requirements, regardless of the type, around the key nodes that address the aspects that frame the focus of interest. However, during the design process, designers constantly change their perspectives of problems. They may address the problem from the perspectives of Physical, Performance, or Procedural aspects or any combination of them. During the decision-making process, expert designers address problems from myriad viewpoints and determine tradeoffs among several requirements. As perspectives change, multiple types of requirements linked to the same node appear to accumulate. This requirement stacking defines the context for tradeoffs among requirements attached to the same object since they establish different levels of determination regarding the resolution of whatever the node is. In addition, some of

the requirements are compatible, so we can link them using “and” operators; others represent alternatives and therefore “or” operators, or even conditionals defining a large set of possible combinations of requirements.

The Role of Abstract Patterns of Organization

Even though descriptions of *Physical Components* are always incomplete and seem to represent permanent work in progress, from the perspective of *Patterns of Organization,* the descriptions appear to include the entire complexity of designs from a conceptual approach. For example, while we can conceptually describe case study 1 as a *Wrapping Diagonal Grid* of micro *brise-soleils* over *Vertical Surfaces* and *Interstitial Zones* also performing as a *Lateral System* of the building, we do not have the entire list of parts and attributes. Nevertheless, the above description, based on a *Design Schema* and a few key *Conceptual Structures,* addresses the complete design in a very synthetic way. The challenge is to identify such an abstract structure that preserves the integrity of the design without the need for extensive and detailed descriptions and to establish the recall *of Physical Components* according to the level of abstraction.

Another salient feature is that *Conceptual Structures* show a tendency to embody design aspects that determine roles and tacit requirements at large. *Rain screens, wind structures, or brise-soleils* implicitly assume strategic roles in the overall system.

CHAPTER 5

META-MODELS OF DESIGN KNOWLEDGE

Overview

The proposed methodology addresses the problem of capturing design expertise from a design domain for further reutilization. It relies on meta-modeling techniques, that is, techniques that model information about the model. To do so, the methodology entails a meta-model of design knowledge, maps the objects of the meta-model with an actual repository of parametric models of physical parts, and creates design alternatives by recalling and combining knowledge. Actual design knowledge manipulated in this research comes from three case studies from the field of custom façade systems. While this chapter specifically focuses on capturing and modeling distilled knowledge from case studies, the next chapter introduces a new generation of solutions in the early design stages.

To describe the objects that shape the design domain, the Model- Based System Engineering (MBSE) process is adopted. MBSE is a model-based process that classifies information into a structure that refers to various system components, the behavior of the system of components and their interactions, and requirements that the system of interest is supposed to satisfy. The proposed methodology uses the object-oriented System Modeling Language (SysML) that makes multiple resources that support the creation of objects and instances, generalizations and specializations, and extensions accessible. It also enables human assessment through multiple kinds of graphic representations of the meta-model (Figure 5.1). Finally, this chapter discusses the implications of the convergence of the three case studies of the same domain into general categories that constitute the emergent framework of the fundamental entities of the domain.

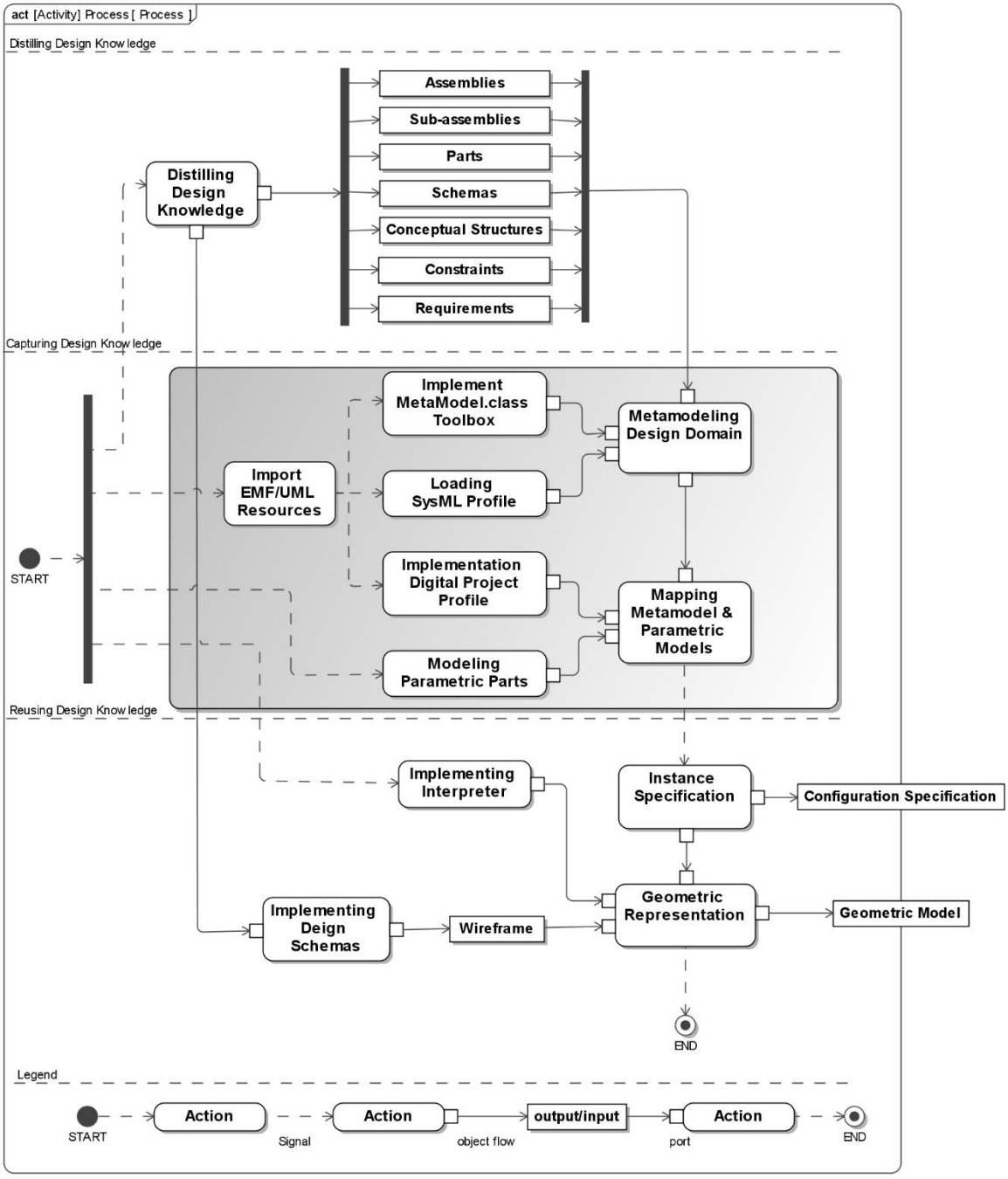


Figure 5.1. Capturing design expertise and modeling the design domain in the context of the overall process

5.1. Approaches to Meta-modeling

In the context of the general problem of this research on capturing and reusing design expertise, the purpose of building a model of a design domain is to identify, classify, and facilitate access to knowledge in order to manipulate and reuse it to produce drafts of design configurations. The meta-modeling process adopted in this research is rooted in complementary conceptions and perspectives of the modeling activity. The most common understanding of a model is that it is an abstraction of an aspect of a real-world phenomenon. According to the early definition of computer models of Kalay (1989), a model can be defined as a representation of these aspects using symbolic structures that allow their manipulation. From a Model-Based System Engineering (MBSE) perspective, a model is an approximation of aspects of the structure, the behavior, or the operation of a process, a concept, or a system (Reichwein & Paredis, 2011); and more precisely, a meta-model is the model of the data of the model (Kühne, 2006). Kalay also points out that a computational model should satisfy four conditions*: well-formedness,* which guarantees correspondence between the model and its representation, a *generality* that allows the model to represent a variety of objects, *completeness*, which allows the representation of all the necessary features of the aspect of interest, and *efficiency*, which minimizes the required computational resources for implementation. On the MBSE side, Reichwein & Paredis assert that the model should avoid *ambiguity*, satisfy *accuracy* in terms of the correct representation of objects, and achieve *precision* in the level of detail.

This research explores the use of computer-interpretable meta-models that capture, integrate, generalize, and manipulate design knowledge. The data of the design knowledge of this study is distilled from three case studies in the field of custom façade systems presented in Chapter Four. The proposed meta-modeling process of design knowledge relies on principles of *abstraction* of the declaration of the objects of the design domain that facilitate generalizations, *mapping* these abstract definitions with

multiplicity of external representations and the *continuous growth* of the repository of design knowledge.

Abstraction

The proposed methodology for meta-modeling provides an integrated repository for distributed design knowledge of a well-defined domain. This design knowledge from the case studies shows a taxonomy of objects that range from detailed specifications of physical components to very abstract conceptuality. To define the level of detail of the specifications of the meta-model, distinguishing the semantics, or meaning of the objects, from the complexity of their representations through any kind of computational tool. While a design concept is a generalization based on semantics, the representations can differ markedly, depending on the type of object and the means of representation.

In this regard, the meta-model is defined in three layers of abstraction (Figure 5.2): object specification, geometric representation, and the mapping between the two. Every layer hides information from the next one (Parnas, 1972). The first layer captures the design concept, the properties, the attributes, and the relationships in a non-system specific language (Schaefer, 2006), avoiding any reference to the final means of representation such as parametric models or any kind of computational implementation in order to preserve the generality of the definition specified in the meta-model. The bottom layer corresponds to the actual representation of the objects through parametric tri-dimensional models or function. The final geometric model does not deal with any conceptuality or specification of objects with parametric modeling issues. This abstraction allows multiple mappings between a design specification and computational tools.

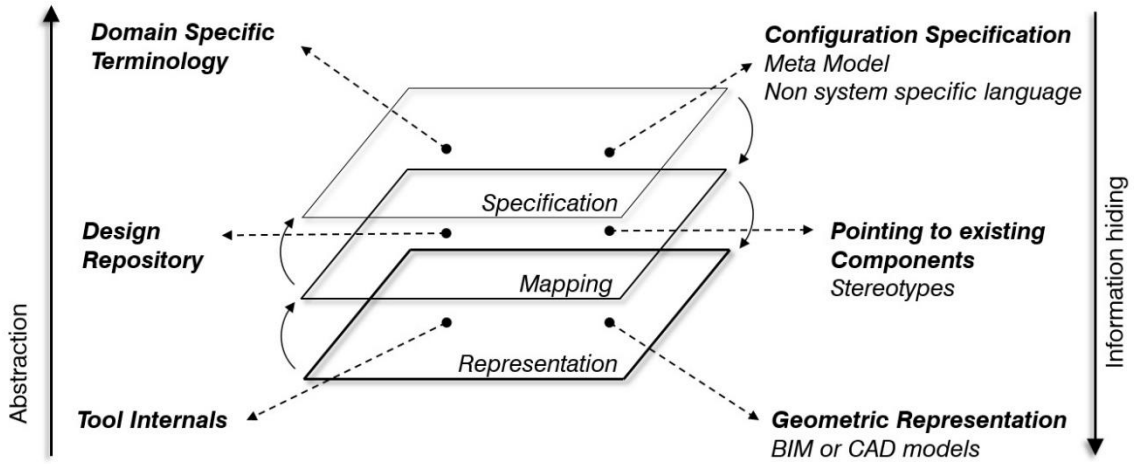


Figure 5.2. Layers of abstraction

Mapping for Multiplicity

While design knowledge is internally represented in the minds of experts, the external representation of the same knowledge expands to multiple formats. The mapping approach addresses the link between these external representations (i.e., 3D models or requirements) and the abstract specifications of objects in the meta-model. It maps the taxonomy of objects in the domain of the meta-model by pointing to specific CAD format files that address all of the complexities of geometric modeling. The distilled knowledge can be represented by any tool through proper mapping that links domain-specific definitions with the local requirements of the tool. This mapping potentially produces a multiplicity of external geometrical representations for the same domain definition. Although CAD or BIM tools can represent objects by various modeling methods, from the designer's perspective, they signify the same. Therefore, the generalization of the specifications in the meta-model must be capable of anticipating multiple possible external representations.

Continuous Growth Based on Modularity

Capturing design knowledge is not a unique task. On the contrary, it is a process of continuously extending the scope of the meta-model and complementing

incomplete declarations through time (Steinberg, Budinsky, Merks, & Paternostro, 2008). Continuous growth (Tapscott & Williams, 2008) implies extending the capabilities and the scope of existing models, which can be constantly refined to increase the accuracy of the abstract representation of embedded design knowledge. Extending existing objects facilitates customization and standardization based on already existing modules. Such modularity is based on the relationship between functionality and the physical structure supporting it. An assembly performing one main function can be decomposed into several sub-functions. From such a perspective, every part of the system can be a module addressing a specific or group of sub-functions. Thus, modularity enables changes in functions, internal rearrangements, and replacements of physical components. Modularity also provides interfaces that address these internal changes, enabling the system to provide variety of outcomes that minimize effort by promoting changes at the local modular level rather than across the entire system. Rules and constraints describing the relationships among parts can be continuously added, which leads to gradual changes in explicit, non-declared tacit rules. Complex parametric objects are also assemblies of individual parts with a modular architecture. This modular decomposition allows updates to the meta-model with regard to the replacement, the refinement, and the extension of components. All of these features facilitate the standardization and continuous growth of the meta-model of the expert design domain.

5.2. Methodology for Meta-modeling the Design Domain

The need for modeling the design knowledge domain is based on recognition that internal mental representations of knowledge and external representation through any means are not the same (Eastman, 2001a) As discussed in Chapter Two, while a mental representation supposedly captures the entire design knowledge of the expert, the externals capture only the subsets. Furthermore, although these external representations facilitate communication and the creative manipulation of knowledge,

they differ in type and purpose, and they also extend to all numerous digital formats. The proposed meta-modeling process attempts to provide an external computer-interpretable model that integrates knowledge from myriad sources and gradually extends the scope of the representation to reduce the gap between the mental representation of design knowledge and external representations.

The platform for the proposed methodology for the meta-modeling of design domain knowledge entails an object-oriented non-system specific language that allows independence from any CAD or BIM tool, which reduces the complexity of geometric models. The System Modeling Language (SysML), an extension of the Unified Modeling Language (UML) for software engineering, provides capabilities for modeling and visualization through either graphic visual interfaces or programming languages that allow access to its resources, which enable the automatic generation of objects of the meta-model. With regard to the parametric modeling of the physical components, the chosen tool is Digital Project from Gehry Technologies. Digital Project is the tool that the expert, Marc Simmons, regularly uses in his professional projects. The meta-model and the repository of parametric models are linked through *Stereotypes*, which are extension mechanisms of the language that maps the correspondence between abstract definitions and the actual geometric models of parts.

The Adoption of the MBSE Process

This research intends the meta-model to be an abstract representation of the main objects of the design domain of custom façade systems, their attributes, and their cross relationships. These objects can be immaterial patterns of organization, physical components, design rules, or any other relevant entity that embeds design knowledge. Building a meta-model contributes to the structure, organizes information from a variety of sources, and integrates non-geometric information that includes abstract objects representing design concepts. Since the main purpose of creating the meta-model is to reuse design knowledge, the definition of a taxonomy of objects should constitute a

repository that is most likely to allow the integration of similar objects through time and the extension of existing definitions that represent particular cases.

Reichwein & Paredis (2011) provide a detailed survey of the main concepts of the MBSE that are the core of its modeling process. In addition to the already discussed definition of the *model*, concepts such as *system*, *system of systems, system view, view point, system architecture, system modeling language,* and *architecture framework* play distinct roles in the constitution of the meta-model and derived interpretations. Although these terms originated in the context of MBSE, they can be adapted and extended to address other domains of knowledge such as architectural design.

While a *system* in MBSE corresponds to the collection of parts and also the functionalities they support, a *system of systems* is the decomposition in subsystems, usually related to interdisciplinary large- scale problems. In this regard, the façade domain matches the above definition since it is a collection of physical components organized into parts, sub-assemblies, and the overall assembly according to specific functions, installation, and fabrication, among other requirements. The notion of *system view*, which refers to a representation of the entire *system* from a particular perspective, and the notion of *viewpoint,* which is a specification of a view according to the needs of a stakeholder, correspond to the notion or *Design Aspects*, which defines the perspective from which the design is defined. As discussed in Chapter Four, these views are constantly changing while the design proceeds. Furthermore, we can find examples from the MBSE that also integrate multiple views or perspectives in design tasks (Jobe et al., 2008; Shah, Kerzhner, Schaefer, & Paredis, 2010). The concept of *system architecture* corresponds to the arrangement of objects defined from multiple views or perspectives that provide the solution. Although *system architecture* is not a *pattern of organization*, it represents the internal logic of the composition of a system. This type of *architecture* is represented through notation *modeling languages*, which will

be discussed at length in the following section. Finally, the *architecture framework* corresponds to a minimal set of content derived from best practices that define the generic objects from which the domain can be extended. This concept is particularly intriguing since it distills and gathers a set of fundamental definitions that determine the core of a design domain.

The tasks of this study, therefore, are to model the custom façade system from multiple views based on distilled information from the case studies in order to identify their architecture and to distill the framework so that it can be generalized and extended. The modeling task addresses the common and singular objects identified in the case studies, their attributes and values, their generalized relationships, and their part-composition relationships. These objects range from physical components to a variety of abstract entities.

Modeling with Object-Oriented Non-System-Specific Language

The need for a non-system-specific language stems from the diversity of design concepts that exceed the capabilities of individual computational tools to represent a single system comprised of the entire scope of entities that define a design domain. The BIM paradigm also acknowledges the same need for system independence by developing IFC schema based on the STEP standard that, similar to other neutral file formats, allows data exchange among the tools. However, interoperability approaches have less impact on conceptual design stages since the information does not necessarily adhere to industry standards, it is not well structured, and it uses abstract conceptuality. MBSE, by contrast, uses high-level object-oriented language that can be adapted and extended to describe abstract entities with partial definitions.

The System Modeling Language, or SysML, developed by the Object Management Group ("OMG Systems Modeling Language," 2015) to support MBSE, provides semantics and representations of meaning. SysML is a general purpose language for the design, specification, analysis, and verification of complex systems. It

provides graphical notations that represent the structure, behavior, and requirements of a system of interest. As a subset and an extension of the object-oriented Unified Modeling Language for software engineering ("Unified Modeling Language®," 2016), it satisfies the needs of the community of the International Council on System Engineering ("INCOSE," 2016). SysML can graphically represent the structure of any system through *Structure Diagrams* such as the *building block diagrams* (bbd), the behavior through *Behavioral Diagrams* such as the *activity diagram* (act), and requirements via *Requirement Diagrams.* All of the diagrams can be created using a graphical editor such as Magic Draw ("MagicDraw," 2016).

The fundamental unit of SysML is the *block*, which is an extension of *class* in UML. The *block* can represent any material or abstract object of the structure of a system. A user can specify the attributes, slots, and parts of the block through the diagrams or program the blocks of the model by accessing the UML libraries using a programming editor such as Eclipse for Java developers ("Eclipse," 2016). For the purpose of this research, the meta-models in SysML are implemented taking advantage of the UML API, which guarantees the consistency and integrity of the models and facilitates access to information for further manipulation. UML resources include methods of creating objects and related attributes as well as generalizations, aggregation, and associations of the composition. The resulting models can be plotted in the MagicDraw (MD) editor for visual evaluation and communication.

The Modeling Language Resources

The setup of the computational infrastructure begins in the Eclipse Java editor, which imports the Eclipse Modeling Framework (EMF). The EMF supports the creation of applications based on structured data models such as UML-based models ("Eclipse Modeling Framework," 2016). Eclipse imports the UML2 Model Development Tool (MDT), which is based on the EMF, enabling the use of UML resources in the Java environment. To assure the integrity of the domain meta-model and to minimize human

error, Eclipse executes the process of building the model using the UML libraries in the Java programming environment and translates it into SysML. The conversion of UML models into SysML is executed by adding the already existing SysML Profile to the UML model. A profile is a group of stereotypes that comprise the particular extension mechanism of UML for the customization of classes or attributes that adapt the language to various domains. The stereotypes basically assign specific domain features to generic definitions. In fact, SysML has been extended from the UML by developing a profile (or collection of stereotypes) that adds specificity to the UML object according to the needs of the MBSE community (Figure 5.3). A detailed list of imports related to the creation of the meta-models can be found in Table 5.1. They are regular JAR (Java Archive) package files that aggregate Java classes representing UML resources in the Java work space.

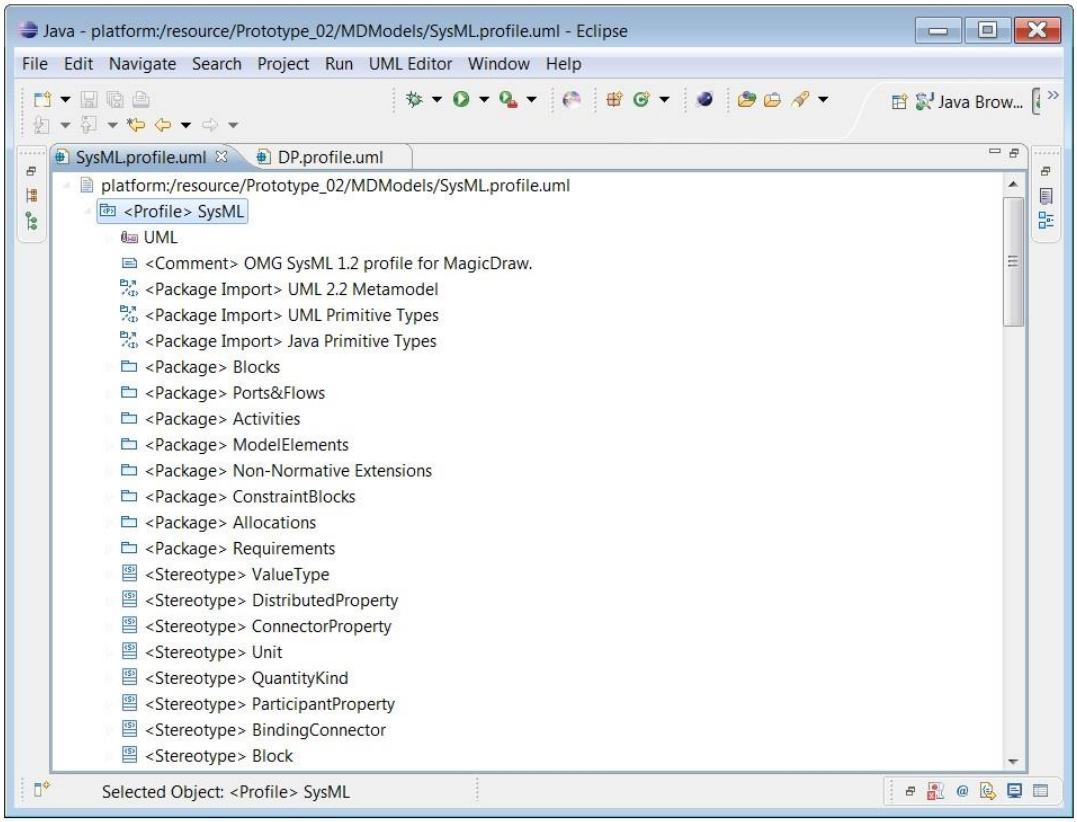


Figure 5.3. Sample of the SysML profile

Table 5.1. Specific resources of MDT UML2

import	***Definition from OMG***
org.eclipse.uml2.uml.**Model**	*A model captures a view of a physical system…Thus, the model completely describes the aspects of the physical system relevant to the purpose of the model at the appropriate level of detail.*
org.eclipse.uml2.uml.**Profile**	*A profile defines limited extensions to a reference meta-model with the purpose of adapting the meta-model to a specific platform or domain.*
org.eclipse.uml2.uml.**Stereotype**	*A stereotype defines how an existing metaclass may be extended and enables the use of a platform or domain-specific terminology or notation in place of, or in addition to, the ones used for the extended meta-class.*
org.eclipse.uml2.uml.**PrimitiveType**	*A primitive type defines a predefined data type without any relevant substructure (i.e., it has no parts in the context of UML). A primitive datatype may have algebra and operations defined outside of UML, for example, mathematically.*
org.eclipse.uml2.uml.**Class**	*A class describes a set of objects that share the same specifications of features, constraints, and semantics. A class may be designated as active (i.e., each of its instances has its own thread of control) or passive (i.e., each of its instances executes within the context of some other object). A class may also specify which signals the instances of this class handle. A class has the capability to contain an internal structure and ports. Class has a derived association that indicates how it may be extended through one or more stereotype. Stereotype is the only kind of meta-class that cannot be extended by stereotypes.*
org.eclipse.uml2.uml.**Property**	*A property is a structural feature of a classifier that characterizes instances of the classifier. A property… represents an attribute and might also represent an association end. It relates an instance of the class to a value or a set of values of the type of the attribute…*
org.eclipse.uml2.uml.**Association**	*An association describes a set of tuples whose values refer to typed instances. An instance of an association is called a link.*
org.eclipse.uml2.uml.**Generalization**	*A generalization is a taxonomic relationship between a more general classifier and a more specific classifier. Each instance of the specific classifier is also an indirect instance of the general classifier. Thus, the specific classifier inherits the features of the more general classifier. A generalization, owned by the specific classifier, relates a specific classifier to a more general classifier.*

Implementation of the Meta-model

Once the UML resources are imported into the Eclipse environment, a class with the necessary tools for building the meta-model is created. This research refers to that Java Class as *MetaModel* since it the purpose is to provide the required methods for building a model by taking advantage of the specific EMF and UML2 imports described in Table 5.1. Like the internal complexity of each method of the *MetaModel* class, the process of creation of the meta-model is executed according to the sequence described in Table 5.2. This example meta-model has one *Assembly* made by two *Parts* and attributes that will be further extended to represent the case studies in the following sections.

Table 5.2. Methods of the Structure class for programming the meta-model

Method execution from *MetaModel* Class	Comments
createModel(String Metamodel)	Crating and naming the model
load(Profile sysmlProfile)	existing SysML profile
applyProfile(Metamodel metamodel, Profile sysmlProfile)	applying profile
sysmlProfile.getOwnedStereotype(String Block)	retrieve the SysML Block
createPrimitiveType(Metamodel metamodel, String string)	or “Integer” or “Boolean”
createClass(Metamodel metamodel, String Assembly)	or “Part”
createAttribute(Part part, String attributeName)	create attributes
createAssociation(Assembly assembly, Part part)	Assembly made of parts
createDirectedAssociation(Part part, Part part)	“has” relationship
createAggregation(Part part, Part part)	
createDirectedComposition(Part part, Part part)	“made of” relationship
createComposition(Part part, Part part)	
createGeneralization(Assembly assembly, Assembly subassembly)	generalization
applyStereotype(Assembly assembly, Stereotype Block)	or to a Part
save(Metamodel metamodel)	save the model

Figure 5.4 illustrates the resulting meta-model in the Eclipse editor. Although this example meta-model is generated in UML using the Java programming environment,

the applied Profile converts it into a SysML model that is imported to the MagicDraw graphic editor. While Figure 5.5 shows the Magic Draw SysML version of the same meta-model in the containment tree of the editor, Figure 5.6 shows the SysML block definition diagram of the model. Since the diagram is only a view of the meta-model, it could have a similar or lower degree of resolution by representing the entire set or subset of blocks and attributes. In both representations, we can see the property fields of the Blocks, their types, their multiplicity, and their associations.

The input values of the object are properties that can comprise strings, integers, doubles, Booleans, and similar primitive types. The multiplicity notation of the block represents how many objects participate in the associations. For example, [1] represents a single object zero to many [0..*] or one to many [1..*]. Associations with other objects define the nature of the relationship. While the directed composition (arrow with the black diamond) in Figure 20 indicates that the Assembly is made of zero to many, Part_2 objects and Part_2 can belong to one Assembly. The same Assembly can have (arrow with the white diamond) from one to many Part_1 objects and Part_1 can be used by one to many Assemblies. Finally, the header of each block also shows the assigned DP_Assembly or DP_Part Stereotype that will be discussed in the final subsection.

This modeling procedure, which creates the meta-model by taking advantage of the UML Resources supporting the SysML model, is applied to an example design domain based on the three case studies. The tree representing the example meta-model shows the application of the SysML Profile at the bottom, the Type of the object within single- angle brackets, and the SysML *block* stereotypes applied to the UML classes within double-angle brackets. In addition, each class shows its property types and associations.

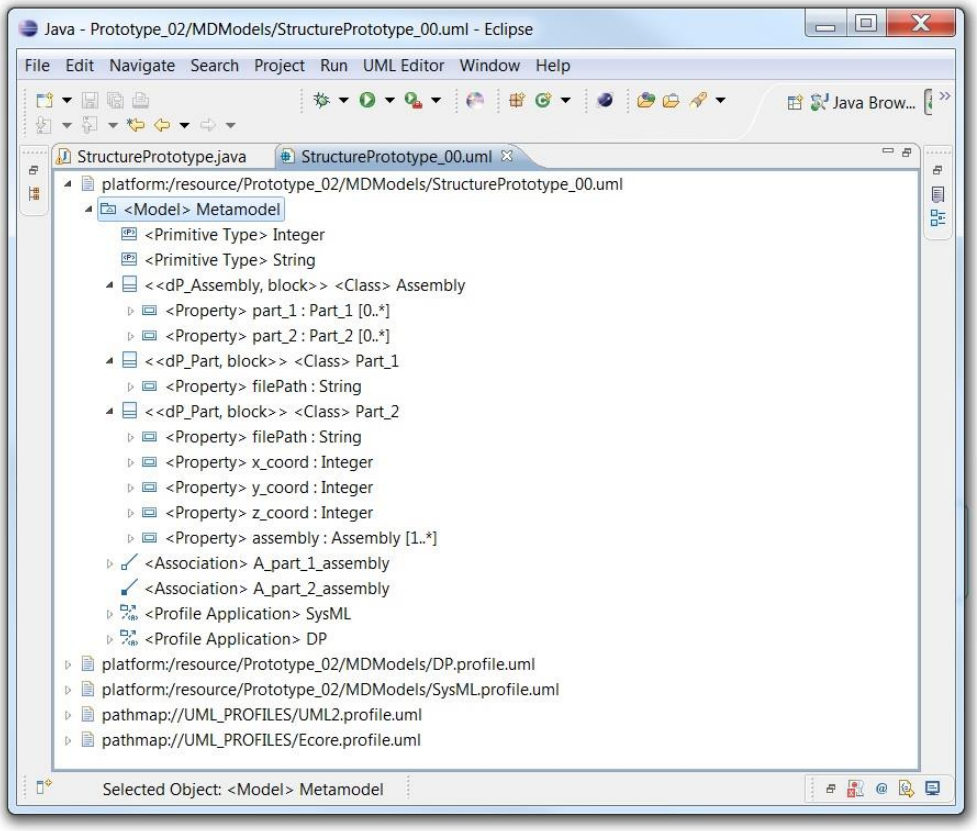


Figure 5.4. UML version of the meta-model in the Eclipse Java editor

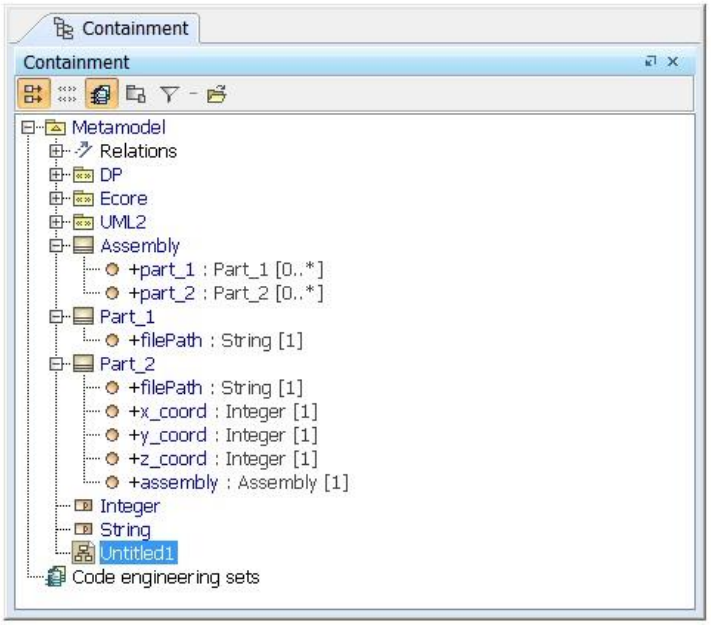


Figure 5.5. SysML version of the example meta-model imported into MagicDraw graphics editor

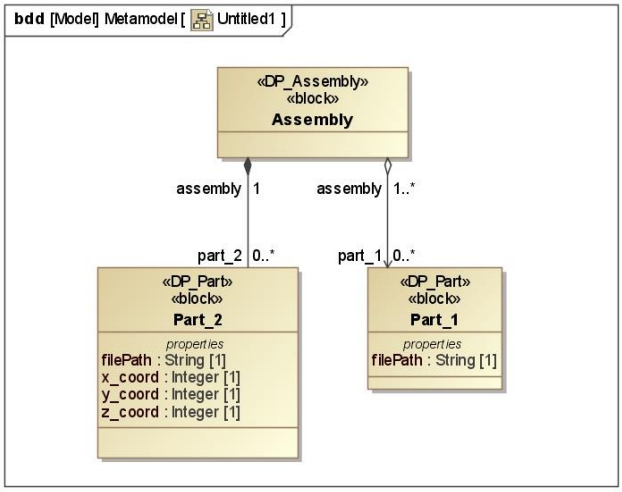


Figure 5.6. Example of the SysML Block Definition Diagram representing the meta-model in MagicDraw

Mapping Between the Meta- and Parametric Models

With a meta-model that describes the objects of the domain in abstract general terms and the logical step is to link them with a repository of actual parts, this link is through the implementation of an interfacing *Profile*. For every *Physical Component* described in the meta-model exist a *Stereotype* that links the abstract description with the chosen tool. For the purposes of this research, an entire *Profile* called *DP.profile* has been implemented. It groups all of the stereotypes that interface the meta-model and the Digital Project BIM tool (Figure 5.7). The example shows two Stereotypes in the *Profile: DP_Assembly* and *DP_Part.* In addition to other properties, Stereotypes have a key property called "filePath", which is a *String* that defines the location of the corresponding CAD or BIM file in the repository of *Physical Components.* The *Stereotypes* are independent from the meta-model domain (Shah et al., 2010) because design concepts can be modeled with different tools in multiple ways. For example, what a column internally means in the expert's mind can be externally represented as a simple extrusion or as a complex parametric object, depending on the kind of tool we are using for geometric representation. While the extrusion is more suitable for CAD tools, the parametric object is for BIM tools. This difference clearly defines the boundary between the meta-model of the domain storing the design knowledge and versions of

the objects externally represented. In fact, with this property, *multiplicity*, different objects of the meta-model can have multiple *stereotypes* depending on the selection of the tool.

Although Profiles and Stereotypes can be manually created, this approach to programming the meta-model and its mapping mechanisms has been successfully tested and subsequently adopted in the aerospace industry (Bohnke et al., 2009), as it guarantees the integrity of the resulting model. A second Java Class called *DPprofile* contains the methods that enable the creation of such a profile and UML resources.

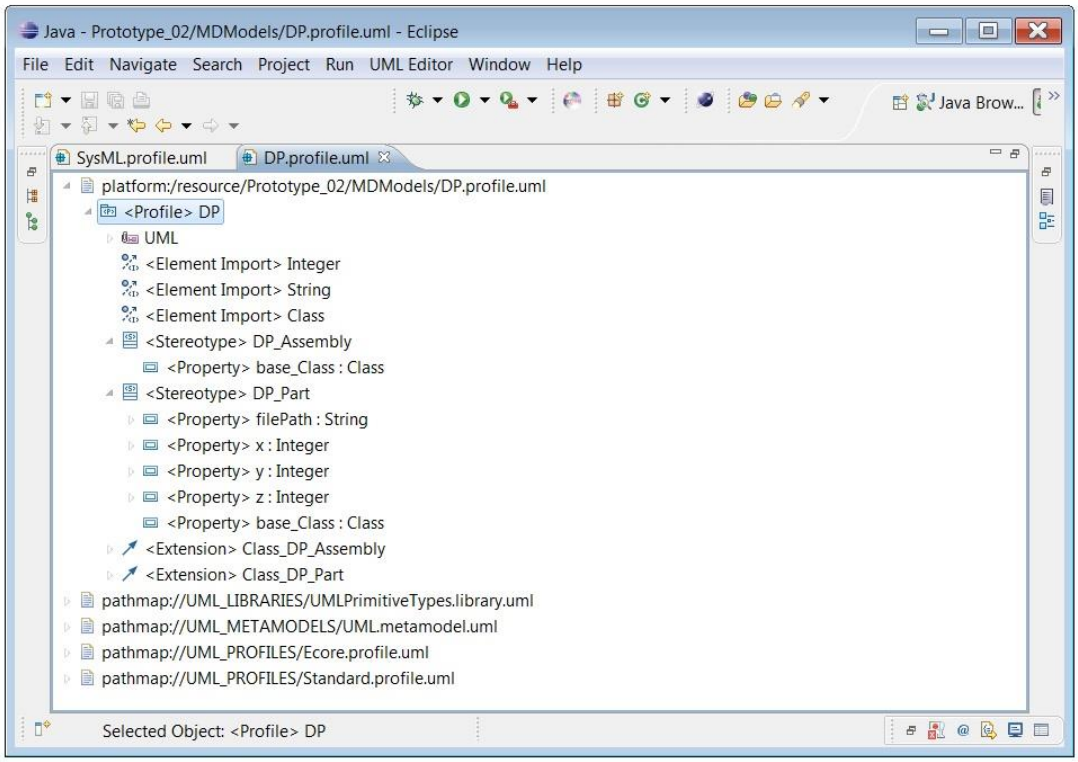


Figure 5.7. Profile and Stereotype for mapping the meta-model with the Digital Project BIM tool

Imported resources enable the capability of creating *Profiles* and *Stereotypes* in the Java environment and attaching them to the meta-models through build-in methods. It requires an entire Java Class that operates as a tool box with all of the static methods for creating a custom *Profile* interface with the tool, which creates the *Stereotypes* and their attributes (Table 5.3).

Finally, once the *Profile* is complete, it is applied to the meta-model, and the corresponding *Stereotypes* are applied to the UML Classes. Figure 5.8 shows the two Profiles, SysML and Digital Project, applied to the same meta-model. Additional stereotypes will be added to the profile to map sub-assemblies, conceptual structures, and design schemas with their corresponding BIM models

Table 5.3. Sequence of creation of Profile mapping with a BIM tool

Method execution from *DP profile* class	Comments
createProfile(String DP)	interfacing with Digital Project BIM tool
createPrimitiveType(Profile DP, PrimitiveType string)	or Integer
createStereotype(Profile DP, String DP_Part)	*or "DP_Assembly" or any other required Stereotype*
createAttribute(Stereotype DP_Part, String filePath)	*file location*
setValue(Stereotype stereotype, String propertyName, Object newValue)	*Set the stereotype property specific value*
Save(Profile DP)	

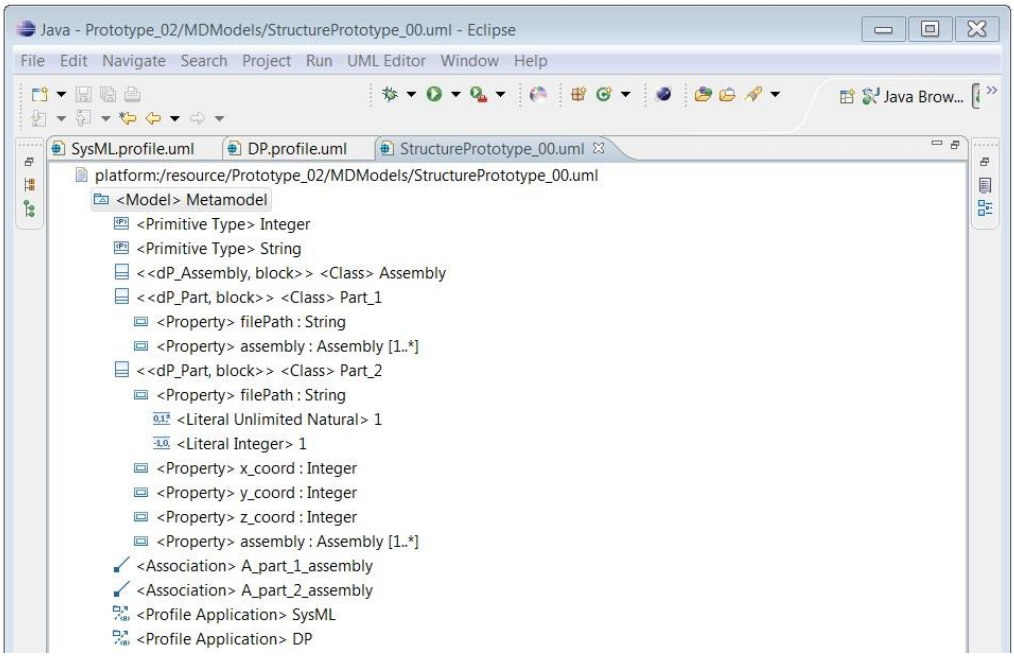


Figure 5.8 Applying SysML and the DP profile to the meta-model

5.3. Case Study Integrated Meta-model

The actual meta-model of this research is based on the integration of distilled objects from case studies. Creating the meta-model and accessing the repository of *Physical Components* involves three steps. First, the *Physical Components* SysML *blocks* must be implemented and specialized for the *Assemblies*, *Sub-assemblies*, and single *Parts*. Patterns of Organization follow a similar process. Second, the set of *Parts* and *Sub-assemblies* must be geometrically modeled in a parametric tool or retrieved from an existing repository. For this research, the parametric models of these Parts correspond to lighter and simple versions of the actual models, with a sufficient level of detail to efficiently be re-called and adapted. Finally, a specific *Profile* containing *Stereotypes* that links the description embedded in the blocks and the Digital Project parametric files must be implemented.

The following subsections present the *Architecture* of the design domain meta-model illustrated on a series of building-block diagrams created using the Magic Draw visualization tool. The reader should be aware that while the model, stored in memory, can be fully visualized through UML or SysML tree structures, the diagrams are only partial views of the model. These views include only the blocks, the focus of interest. Figure 5.9 shows the parallelism of the two versions of the meta-model. On the left, the UML version with the applied Stereotypes maps the Digital Project parametric models and the SysML blocks. On the right is the resulting SysML translation of the model. Both versions highlight the same Collage Panel object.

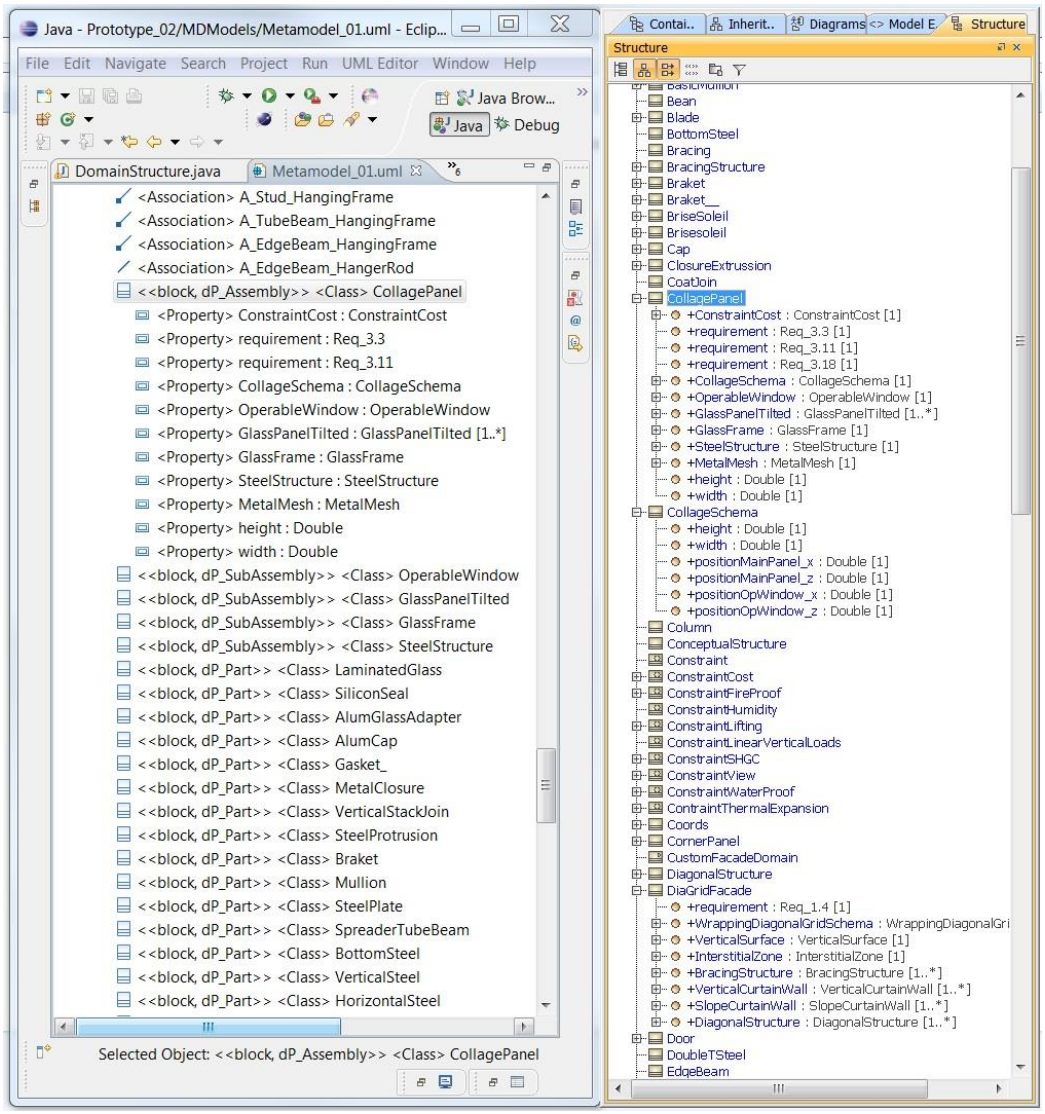


Figure 5.9. UML model to the left, SysML model to the right

Physical Component Building Blocks

Chapter Four shows the distilled objects of every case study in graph-like diagrams (Figure 4.7, Figure 4.9 & Figure 4.11) with three main categories that classify the Physical Components nodes*:* Assembly, or the final product; sub-assemblies, or the compositions of parts,; and single Parts. These blocks capture the domain knowledge in terms of object composition and attributes that correspond to the physical structures of the *Design Domain*. It is a vast collection of types of components and their relationships.

The Assembly Blocks

The following diagram (Figure 5.10) presents an overview of the composition of the general *Assemblies* of each case study. They are composed by the following fields*:* parts, properties, constraints, and references. First of all, in this research, the SysML *part* field (lower case) differs the so-called *Part* (capitalized). While the *part* field includes any object such as *SubAssembly* or *Schema,* the *Physical Components* sub-category *Part* specifically refers to an indivisible component. Although in this model the type and the name of objects in the fields are the same, while the type of field is boldfaces, the name of the object is not. The number of objects per field are defined by the multiplicity notation.

The *part* field of the block establishes the relationships of *directed compositions*, indicating that the fundamental components that comprise the block. If the block is deleted, these parts will disappear along with it. The *properties* field captures any single attribute or parameter such as dimensions, coordinates, or *requirements*, which will be discussed in the following section in more detail with the *constraints* field that represent the restrictions that the block must satisfy. Finally, the *references* field represents the directed aggregation relationships, or the reference to an object that the block uses, but it does not belong to its internal structure. The difference between *directed aggregation* and *directed composition* can be illustrated through the following description: While the MegaPanel is made of (directed composition) a frame structure supporting the insulation protected by the rain screen, the balconies, brise-soleils and window types are optional features (directed aggregation). The approach to determining such associations is not predefined. They depend on the purposes and uses of the meta-model. In this model, all of the objects that may be used by several objects and that do not constitute the core structure of the objects they belong to are related by *directed aggregation* association. They can be also represented in the *reference* field or through the block in a compact form.

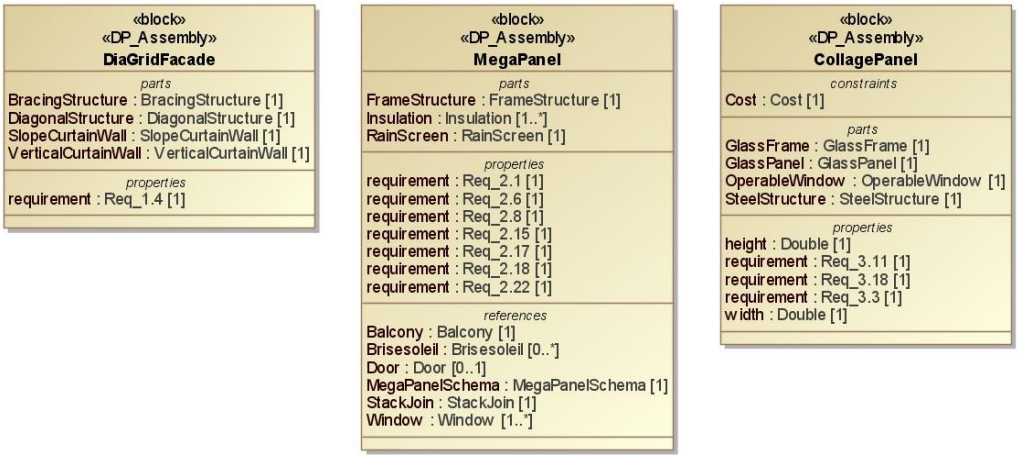


Figure 5.10. Final Assemblies block for every case study

Sub-Assembly Blocks

Sub-assembly blocks are *part* attributes of *Assembly* blocks that can be either represented as *part* field in the *Assembly* block similar to that in Figure 5.10 or deployed as actual SysML blocks. They represent higher-level objects consisting of many parts or even other sub-assemblies that play key roles in the overall system. In the diagrams, the *directed composition* association between the Assembly and its main sub-components is represented as an arrow with a black diamond in the beginning.

Case study 1 (Figure 5.11) can be synthesized into four major Sub-assemblies: two variations of the curtain wall system with its related structural support systems. Case study 2 (Figure 5.12) consists of three components*:* a metal frame structure with all connections to the building that supports the rain screen protecting the insulations. Brise-soleils, balconies, and various types of windows can be added to this well-insulated assembly. Case study 3 (Figure 5.13) is comprised of a steel structure that supports glass aluminum frames holding the glass panels and operable windows. All of these descriptions are highly synthetic and include the major issues of the cases, but they leave a great deal of information tacit. These top-down synthetic descriptions are based on abstractions and hidden information in the lower level.

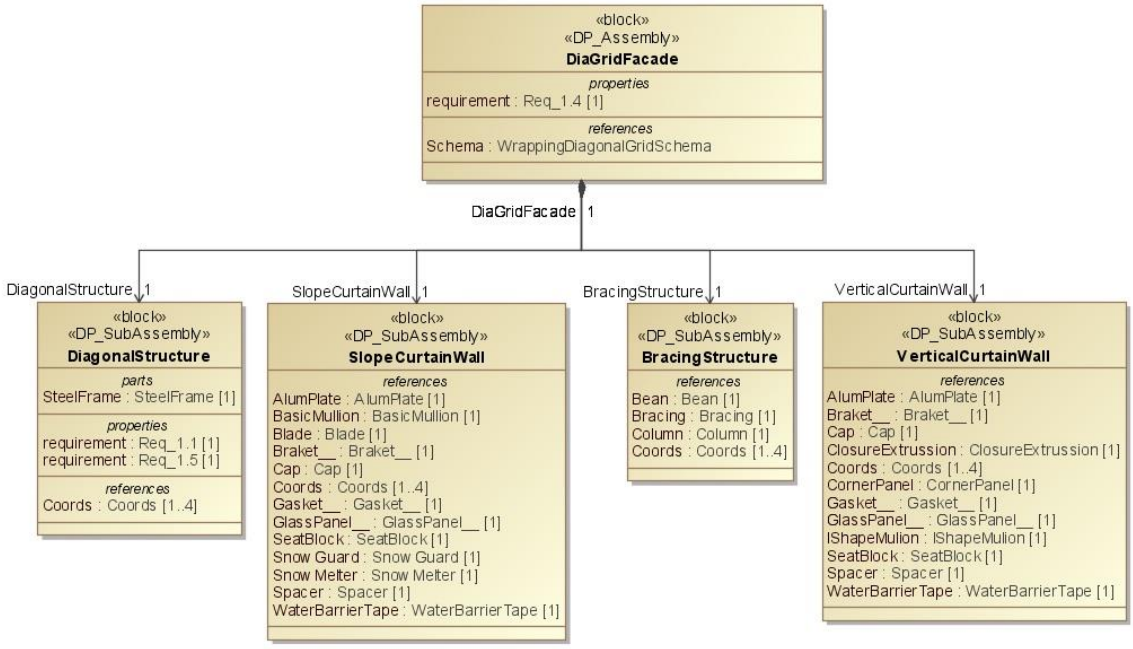


Figure 5.11. Case Study 1: Diagrid-directed composition associations with sub-assemblies

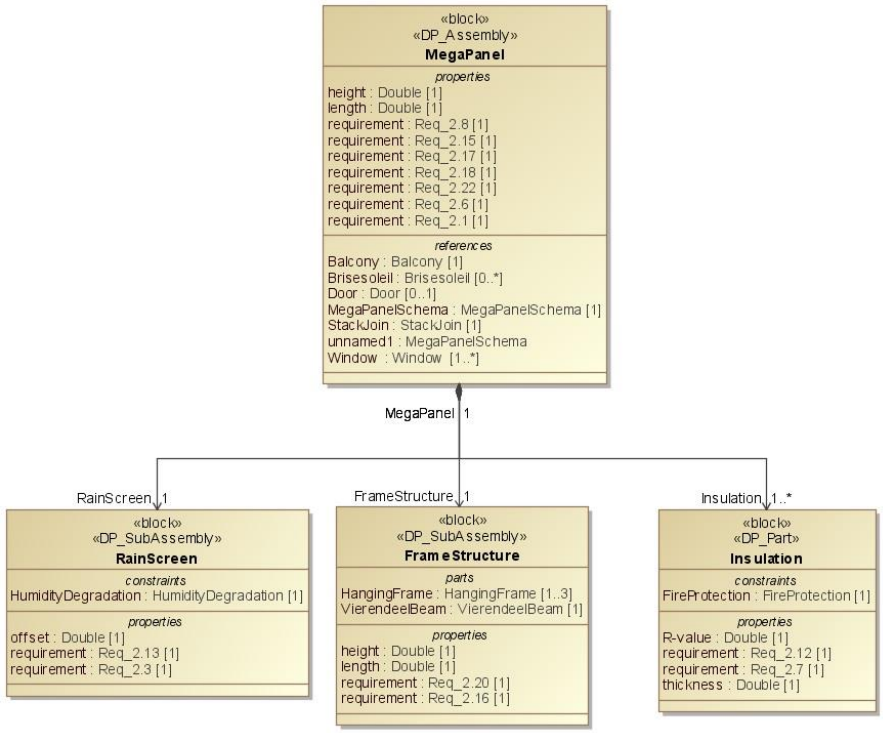


Figure 5.12. Case Study 2: MegaPanel-directed composition associations with sub-assemblies

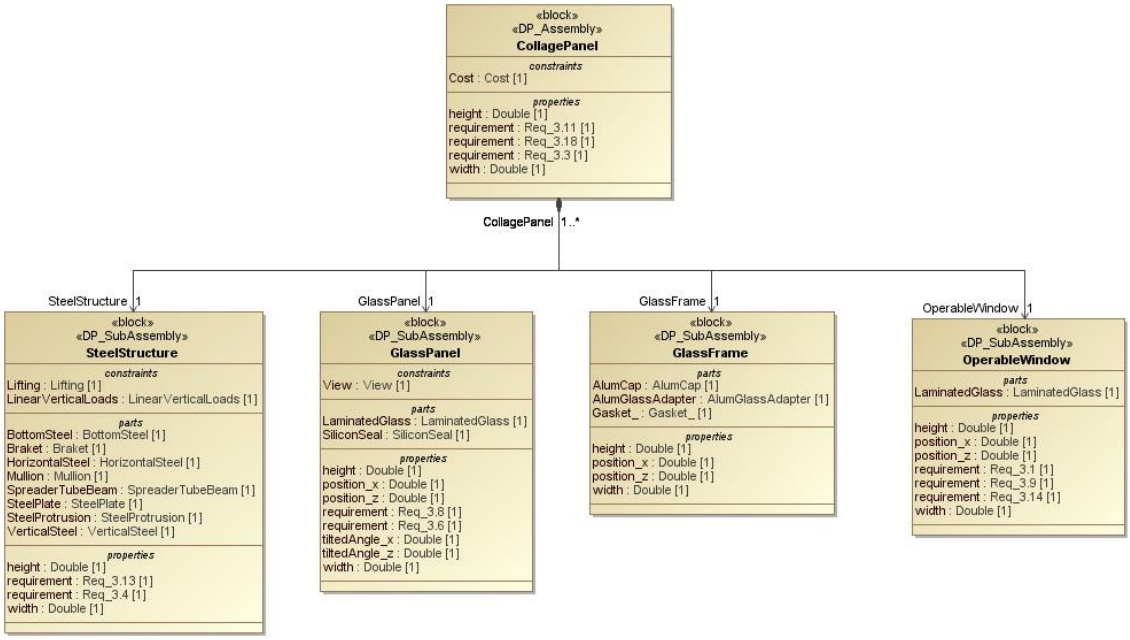


Figure 5.13. Case Study 3: Collage Panel-directed composition associations with sub-assemblies

The Part Blocks

Different from the *Sub-assemblies* characterized by implicit references to their functionality, *Part* blocks correspond to the final building components comprising the Sub-assemblies. Figure 5.14 is an example of decomposing a branch of the main *Assembly* of the DiaGridFacade until it reaches the final Glass Part component and its attributes. The narrative of the diagram states that one-to-three glass pieces of various types and one perforated metal mesh of various opening sizes compose a glass panel, many of which are used by one curtain wall of the facade. While some of the branches are fully and explicitly decomposed until the very final attribute of the terminal parts, other stops in the sub-assemblies maintain the tacit nature of its decomposition or only partially reveals their sub-parts (Figure 5.15). Case study 2 shows the MegaPanel, including references to several *Sub-assemblies* such as balconies, doors, windows, stack joints, and brise-soleils with no further decomposition. Although they seem to form a complex arrangement of parts, they contain no level of detail, and their internal components remain hidden.

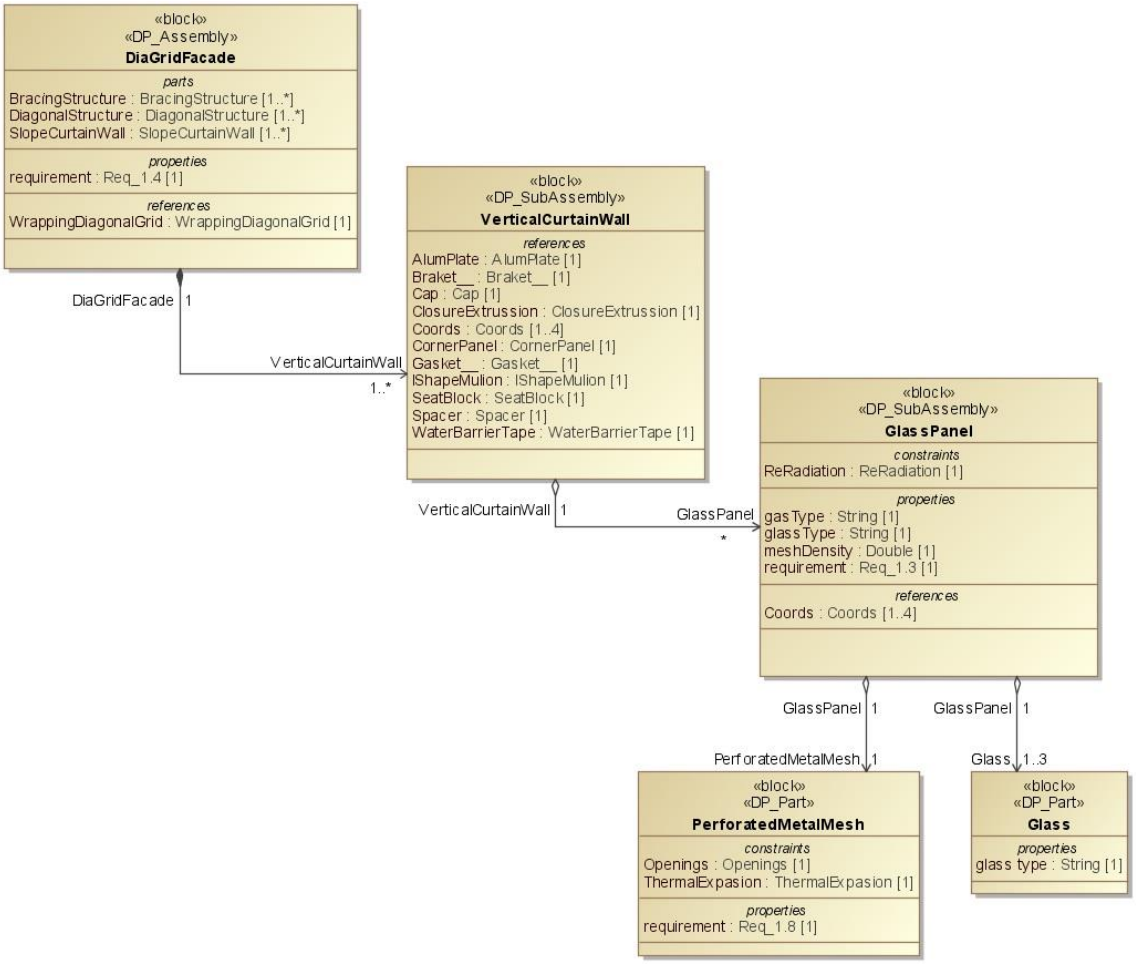


Figure 5.14. Case study 1: Extended branch showing the final leaf representing the Part components.

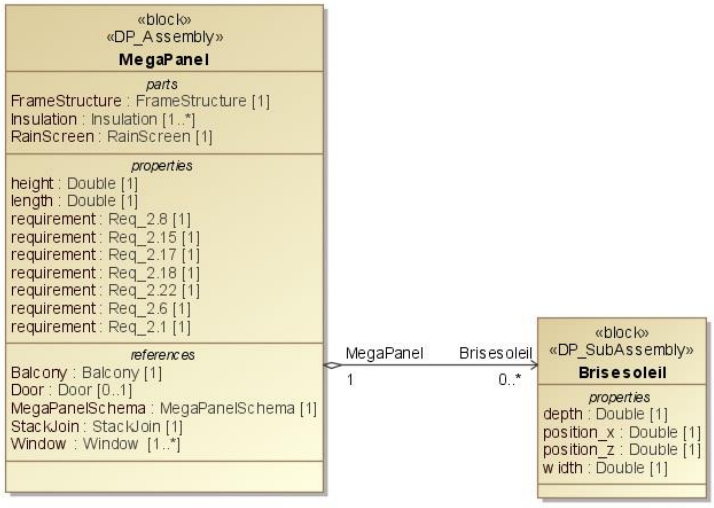


Figure 5.15. Case study 2: Aggregation branch decomposed until the ending Sub-assembly

Design Schema Blocks

The notion of schemata (Lawson, 2004) refers to an abstract entity even more abstract than a typology. A pattern of organization rather than a typology, it includes geometry and related relationships. In Simmons' words, the expert that provided the case studies, one of the most important elements in early design stages is what he calls the *wireframe* (Simmons, 2012). It is a digital model built by all the auxiliary geometry that organizes the *Physical Components*. The design schema blocks embed knowledge about the subdivision of the façade, the modularity, and other geometrical relationships. They are closely related to Lawson's notion of schemata. They define the arrangement of key points that drive axial lines, intersections, offset distances of reference planes, and parameters of building scaffolding to coherently instantiate *Sub-assemblies* and *Parts* for every *Assembly.* Figure 5.16 shows the schema blocks for every *Assembly* block.

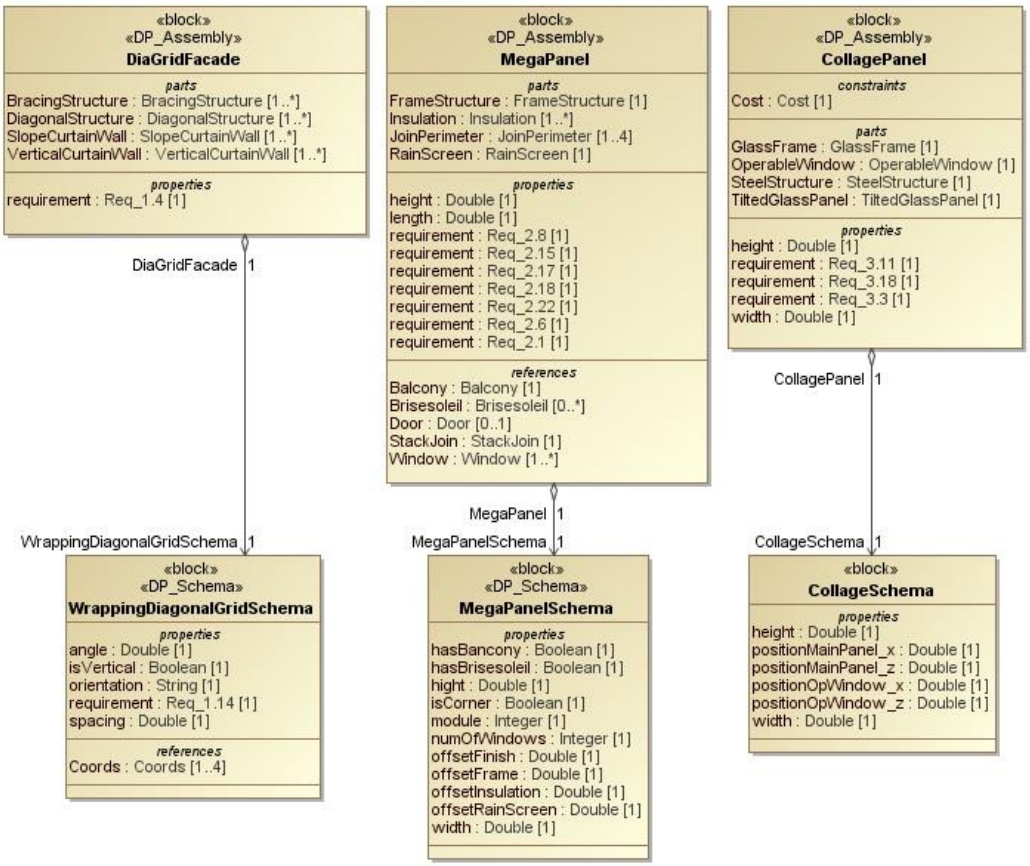


Figure 5.16. Design schema block for every case study

The *properties* and *reference* fields of every block are defined by the minimum set of entities that drive the schema. The block for the schema of the grid of case study 1, the ***Seattle Public Library***, captures the values in these field parameters to produce a wireframe of parametric auxiliary geometry similar to that in Figure 5.17. The surface is defined by four corner points that determine the border of the façade region. The grid is controlled by the angle of inclination and the distance or interval of the tile. The propagation of the components of the vertical or slope curtain wall do not depend on the schema. On the contrary, it depends on the boolean variable of the curtain wall *Sub-assemblies.*

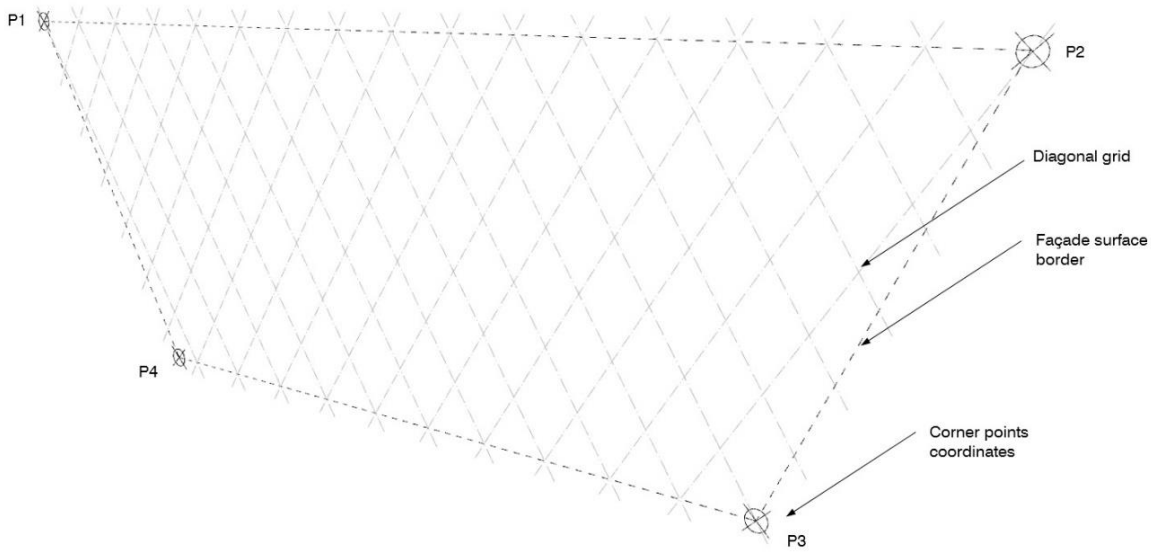


Figure 5.17. Wrapping diagonal grid schema

The block for the schema of Case study 2, ***Via Verde***, captures several optional features with Boolean properties that determine if the mega-panel is in the corner of the building or not or if it has additional features such as balconies or brise-soleils. Besides the dimensions of the mega-panel, this block also has parameters in the *property* fields that control several reference planes for the interior finish of the panel, the end of the structure, the insulation, and the external rain screen. In addition, the modulation of the

panel determines the layout of the studs of the Vierendeel beam that goes along the edge of the slab, and the area below, where windows of different sizes and frames (also made of studs) hanging from the Vierendeel beam are combined in distinct arrangements (Figure 5.18).

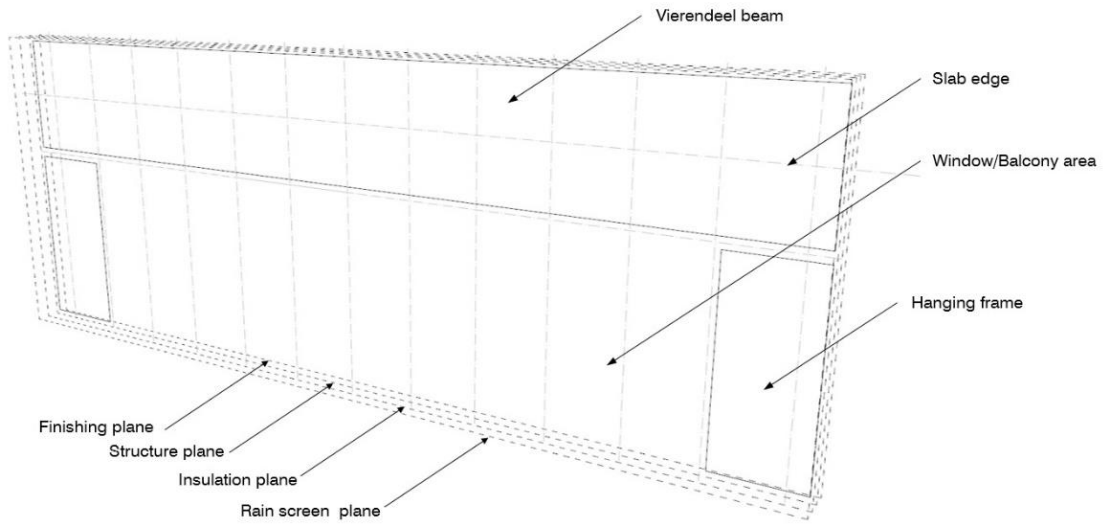


Figure 5.18. Mega-panel schema

The block for the schema of Case study 3, ***100 & 10th Avenue***, uses a one-by-one foot fixed module to organize the collage (Figure 5.19). Besides the dimensions of the panel, the block stores the parameters in the *properties* field for positioning the very first two glass panels that define the rest of the arrangement. The main glass panel is positioned where the view is most appreciated—aligned with the main spaces. The second panel holds the operable windows. New York City code requires that 10% of the façade consist of operable windows. A special glass panel is close to the kitchen areas, where natural ventilation is needed. According to the definition of the position and the size of these two glass panels, the vertical edge of the rectangles that represent such positions extend to the edge of the façade panel, creating the auxiliary geometry for the vertical mullions. Successive subdivisions define the remaining panels. An external design rule defines the tilted angle of the glass panels and the type of glass.

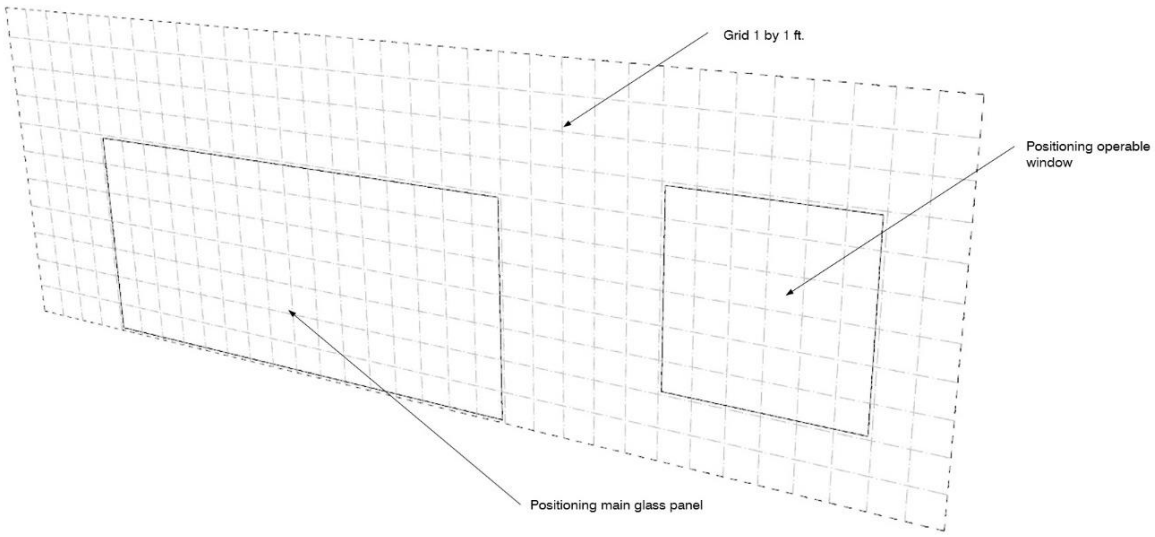


Figure 5.19. Collage schema

Conceptual Structure Blocks

The blocks of *Conceptual Structures* are subsets of *Parts* and *Sub-assemblies* with specific semantics. They gather objects that form abstract constructs determined from various perspectives that traverse the hierarchical structure *Assembly*, *Sub-assembly,* and *Part*. Table 4.14 from Chapter Four shows the impact of such aspects on the definition of the design. Case study 1, due to the structurally challenging nature of the design, emphasizes the aspect of defining the main the role of the *Interstitial Zone* and associated *Conceptual Structures* that address shear forces and wind loads. Case study 2 emphasizes aspects such as tolerances, material, energy, waterproofing, and installation. All of them affect the definition of the perimeter of the panel, which houses all of the joints and sealing. Case study 3 exhibits a concentration of cross references in its aesthetic and structural aspects that match only the conceptual structure used by the expert designer to explain the designs: The *Metal Mesh* merges the panel structure and the frames of the glass panel into a single object (Figure 5.20).

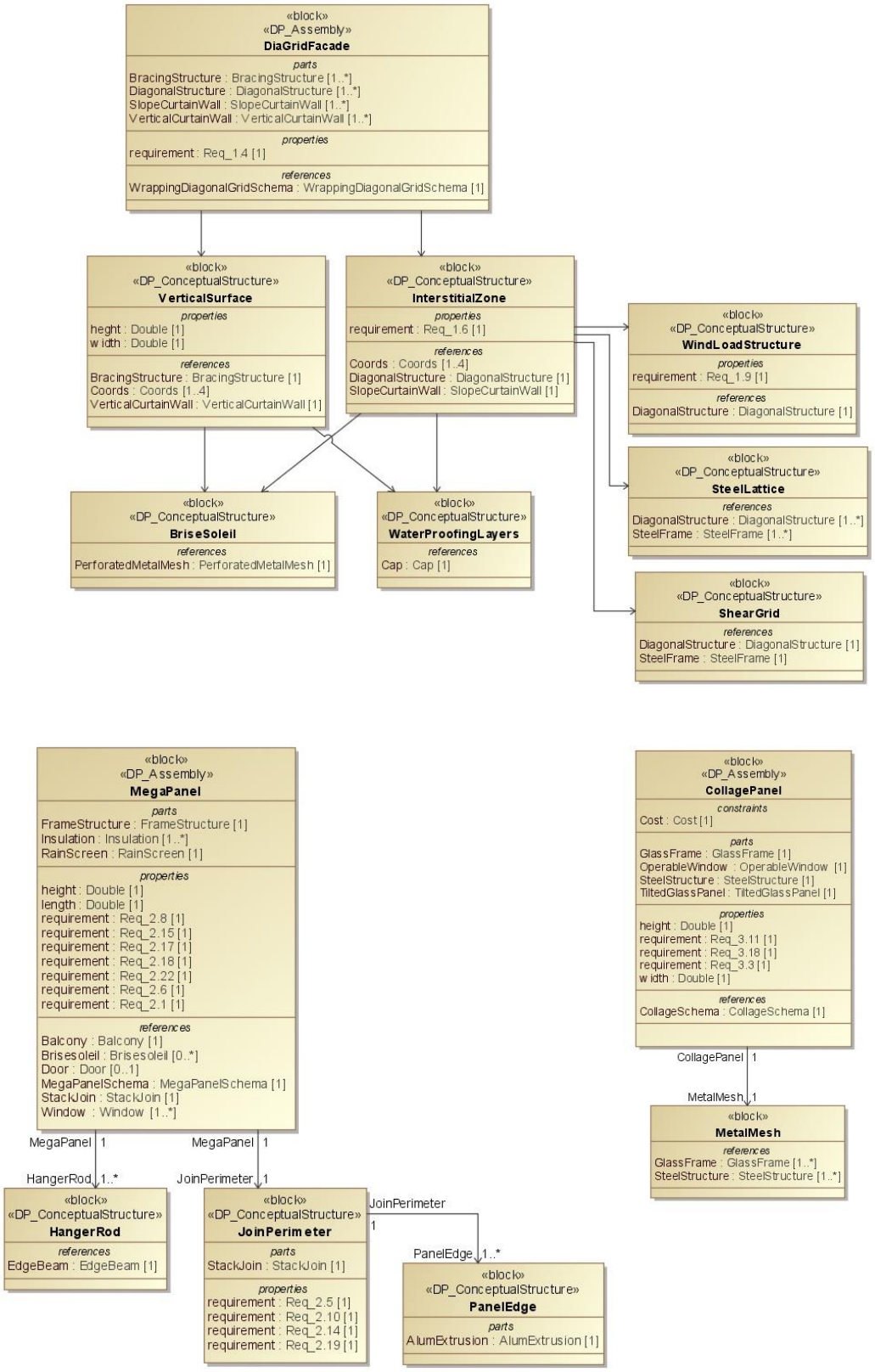


Figure 5.20. Associations of Conceptual structures and the internal composition for each case study

Conceptual blocks are linked to the general *Assemblies* of the three designs through directed associations, represented by a one- directional arrow. In SysML, this link is considered a weak association, since it does not affect the internal structure of the parent block. The selection of this association represents parallel conceptual blocks that overlap Sub-assemblies and Parts blocks by adding redundant juxtaposed groups. The difference is that they describe the design from specific perspectives, and they are named according to the role they play determined by a recognizable feature. In addition, although they are a comprehensive and synthetic way of labeling and providing meaning to groups of components, they do not describe the entire Assembly. The provided descriptions do not include a complete list of objects, probably because the expert designer did not have to provide them. These blocks, however, are types of resources that highlight a relevant aspect of the design. Nevertheless, case study 1, the ***Seattle Public Library***, provides a more accurate, or more complete example of an entire façade system. If we carefully read the diagram from the above Figure 5.20, based on the conceptual structures, it states that the diagrid façade can be understood as *Vertical Surfaces* and *Interstitial Zones* that act as *Brise-soleils* and *Waterproofing Layers.* The *Interstitial Zone* is a *Steel Lattice* that plays an important role as a *Wind Load* and *Shear Grid Structure.* This description, based on the existing blocks, seems to address an important aspect of the designs. However, the major impact is that it implicitly includes all of the *Sub-assemblies* and *Parts* distilled from the case study.

Constraint Blocks

These types of blocks contain restrictions to the design. Their anatomy is made of *constraints* and *parameter* fields. While the *constraint* field contains the expression or the formulation of the restrictions within curly brackets, the other declares the parameters of such a formulation. This block, by adding the SysML *ConstraintBlock* stereotype, also represents an extension of the UML class. However, its definition requires additional steps (Table 5.4). After the proper stereotype is applied, both fields

must be implemented, which entails adding the parameters of the stereotype The first step is to add a *Port* object to the block and then apply the constraint parameter stereotype to the *Port*. The Port object is a property of a classifier that specifies a point of interaction between the block and the environment. The parameter stereotype is retrieved from the Magic Draw profile for SysML customization, and then the parameters are defined. In terms of the *constraints field*, a UML Constraint is first created and assigned the Constraint Block. The actual type of constraint can be an Expression, a Boolean, or an opaque expression. This diversity, which ensures sufficient flexibility to the declaration of the contracts, addresses the scope of the definitions distilled from the case studies. Some of the constraints are explicitly declared and others require interpretation, or their definitions remain open to interpretation. In such cases, the opaque expression allows the addition of textual descriptions for human interpretation rather than computer-readable Expressions.

Table 5.4. Method of creating the constraint block

Methods from the *MetaModel* class	Comments
Profile MDprofile = load((Profile) MD_Customization_for_SysML. additional_stereotypes.profile)	*Load the Magic Draw profile for SysML customization*
applyProfile(Metamodel metamodel, Profile MDprofile)	*applying profile*
Stereotype parameterST = MDprofile.getOwnedStereotype(String ParameterConstraint)	*Retrieving the parameter stereotype from the Magic Draw profile*
Stereotype constraintBlockST = sysMLProfile.getOwnedStereotype(String ConstraintBlock)	*Retrieving the constraint block stereotype from the SysML profile*
Class constraint = createClass(Metamodel metamodel, String constraintName, IsAbstract, constraintBlockST)	*Crating the constraint object*
Port port = createConstraintParameterPort(Type type, String name, Stereotype parameterST, Class constraint)	*Converting the port into parameter and adding it to the constraint object*
addConstraintExpression(String name, Expression formula, Type type, Class constraint)	*Creates the UML constraint and adds the expression to them*
addConstraintOpaqueExpression(String name, String text, Type type, Class constraint)	*Adding an alternative text description to the constriants*

The recorded constraints, which are recalled and concentrated in particular aspects of the description of the design, do not include the entire scope of the designs. On the contrary, they appear only to highlight what is relevant to an understanding of the key issues of the case studies. Although Table 4.10, Table 4.11, and Table 4.12 from Chapter Four collect the identified constraints, they need rationalization for interpretation. Case study 1, the Seattle Public Library, focuses constraints on the restrictions of the diamond glass panel. The constraints regarding re-radiation of solar heat and the openings of the perforated metal mesh lead to the constraint of the solar heat gain coefficient (SHGC) of glass panels. The SHGC, indicating the amount of solar heat that penetrates the panel, ranges from zero to one (Szokolay, 2008). Because of the low-E coating glass and the existence of the perforated metal mesh inside the panel, this value decreases. Glass panels without metal mesh have an acceptable SHGC of 0.3, and with the mesh, it can be as low as 0.17. The formalization of the constraint corresponds to a “less than” expression that can be evaluated as true or false (Figure 5.21).

The diagram also shows the thermal expansion block that affects the metal mesh. Such a constraint is also shared with the stack joint of case study 2, ***Via Verde,*** whose structure is defined by a percentage value that must be between the upper and lower bound parameters. Every case defines a distinct value for the same generic constraint block. Case study 2 also has a waterproof constraint applied to the gaskets according to which water pressure must be less than 8 pounds per sq ft., and Boolean fire proofing attached to the insulation. In addition to this its rain screen has a humidity constraint that corresponds to what is referred to as a SysML opaque expression type. This expression is used to capture ambiguous definitions. In this particular case, it is a text-based warning for human interpretation. Such an ambiguity can be further rationalized by identifying the parameters that participate. The remaining constraints of case study 2, 100 & 10th Avenue, appear in Figure 5.22.

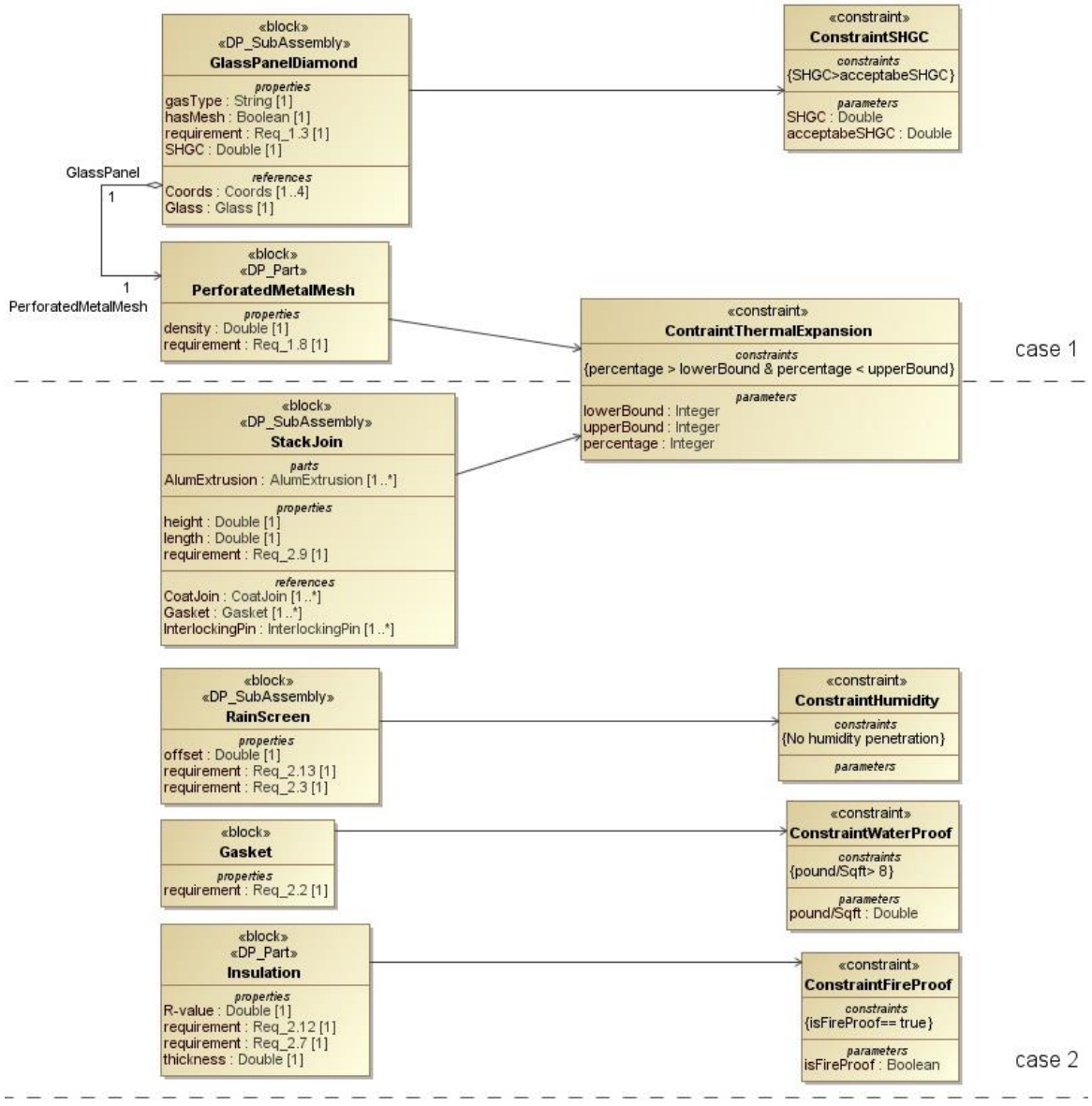


Figure 5.21. Shared and exclusive constraints of case studies 1 and 2

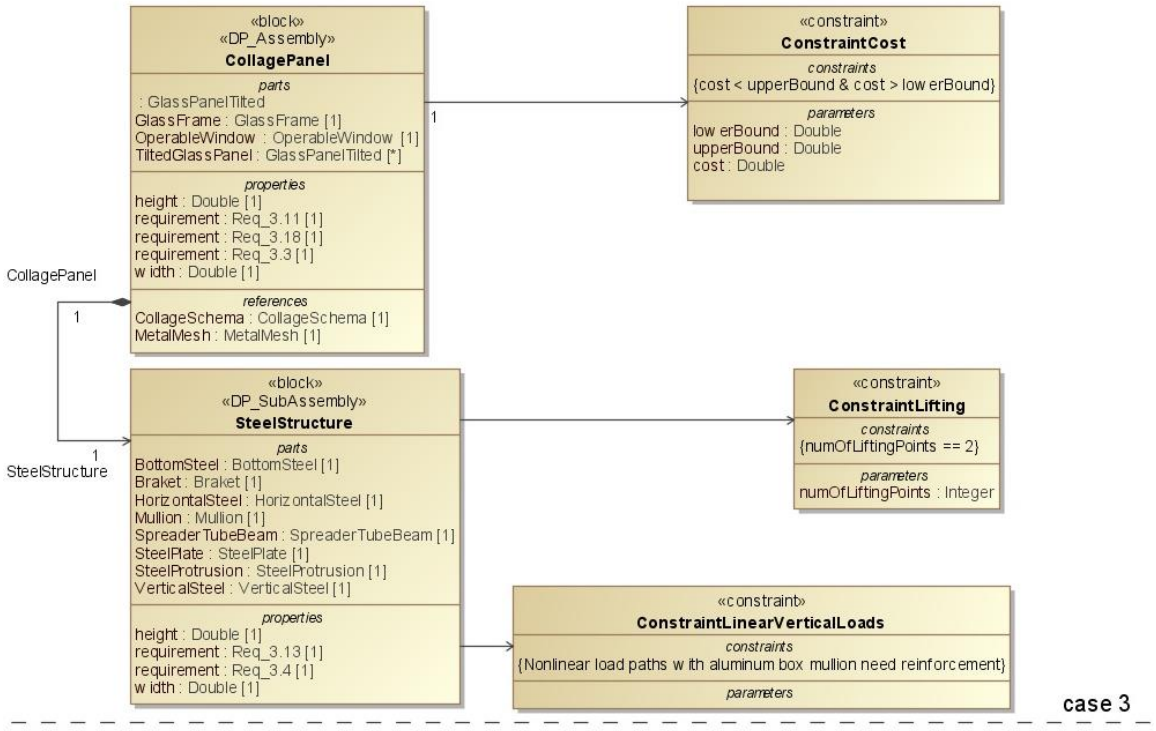


Figure 5.22. Constraints of case study 3

Requirements

Before this research discusses requirements, the difference between requirements and constraints must be declared. While requirements are an expression of a desired goal, constraints represent the boundaries within the requirements that must be satisfied. From this perspective, the requirements themselves are open-ended declarations that require interpretation.

SysML provides formal specifications of requirements by attaching a text-based declaration to blocks that must satisfy them. Although the objects of requirements are not available in the UML, they are an extension of SysML. Table 5.5 lists the steps to converting a UML generic class into a SysML requirement by applying the proper stereotypes. The table summarizes the sub-methods used to implement the general method of creating the requirements of the MetaModel class.

Table 5.5. Sequence of the method of creating requirements of the MetaModel class

Methods for the requirements of the *MetaModel* class	Comments
Stereotype sysmlRequiermentST = sysmlProfile.getOwnedStereotype(String "Requirement")	*Retrieving the Requirement stereotype from the SysML profile*
Class requirement = createClass(Metamodel metamodel, String requirementName, IsAbstract)	*Crating the Requirement object*
applyStereoType(Class requirement, Stereotype sysmlRequiermentST)	*Applying the SysML requirement Stereotype*
requirement.setValue(Stereotype sysmlRequiermentST, String propertyName, Object newValue)	*Setting values for property "Id" and "Text" with the description*
createAssociation(Class class, Class requirement)	*Attaching the Object that must satisfy the requirement*

Studies in the Engineering Design (Pahl et al., 1996) and MBSE (Friedenthal et al., 2011) literature assert that requirements constitute hierarchical structures with general declarations at the top that are gradually specified at the bottom. However, case studies in architectural design from Figure 4.13, Figure 4.14 and Figure 4.15 in Chapter Four shows that the requirements are attached to particular components rather that organized in a top-down structure. Furthermore, besides differences between the classifications of target oriented and failure preventive, the requirements range from determined to under-determined. While in some cases, the determined can be expressed in computer readable formats and implemented through the formalization of constraints, most of the requirements require human interpretation. This incompleteness or ambiguity is a fundamental feature of design problems that can probably be refined in the meta-model through time. In this regard, SysML complements the formulation of constraints with the integration of requirements as text-based properties attached to the blocks. Figure 5.23 deploys the requirements from the major. Assemblies. These examples address a variety of situations that acknowledge the variety of requirements from a qualitative point of view.

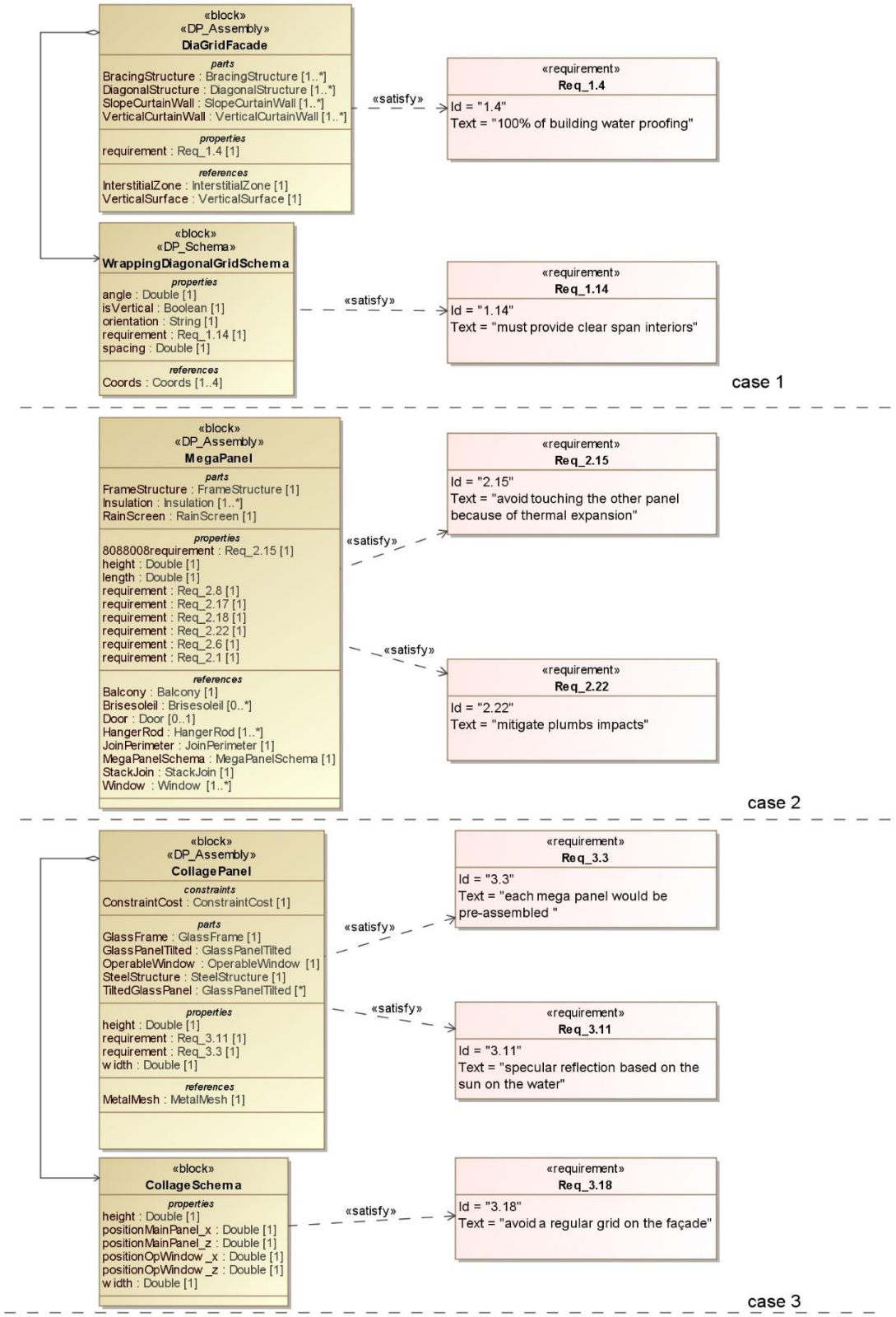


Figure 5.23. Associations of requirements

The main block of the diagrid façade from case study 1 was originally associated with the under-determined requirement 1.14, which demands clear span interiors. However, the diagonal grid schema aggregated to the major Assembly is the block that actually satisfies such a requirement by determining that most of the loads go through the steel structure. By contrast, the determined waterproofing requirement 1.4 does not require interpretation and establishes a clear goal that can be implemented as a constraint.

Case study 2 shows two examples of the opposite category of failure preventive definitions. Determined requirement 2.15 clearly constrains thermal expansion by requiring that contiguous panels not touch, which impacts the percentage of tolerance. Un-determined requirement 2.22 necessitates human interpretation and assessment since it requests a strategy for avoiding the impact of plumbing rather than a specific quantitative and computable goal.

As in case study 1, in case study 3, the original un-determined requirement 3.18 of avoiding a regular grid was assigned to the prefab panel even though the schema is the block that actually satisfies it. The outcome of this open-ended requirement certainly requires human assessment. The remaining requirements, determined 3.11, which requires pre-assembly, and un-determined 3.11, which calls for materializing the analogy of the sun reflected on the surface of the water, also requires human interpretation even though one of the requirements is determined.

In summary, although some requirements lead to the definition of computable constraints, the association with the blocks is a sort of redundant warning mechanism that facilitates human interpretation and assessment.

Parametric Models

Parametric modeling allows a range of geometric variations within the boundaries of its own constraints that represent a design space of possible geometric variations. It has also proven to capture best design practices into parametric relationships, constraints, or even functions. It can embed knowledge into *Parts* and their *Assemblies*. Such knowledge can be classified into object-related knowledge (ORK) and assembly-related knowledge (ARK). While the first describes the features of a singular object or part, the second describes the relationships among the various *Parts* within a parametric *Assembly*. This distinction indicates that a repository of parametric models is intended to capture parts that contains ORK and modules that contain ARK.

For the purpose of this research, parametric models of the collection of *Parts* and *Sub-assemblies* from the three case studies are created using the Digital Project parametric modeling tool. Parts are separated into files that are integrated into one product file for every new Assembly. The level of detail of the models is determined by the descriptions from the verbal analyses in Chapter Four. While some parts are highly detailed, such as those of case study 1 (Figure 5.24), others have only partial descriptions. Even though some of the objects are sub-assemblies, they are also treated as plug-in objects that describe only their connection to the general assembly while hiding their internal complexity, and the knowledge embedded in these tri-dimensional parametric models remains tacit such as that of the brise-soleils of case study 2 (Figure 5.25). Finally, *Sub-assemblies* of parts such as the structure frame of case study 3 (Figure 5.26) are files comprised of files of singular parts. A *Stereotype,* in the Digital Project profile class, with the “filepath” attribute interfaces between every block and the parametric model of the *Part* of the *Sub-assembly*

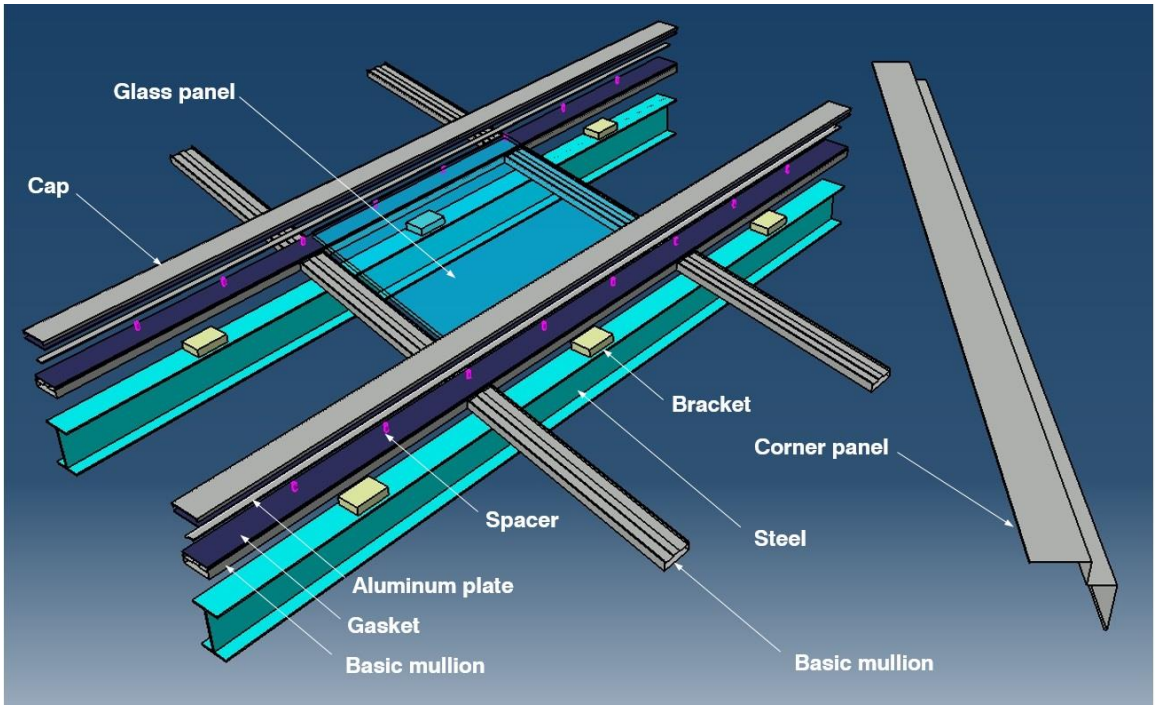


Figure 5.24. Parametric models of parts and sub-assemblies of the Diagrid Facade

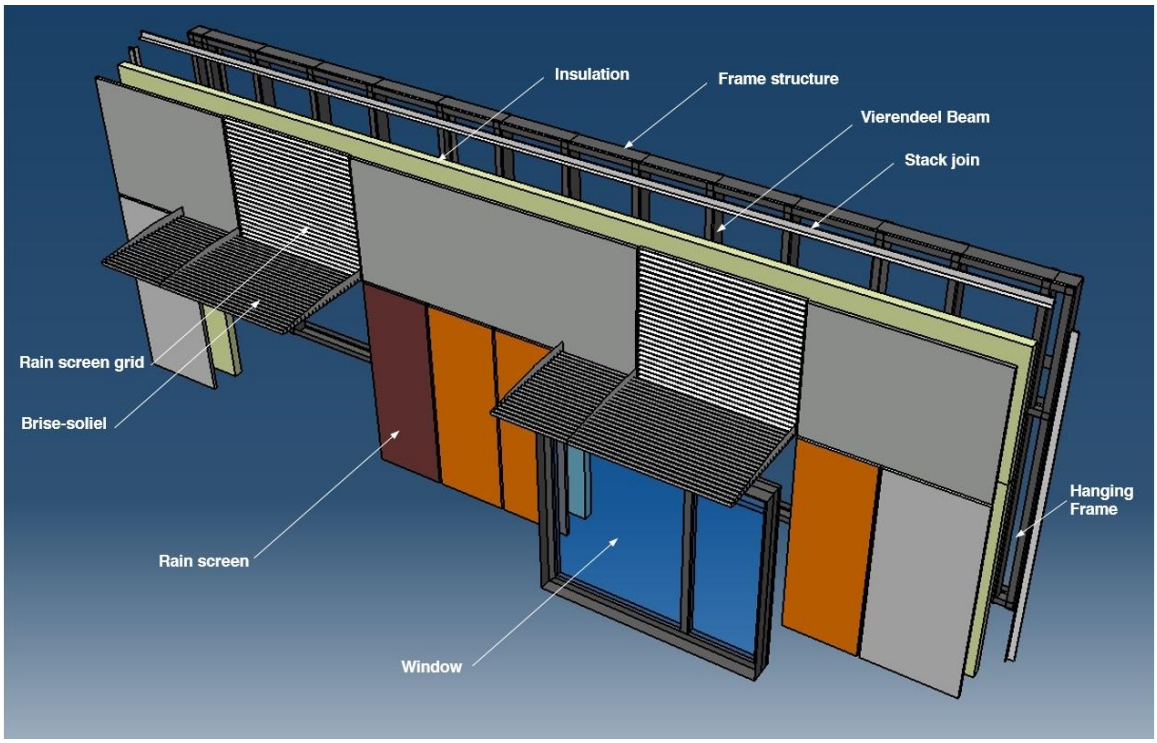


Figure 5.25. Parametric models of parts and sub-assemblies of the Mega- Panel

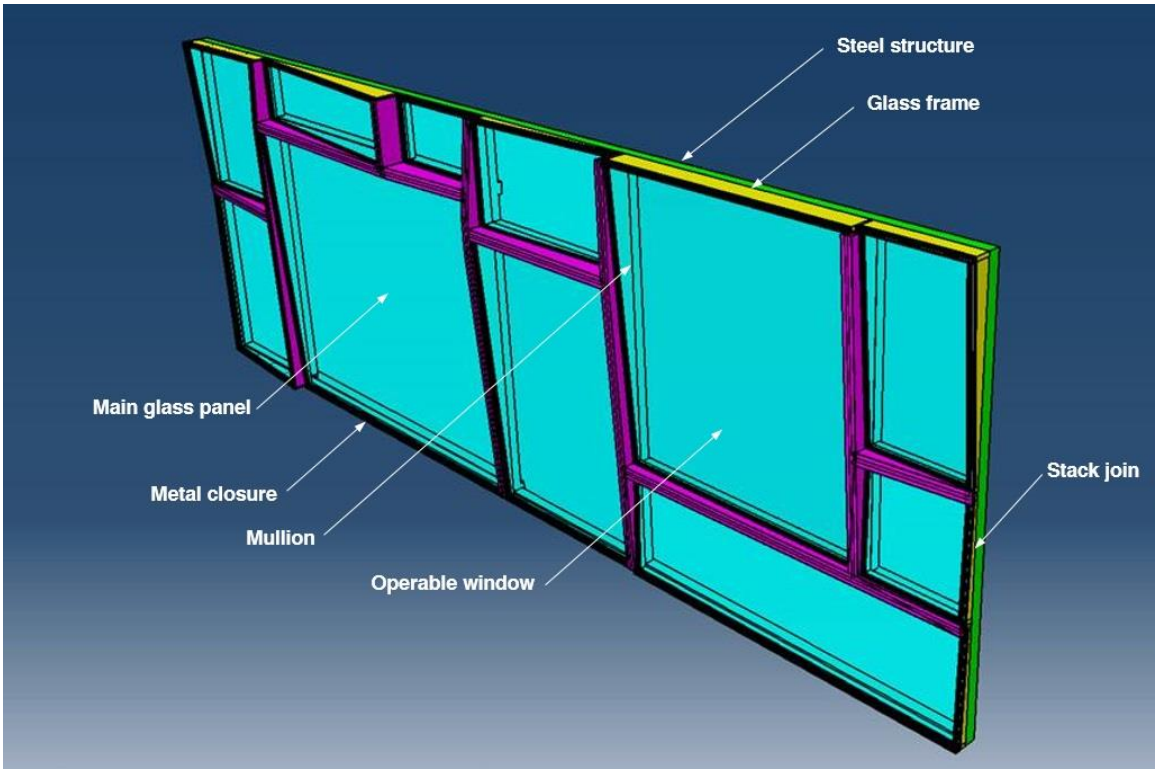


Figure 5.26. Parametric models of parts and sub-assemblies of the Collage Panel (Courtesy of © Marc Simmons, 2015)

Emergent Framework for System Architecture

Through the process of building the meta-model, the identification of commonalities across the case studies leads to a generalization pertaining to a series of constructs of the domain. These generalizations are the fundamental entities that constitute the framework from which the domain can be continuously extended. In terms of implementation, we need to introduce the SysML domain block Stereotype, which labels the object representing the domain, and the generalization method of the MetaModel class (Table 5.6). These generalizations are graphically represented in Figure 5.27 as arrows ending in a white triangle that indicate that the bottom block is a specialized block of the upper block.

Table 5.6. Domain Stereotype and generalization methods required to build the framework

Methods of the *MetaModel* class	Comments
Stereotype sysmlDomainST = sysmlProfile.getOwnedStereotype(String "Domain")	*Retrieving the Domain stereotype from the SysML profile*
applyStereoType(Class domain, Stereotype sysmlDomainST)	*Applying the SysML domain Stereotype*
createGeneralization(Class generalClass, Class specilizedClass)	*Crating the Generalization association*

The diagram shows the Custom Façade System domain block at the top disaggregated in two main sections*:* General and the Specific design domain frameworks. While the general framework section captures and represents in very abstract terms the relationships across objects that represent the structural aspects of design practice, the specific one distills the common objects that embed particular knowledge in the custom façade design.

The general design domain framework section states that the domain has *Physical Components* and *Patterns of Organization*. These components are *Parts* or *Assemblies* of *Assemblies* that can also belong to *Conceptual Structures*, which gather them from various perspectives. The *Schemas* share the *Wireframe* with the *Assembly* driven by the reutilization of distilled *Design Rules* that provides all the necessary auxiliary geometry to coherently build the arrangement of *Parts* and *Sub-assemblies.*

The Specifics section groups typical building Parts, *Sub-assemblies* of complex prefabricated objects, typical types of *Assemblies* such as curtain walls of *Prefab Panels*, and finally a collection of *Schemas.* Any custom non-standard object can be extended from the General framework. Regarding constraints and requirements, although the MBSE engineering literature encourages building hierarchical specialized structures, the case studies provide evidence of only discrete linkages to specific blocks.

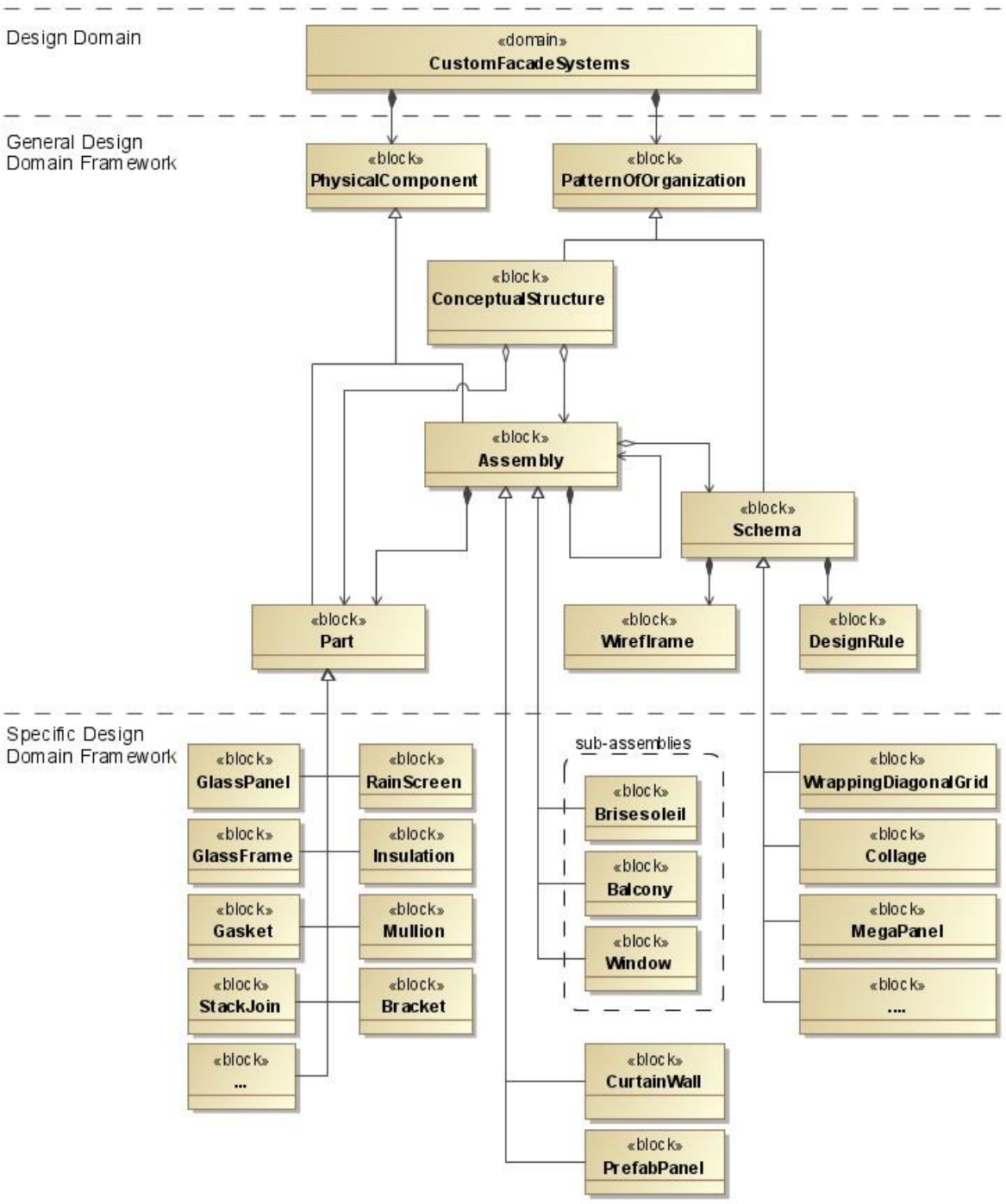


Figure 5.27. Domain framework

5.4. Discussion: Building a Meta-model

The process of building the meta-model based on the three case studies raises questions regarding the degree of completeness of the meta-model, the level of detail, and the adoption of industry standards for descriptions of parts that will be further combined in assemblies reusing design knowledge.

Incompleteness and Sufficiency of the Meta-model

Capturing and modeling design knowledge is a process of continuous growth. From the perspective of the description of the units of knowledge distilled from the case studies in the previous chapter, the process is clearly incomplete. While some parts are described with precision, the internal structure of many sub-assemblies remains hidden, and thus, incomplete. However, from the perspective of the identification of the role of the objects and their cross associations, the descriptions provided by the expert are sufficient for the construction of the meta-model. Since the meta-model focuses on the representation of integral features of the aspect of interest rather than carrying out an exhaustive decomposition, it satisfies the notion of completeness introduced early in this chapter.

The efficiency of the mechanisms of recalling tacit knowledge relies on blocks representing sub-assemblies that contain relationships regarding how to build arrangements of parts. These blocks hide such information from the meta-model, keep that knowledge in the parametric models of parts and only expose the necessary information to build the meta-model.

The Level of Detail

The level of detail partially depends on not only the availability of detailed descriptions but also the application of stereotypes. The continuous addition of descriptions of objects can extend the level of detail. However, the application of stereotypes is what determines the level of detail of the recall. These stereotypes can be

applied at any level on the hierarchy of physical components since they can map blocks with files of subassemblies while mapping other blocks with the files of the parts that constitute them. This redundancy fosters enough flexibility to execute the recall of components at variable levels of detail. The higher the level of application of the stereotype, the higher the amount of tacit knowledge that the block contains. The lower the level, the less tacit knowledge, but the greater the flexibility for recombining the parts. The challenge during instantiation of the objects is to determine the level of recall: either Sub-assemblies that reduce flexibility in terms of creating new configurations but include tacit knowledge, or Parts that have more flexibility but that are more demanding in terms of assembling related knowledge.

Standardization of the Architecture of the Specific Domain Framework

The framework of the design domain, which provides fundamental objects from which to extend the domain, is based on the integration of common modeling constructs and general categories. The differentiation of the objects of the general design domain frameworks from those that belong to a specific domain of the custom façade design, also proposes extensions in both branches. Although the definition of the specific domain components in this research is based exclusively on the three case studies, ongoing efforts to define domain-specific objects such as that of the bSDD ("buildingSMART Data Dictionary," 2016) already exist. These initiatives build libraries with definitions of objects, their attributes, and relationships in an effort to standardize the definition of common building objects, which facilitates their exchange and eliminates ambiguity. These object definitions are available to not only designers but also software developers. Adopting industry standards that complement the library of parts at the bottom of the specific domain framework can contribute to building assemblies driven by design knowledge based on existing extensible and exchangeable objects.

CHAPTER 6

CONFIGURATION SPECIFICATION AND GEOMETRIC REPRESENTATION

Overview

This chapter addresses the challenge of extending the scope of the generation of design alternatives in early design stages by manipulating the design schemas. The aim of the proposed generation process is to augment the capabilities of expert designers in the production of parallel solutions early on. In this regard, the working hypothesis is that by separating the specification of the configuration and its geometric representation, the degree of freedom to specify and produce various configurations increases and the design space expands. The following sections discuss the process of specifying instances from the meta-model according to the patterns of organization, the specification of the attributes and associations of the instances, and the interpretation of the specification in order to produce the geometric representation of the resulting design configurations (Figure 6.1).

To test and validate the process, this research uses instance specification and geometric representation techniques to reproduce the case studies, to expand the range of potential configurations, and to explore hybridizations by combining parts from a variety of sources. Although the outcome of the process is an automatic response that produces a human editable parametric model, it is only as a preliminary design rather than a final solution. The implications in terms of parametric modeling for reutilization and potential areas of applications will be also discussed.

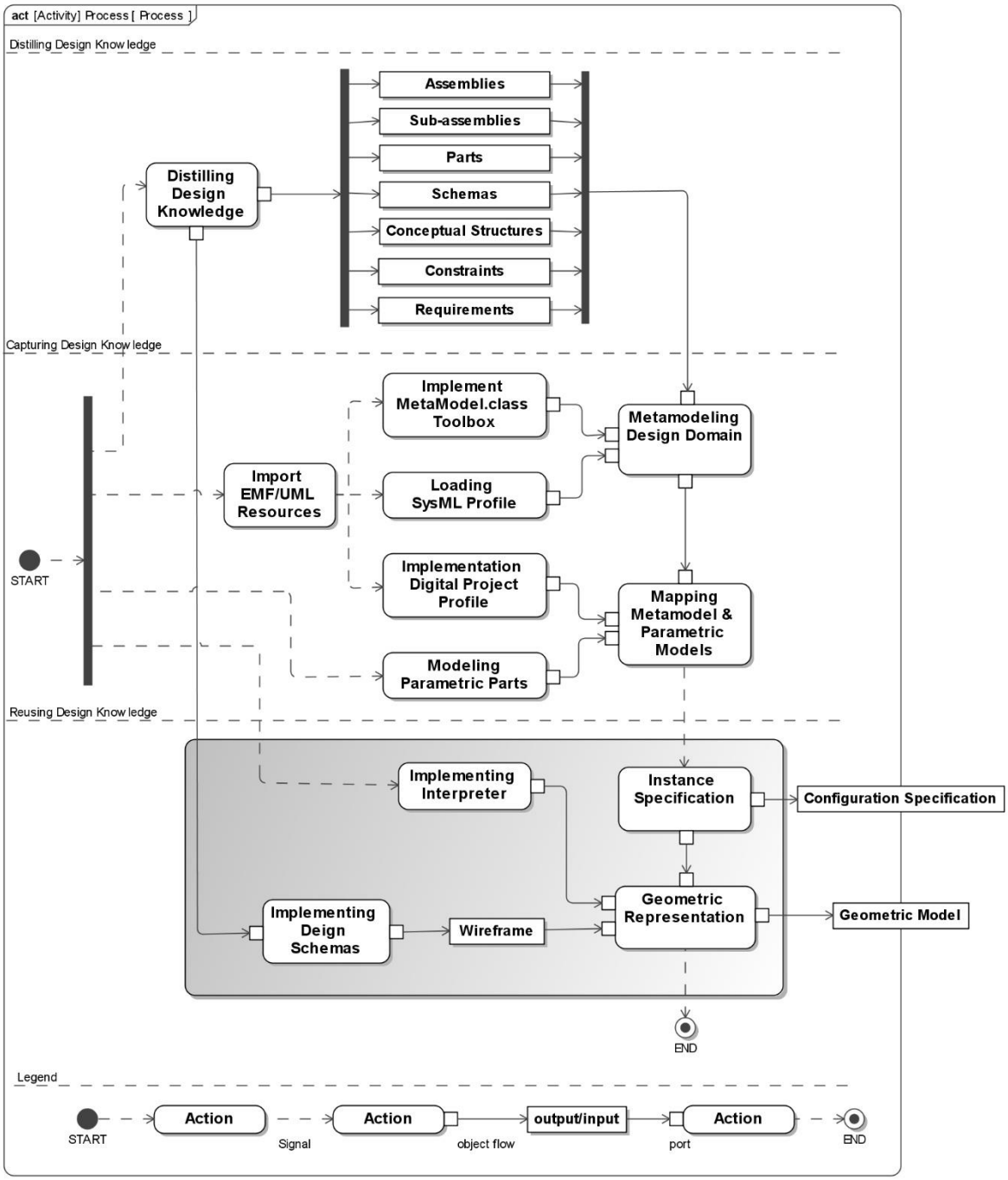


Figure 6.1, Configuration Specification and Geometric Representation in the context of the overall process

6.1. Approaches to Specification and Representation

The terms “instance specification” (IS) and “geometric representation” (GR) belong to two distinct domains. While the first describes the process of creating an instance object from a SysML block, the second corresponds to the execution of a process that leads to the production of a tri-dimensional geometric model (GM) that uses either a CAD or BIM tool. These two components of the process are based on two main approaches: separating them to avoid the complexity of the CAD or BIM models, and using Wireframes, described by Simmons (Simmons, 2013), that is, the translation and implementation of the Design Schema into a tri-dimensional model of auxiliary geometry that embeds such a pattern of organization.

Separation of Configuration Specification from Geometric Representation

The flexibility for variations in the configuration is the product of the separation of the design specification from the geometrical representation. This autonomy provides the flexibility of specifying the configurations of solutions according to the shifting nature of design problems and avoiding the complexity of editing the binary trees or topological relationships of the geometric models. Current parametric modeling techniques generate variation by modifying the parameters of geometric models by reducing the solution space so that it fits within the scope of the parametric structures. Although parametric models facilitate the geometric representation and manipulation of knowledge, they have limitations regarding the scope of knowledge that they can embed and their capabilities of generating variations beyond the limits of the hierarchical structure of the parametric relationships within the assembly of parts, prematurely limiting the generation of diverse possible candidate solutions. In addition, in the field of architecture, PM is strongly tool dependent, since the sharing of parametric features by a number of tools is still an open research area (Eastman et al., 2010; Tarandi & Froese, 2002; Venugopal et al., 2012). Because of this limitation, storing knowledge in specific file formats that mix design conceptuality with tool

internals reduces the scope of the reusability of the models. However, the mechanism that specifies the configuration by combining objects from the design domain, building association among objects, and defining their attributes, provides a wide range of variations since it not only creates various configurations from the objects but also parametric variations. All of these candidate solutions support further parametric changes for refinement. Another implication of separating the specification from the representation is that it facilitates preliminary estimations by accessing the data of the models even before the geometric tri-dimensional representation takes place.

Wireframes

The CAD field has used the term "*wireframe"* to refer to techniques for the tri-dimensional representation of solid object input geometry to extract the topology (Agarwal & Waggenspack, 1992), to create solids (Lequette, 1988; Woodwark, 1986), and to visualize B-Rep models (Requicha & Rossignac, 1992). Even though *wireframe,* in this research, is based on the concept of the tri-dimensional representation of vertices, edges, and faces of a solid object, it corresponds to a wider interpretation from a design perspective. It is influenced by Simmons's extension of the term to a tri-dimensional model made of points, lines, and reference planes that represent auxiliary geometry embedding the design intent. Such geometry is the input that propagates models of the physical parts of a design.

Complex models use auxiliary geometry as a reference to define the start and end points of linear objects, reference planes, sections, and paths for all kinds of extrusions and sweep operations. This auxiliary geometry in advanced parametric modeling tools based on binary-tree hierarchical structures corresponds to nodes at the top of the hierarchy that are the input geometry for the nodes below them. Using the *wireframe* to implement patterns encrypted on the *Design Schema* block provides coherence to the configuration or the arrangement of parts.

6.2. Methodology for Configuration Specification and Geometric Representation

The process, which consists of the creation of an assembly of parts from a meta-model, involves the selection of well-defined, normalized, interchangeable objects (Bielak, 1993) that create an assembly according to patterns of organization. The use of the term “normalized” represents an attempt to define the boundaries of the standardization of descriptions of objects of the domain meta-model. While standardization implies compatibility at the industry level, normalization corresponds to compatibility among the sample objects of this research.

Configurations can be produced from two perspectives: implementation and execution. Implementation refers to the development of a required infrastructure that supports execution of a design. Complementing the abstract specification of the *Design Schemas* are two elements that must be implemented: the function that drives the production of the Wireframe and the Protocol for creating instances from the meta-model. In addition, the production of geometric models requires the implementation of the Interpreter, that is, a collection of generic commands that control the parametric tool.

From the execution perspective, the *Configuration Specification*, also based on the *Design Schema* that supplies the main inputs, creates a series of instances from the blocks of the meta-model. Since the blocks describing the physical components have applied stereotypes, the instances have path file pointers that designate the repository of actual parts. This collection of instances and their attributes are the inputs for the *Geometric Representation* that uses the methods of the *Interpreter* to produce the actual *Geometric Model* (GM) by recalling the parametric models of the parts and setting the values of their parameters. The results of this process (Figure 6.2) are two parallel and consistent representations*:* The SysML model of the *Configuration Specification* and the parametric tool file of the *Geometric Model*. While the specification describes the

design in general domain-specific terminology, the *Geometric Model* addresses the complexity of tri-dimensional parametric modeling. It also facilitates visual evaluation and provides a template for further manual editing, if necessary. In summary, the solution gradually gains resolution from the high level of the abstract conceptual specification of the configuration to the hierarchical structure of the GM, passing through the *stereotypes* interfacing both. The following section discusses the entire process in detail.

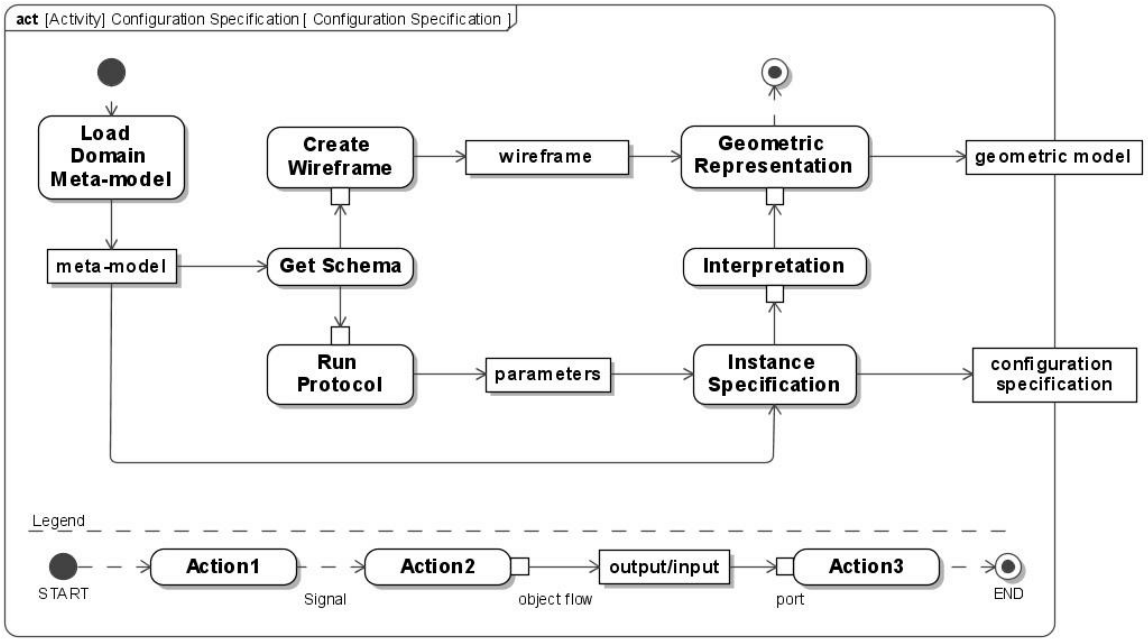


Figure 6.2. Process of configuration specification and geometric representation

Creating Instance Specification from the Meta-model

Creating instances of the blocks of the meta-model of the domain simplifies and facilitates the generation of possible configurations with different topologies, avoiding the limitations of hierarchical relationships GMs. This independence from traditional 3D modeling environments sustains the generality of the process, since the aim is to produce the geometric representation with not only one tool but a variety of tools. Even though generality is a common goal in the modeling culture, the creation of instances should also take into account the capabilities of the tool, indicating that the blocks of

the meta-model must include some key parameters that enable the propagation of components according to the needs of the Interpreter. In other words, although a meta-model is generally abstract, it is not fully independent from the means of representation. The specification must also include normalized parameters that allow compatibility among objects, enabling the creation of coherent parametric structures.

The *MetaModel* class contains all the methods for creating not only the meta-model but also instances from that model (Table 6.1). The mechanics of creating an instance are very straightforward. The general sequence is firs: creating an instance from a block (class) of the model, and then setting the values of the attributes, setting the associations with other instances, and adding the instance to the meta-model.

Table 6.1. Meta-model methods for creating and manipulating instances

Method of Execution from *MetaModel* Class	Comments
createInstance(String name, InstanceSpecification class, Model model)	Creates a new instance from a class or block
createProperty(Class class, String name, Type type)	Create an attribute property for the class of the instance
getAppliedStereotype(Stereotype stereotype);	Getter for the instance stereotype
getValue(Stereotype stereotype, String propertyName);	Getter for the value of the property of the stereotype
setInstacePropertyValue(Property property, String value, InstanceSpecification class)	Set the value for the class property and all the instances
getInstacePropertyValue(InstanceSpecification instanceSpecification, Property prop)	Get the property value for a singular instance
setIstanceAssociation(InstanceSpecification instance1, InstanceSpecification instance2,	Create association between two instances

During the process of creating instances, some of their properties are limited by constraints. While some of these constraints are computer-readable expressions, others are open to interpretation and implemented in the meta-model as opaque expressions. This type of expression, which is a SysML resource for acknowledging ambiguity, registers warnings or recommendations for human interpretation. For example, some computer-readable constraints very specifically define the limits of the

acceptable R-values of a panel or solar heat gain coefficient (SHGC) of the glass, and others such as the constraint to avoid humidity penetration, which lacks of further specification, require interpretation and refinement.

Translating the Design Schema into a Wireframe

The Design Schemas block has two parts, the Wireframe and the rule or rules that create the wireframe. Similar to any *Part* or *Sub-assembly* mapped to SysML Blocks using stereotypes, the Design Schema corresponds to a tri-dimensional external representation or *Wireframe*. This correspondence, or mapping, can be either a recall of a parametric file of auxiliary geometry or the product of the execution of a function. For the purpose of this research, because the schemas are not pre-existing objects, they are implemented as scripts representing the rule executed while reading the *Instance Specification* in order to create the wireframe. These functions can be implemented either in the tools native scripting language or in a tool-independent language such as Java or MATLAB. Since this implementation is built in the Java environment, the class, called *Schema.class*, which contains methods for creating the wireframe, was also implemented in Java. The role of the SysML blocks of the Design Domain meta-model is to capture the attributes that are the inputs to trigger the actual scripts.

Design Schema Rules Protocol

Chapter 4, which focuses on the knowledge-distilling process shows the incidence of the action of applying design rules to pattern of organization and main assemblies Table 4.13. Even though the sample is not conclusive, it exhibits a tendency to link rules with major objects. Although the rules that drive the generation are not explicitly described in the transcriptions, which may be the result of their complexity, partial descriptions combined with the digital documentation of the case studies provide enough information to interpret them. The detailed explanation of the specification of instances derived from these rules were extended in section 6.3 of the case studies.

Implementation of the Interpreter for Geometric Representation

The need for interpretation acknowledges that every tool has a unique way of representing tri-dimensional objects. The task of the Interpreter class, also implemented in the Java environment, is to provide methods of translating the abstract Configuration Specification into software-readable instructions. For that purpose, the Interpreter writes in the native scripting language of the tool in order to be able to deliver instructions to the tool, which executes the geometric representation and creates the actual parametric models of the assembly. While the Instance Specification contains instructions about what needs to be built, the Interpreter tells to the tool how to build it.

Use of scripting languages facilitates the automation of the generation process since they are high-level languages that assume the existence of an interpreter that executes precompiled resources in machine code (Ousterhout, 1998). Interpreters usually use classical parsing techniques (Aho & Ullman, 1972) to identify instructions in the text of the script. Another useful characteristic of scripting languages is that they have typeless syntax that simplifies the redefinition and exchange of variables along the script. Because they execute instructions in a simple and flexible way, they are also called the "glue," or system integration languages, since they can integrate components into large sequences of commands.

The parametric modeling tool used in this research, Digital Project, supports the VBScript scripting language (GehryTechnologies, 2009), based on Visual Basic Language. VBScript is an interpreted language that relies on precompiled Digital Projects functions. It provides functionalities for automating the creation of parametric objects and builds parametric relationships across the assemblies of parts. Digital Project provides a runtime environment and a script editor for creating and executing automation routines, like most CAD and BIM tools. It also supports access to its resources by external applications. This implementation takes advantage of the latter feature and automatically creates and executes the VBScripts from an external Java

application. The implementation of the Interpreter includes generic functions for creating and manipulating Assembly-Related Knowledge. The Interpreter also excludes specific 3D-modeling commands since it is assumed that the parts embed Object-Related Knowledge that remains hidden from the perspective of the assembly. Table 6.2 shows methods for creating the assembly, parts, and parameters, manipulating the parts, and sharing reference geometry and parameters across parts.

Table 6.2. Methods of the Interpreter class

Method of execution from *Interpreter* Class	Comments
startScript(String productName)	Start VBScript file and open DP
startSub()	Start script
createProduct()	Create product assembly file
createProduct(String pathFile)	Or open product assembly file
insertPart(String partName, String pathFile)	Insert Subassemblies or part files
createPart(String partName)	Create part from scratch
addExternalComponents(String partName)	Add part to the assembly product
addIntegerParameter(String partName, String paramName, **int** value)	Add integer parameter to the part
addDoubleParameter(String partName, String paramName, **int** value)	Add double parameter to the part
setIntegerParameterValue(String partName, String parameter, **int** value)	Set integer parameter value
setDoubleParameterValue(String partName, String parameter, **double** value)	Set double parameter value
addExternalParameter(String source, String target, String sourceParam, String targetParam)	Add parameter from another part
addExternalReferencePoints(String source, String target, **int** row, **int** col)	Add reference geometry from another part
instantiateUserFeatureBetweenPoints(String partName, String userFeaturePathFile, String fromPoint, String toPoint)	Create instances from one part into another using two inputs points
instantiateUserFeature4Points (String name, String userFeaturePathFile, **int**[] arrayPoints)	Create instances from one part into another using four inputs points
publishPoints(String partName, **int** row, **int** col)	Making accessible reference geometry for another part
movePart(String partToMove, **double** x, **double** y, **double** z)	Move the part to x, y and z coordinates within the assembly
save(String fileName, String projectFilePath)	Save the part or assembly file
endSub()	End the script

The parts are either retrieved from the repository of parametric objects or created by a function. For the retrieved parts, the Instance Specification and its Stereotype provide the parameter values and the location of the part file. For the created by function objects, the Instance Specification provides the input parameters for executing the function. While the first approach is used for most of the part and sub-assemblies, the second is used for the generation of wireframes based on values provided by the schema blocks. The combination of both types of approaches generates the GM. The following example shows the output for the *insertPart* and *movePart* methods of the *Interpreter* class. They require the name of the part, the location of the files, and the coordinates of the new position as input values. The *Configuration Specification* contains all of these specific values. The methods of the Interpreter return a string value and concatenate all of the strings into a single script of instructions (Figure 6.3).

Example Java methods:

```
String partName = "Vierendeel";
String pathFile = "C:\Repository\Vierendeel.CATPart");
double z = 1714;
Interpreter.insertPart(partName, pathFile);
Interpreter.movePart(partName, 0,0, z);
```

Example VBScript Output:

```
'Insert Existing Vierendeel Part
Set Vierendeel_PartDoc = CATIA.Documents.Open("C:\Repository\Vierendeel.CATPart")
Set Vierendeel_= Vierendeel_PartDoc.Part
Set Vierendeel_ToMove= oRootCol.AddExternalComponent(Vierendeel_PartDoc)

'transformation matrix
       Dim Vierendeel_MoveMatrix(11)
       Vierendeel_MoveMatrix(0) = 1
       Vierendeel_MoveMatrix(1) = 0
       Vierendeel_MoveMatrix(2) = 0
       Vierendeel_MoveMatrix(3) = 0
       Vierendeel_MoveMatrix(4) = 1
       Vierendeel_MoveMatrix(5) = 0
       Vierendeel_MoveMatrix(6) = 0
       Vierendeel_MoveMatrix(7) = 0
       Vierendeel_MoveMatrix(8) =1
       Vierendeel_MoveMatrix(9) = 0
       Vierendeel_MoveMatrix(10) = 0
       Vierendeel_MoveMatrix(11) =z
```

```
'move
       Set move_Vierendeel = Vierendeel_ToMove.Move
       Set move_Vierendeel = move_Vierendeel.MovableObject
       Set move_Vierendeel_Variant = move_Vierendeel
       move_Vierendeel_Variant.Apply Vierendeel_MoveMatrix
'--------------------------------------------------------------------
```

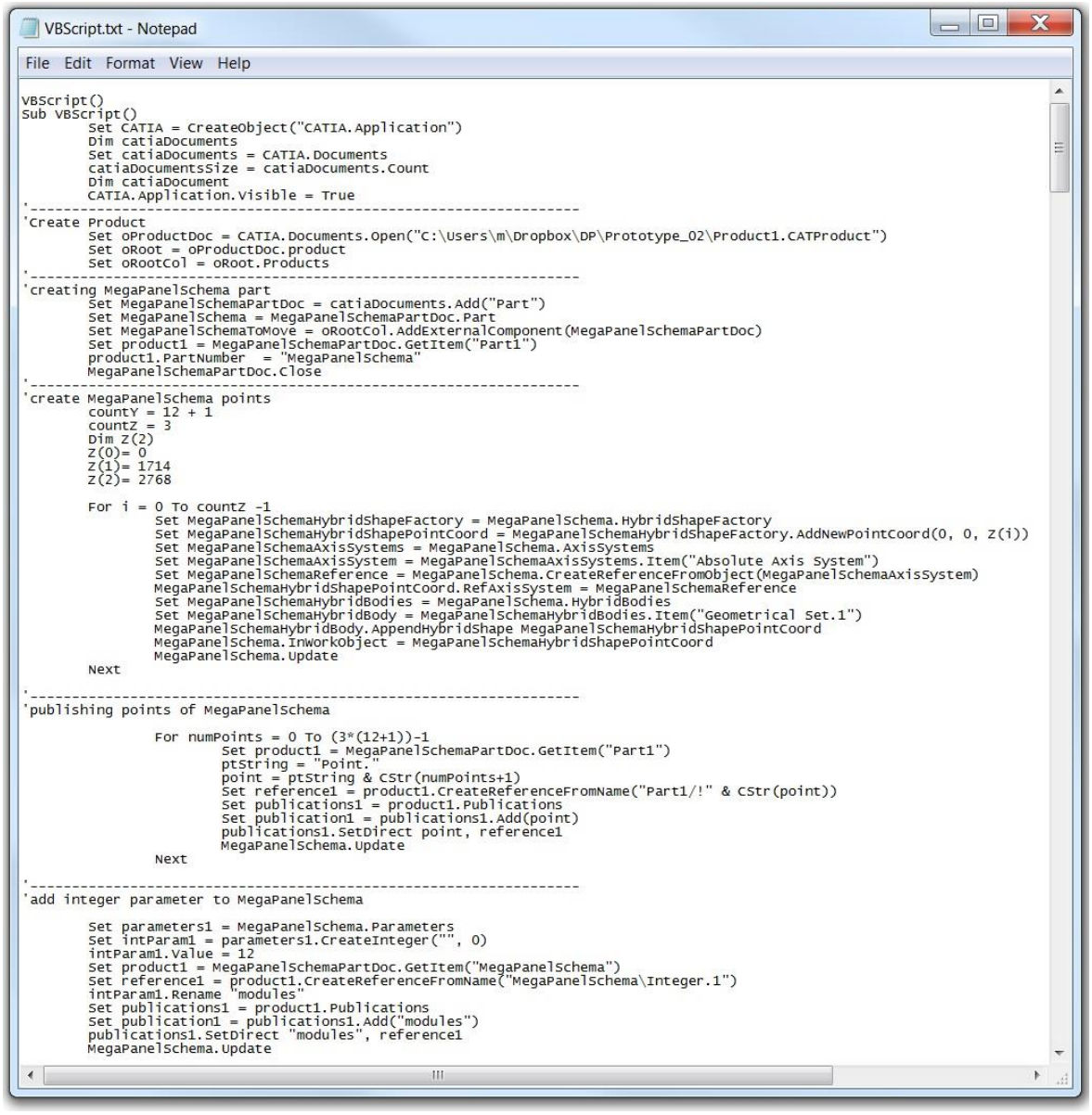


Figure 6.3. Fragment of the resulting VBScript file from the interpretation

6.3. Resulting Configurations

For the purpose of validation and verification of the generative aspect of the proposed meta-modeling process, the methods for specification and representation are applied to the three case studies for reproducing the designs, augmenting the range of alternatives configurations by manipulating the schemas and exploring hybridizations across objects from different case studies.

This section is an exploration of the anatomy of three design schemas. The first one, from the Seattle Public Library, corresponds a simple *grid-based* schema that uses a grid of points from its wireframe as reference geometry to propagate parts. The second, is a *sequence-based* schema from the Via Verde project that use a sequence of numbers to define conditional propagation of object over a grid of reference points. The last one, is a *filling-based* schema that sequentially inserts a predefined list of objects of different sizes until to completely fill a grid of cells provided by its wireframe. Grid, sequence and filling-based approaches, deduced from the descriptions by the expert, combined with the graphical documentation, were explored looking for generalizations of the techniques of propagation of objects to better specify the interface of the required parameters to create the parametric models of the parts. After all, we can make the difference between the parameter that parts or sub-assemblies require for their internal structure, and the parameter that the same parts require for instantiation. The meta-model should able to capture both types of parameters.

Case Study 1: The Grid Schema

Grid-based Wireframe

This first approach is based on a two-dimensional grid as the reference geometry to propagate single parts one by one. The grid of the wireframe, which is generated by recalling a simple function with a two dimensional for loop that creates the points, provides the reference geometry for the insertion process (Figure 6.4).

The attributes of the SysML block of this schema determine if it is vertical curtain wall or not, its orientation, angle, spacing, and the four points that define the corners of the curtain wall. Because the resulting curtain wall is based on inserting parts and not assemblies, this reference geometry must support various intervals for traversing the grid. While the dominant directionality of the façade demands continuous extrusions from top to bottom, the secondary is filled with interlocking small short segments of the same extrusions. The linear objects (i.e. mullions, gaskets, or caps) require two points from the grid for insertion, and the diamond glass four. In summary, every group of parts has its own interval to traverse the grid. Since this parametric tool creates a one dimensional array of points, the general form for propagating linear parts is: from every point “n” in the grid to another point “n + interval”. For four corners parts such as the glass panels, it is for every point “n” find point “n + interval1”, point “n + interval2”, and point “n + interval 3”. In addition, four corner points define the trimmed region of the façade.

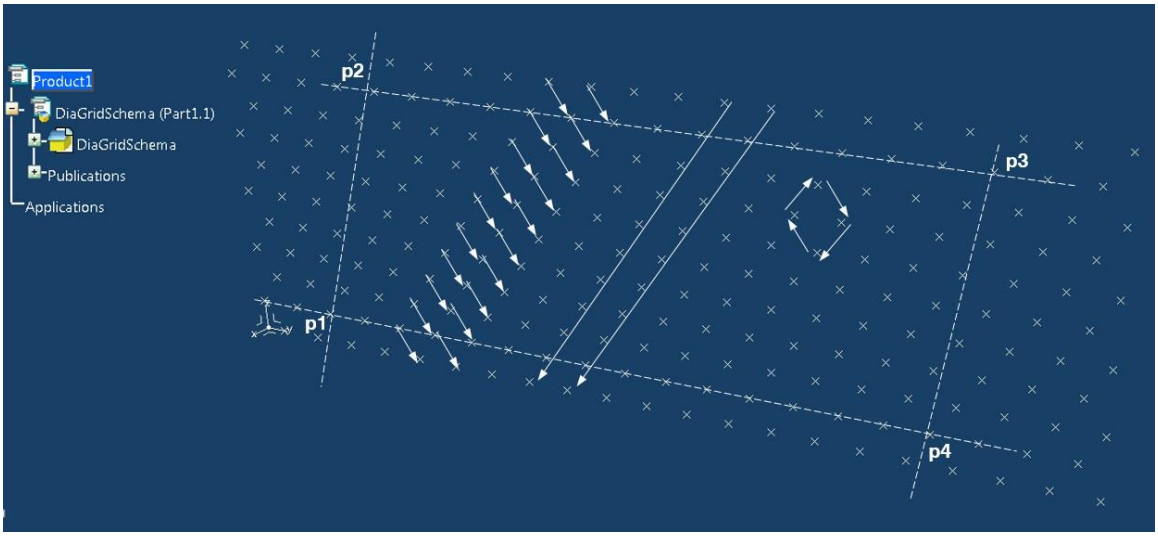


Figure 6.4. Process of insertion of parts across the grid from any point “n”

Protocol of the Configuration Specification based on For Loops

The specification process (Figure 6.5) organizes several rules derived from the interpretation of the transcriptions of the description of the case study by the expert

designer (Table 6.3). It begins with the evaluation of the orientation of the grid of the curtain wall to select among four types of glass panels, three with different metal mesh densities, and one without metal mesh. That grid is a two dimensional array of coordinates that is consistent with the wireframe of actual points. This apparent redundancy allow specifying the instances without the mediation of a geometric model of the wireframe. The process continues evaluating whether the curtain wall is vertical or not. If it is vertical, then it will propagate the I-Shape Million. If it is not, the protocol will instantiate the brackets to support the short span basic mullions. From that point, vertical and slope curtain walls share similar specification of the linear parts such as the gaskets, aluminum plates and caps, either continuous extrusion or interlocking ones.

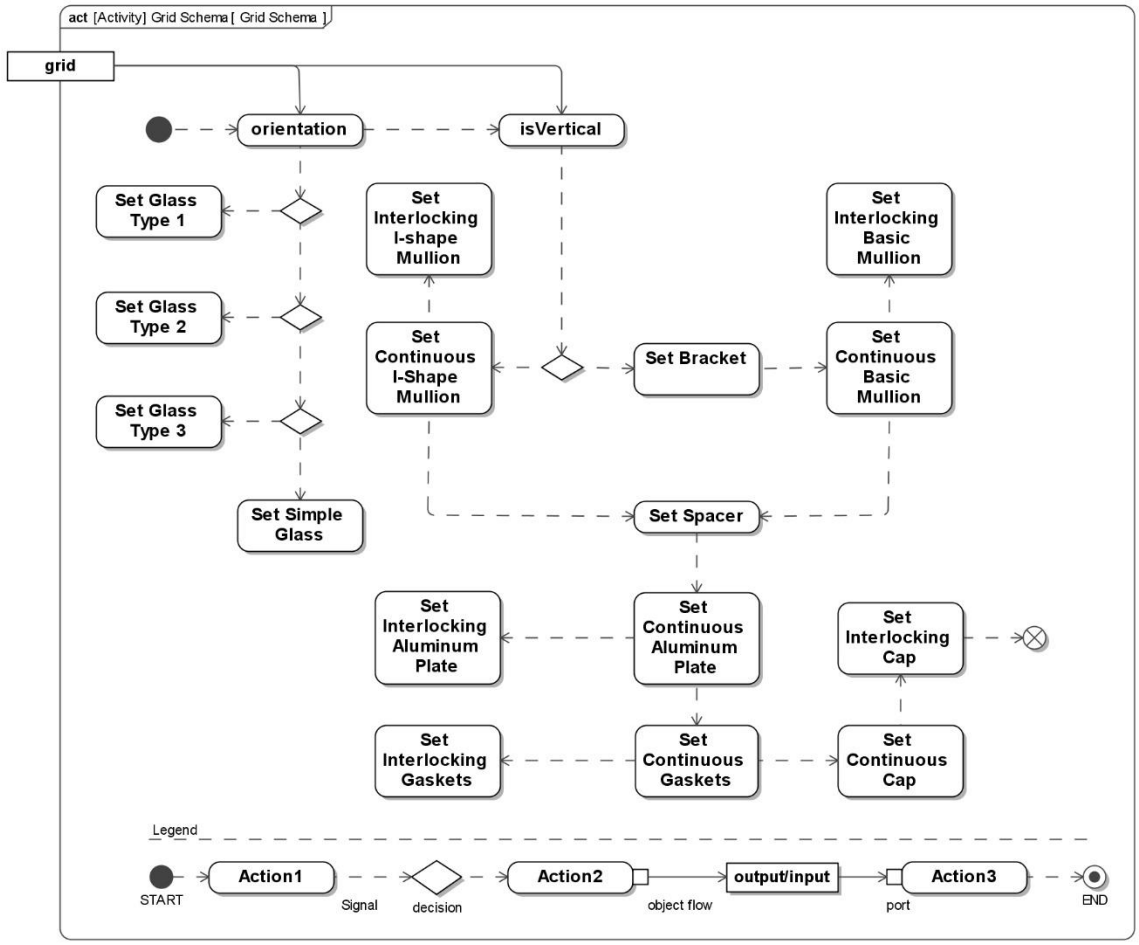


Figure 6.5. Protocol of specification of instances

Table 6.3. Case 1, rules of the protocol interpreted from the transcriptions

interpretation / rule	*transcription*
Defining the metal mesh on the South (S) and West (S) orientation *If (Orientation == S or Orientation == W) then* *Set metal mesh glass Type1,2 or 3*	*The glass has both Argon and Krypton gas depending on the location on the building... Only the 50% of the building is covered in glass with mesh integrated, which is only in the surfaces that face South and West*
No metal mesh in North (N), East (W) or Down Ward (D) orientation *If (Orientation == N or E or W) then* *Set Low-E coating glass*	*…all the North, East and down wards surfaces are actually low-E coatings with pure glass. There are large regions of the building that not have meshes, the mesh is only there where is needed.*
Use structural I-Shape Mullion on vertical surfaces *If(IsVerticalSurface == true) then* *Set I-Shape Mullion*	*But because we don't have diagonal steel behind our vertical curtain wall we need to span floor to floor, and in this case we are spanning on the diagonal… The "I" shape box mullion that has been engineered to span 17' on the clad.*
Use non-structural Basic Mullion on slope surfaces *If(IsVerticalSurcafe == false) then Set Basic Mullion*	*(The) basic mullion extrusion is 1" ½ thick that goes on the slope areas on the building. It only has 4' spanning capacity.*
Place brackets every 4' along the dominant directionality *If(Continuous Basic Mullion== true) then* *Set Bracket every 4'*	*The brackets only happen in the continuous mullion, the brackets only seat on the continuous steel… Every 4' there is a bolted connection back to the flange of the steel, which is 4" wide...*
Continuous extrusions on the dominant directionality *If(IsLinearAxis==true) then* *Set Continuous Mullion* *else Set interlocking Mullion*	*Steel in one axis is linear and the infill is stitched in, the mullion (I-Shape or Basic) is linear in one axis and has an interstitial transit that interlays the other axis*
Beside the mullion, vertical and slope surface have the same specification *For(each surface) then*	*The front parts of the two types of extrusion (Basic and I-Shape Mullion) are identical. So, everything from that forwards is identical for both curtain walls*
Add stainless steel spacers every 1' in every direction *Add (Spacer)*	*…stainless steel spacers are the most reliable long term solution. The pre-assembling and indexing of all the components is incredible.*
Add the aluminum plate that supports the water barrier tape in every direction *If(IsLinearAxis==true) then Set Continuous AlumPlate* *else Set interlocking AlumPlate*	*That stainless steel plate is there to support the tape... This aluminum reinforced tape unrolled onto the glass and just forms this continuous barrier. And the tape is flexible and handles all the size difference in the glass. So the façade has three layers of water proofing*
Given an array of points, where r is the number of rows of point and c is the number of columns of point, find four point to insert the glass. At the end of the process trim the borders *For(int i = r; i = (r*(c-1) ; i = i+2)* *Add (glassPanel (point[i], point [i-c], point [i+2], point [i+r+1]))* *Next* *Trim(glassPanel)*	*65% of the glass panels on the Seattle library are actually non-standard. Even though the infill panel is forced to the scale of 6x7 panels across the entire facade*
Add the Cap *If(IsLinearAxis==true) then Set Continuous Cap* *else Set interlocking Cap*	*What is very different about this curtain wall is that the face cap is actually holding the glass on the building*
Add the Gaskets similar to the Mullions and Caps method *If(IsLinearAxis==true) then Set Continuous Gasket* *else Set interlocking Gasket*	*The gaskets are actually going onto the extrusions.*

The result of the combination of these rules in this process is a large list of instances representing every single part. Figure 6.6 shows a fraction of the total number of parts generated while traversing the wireframe.

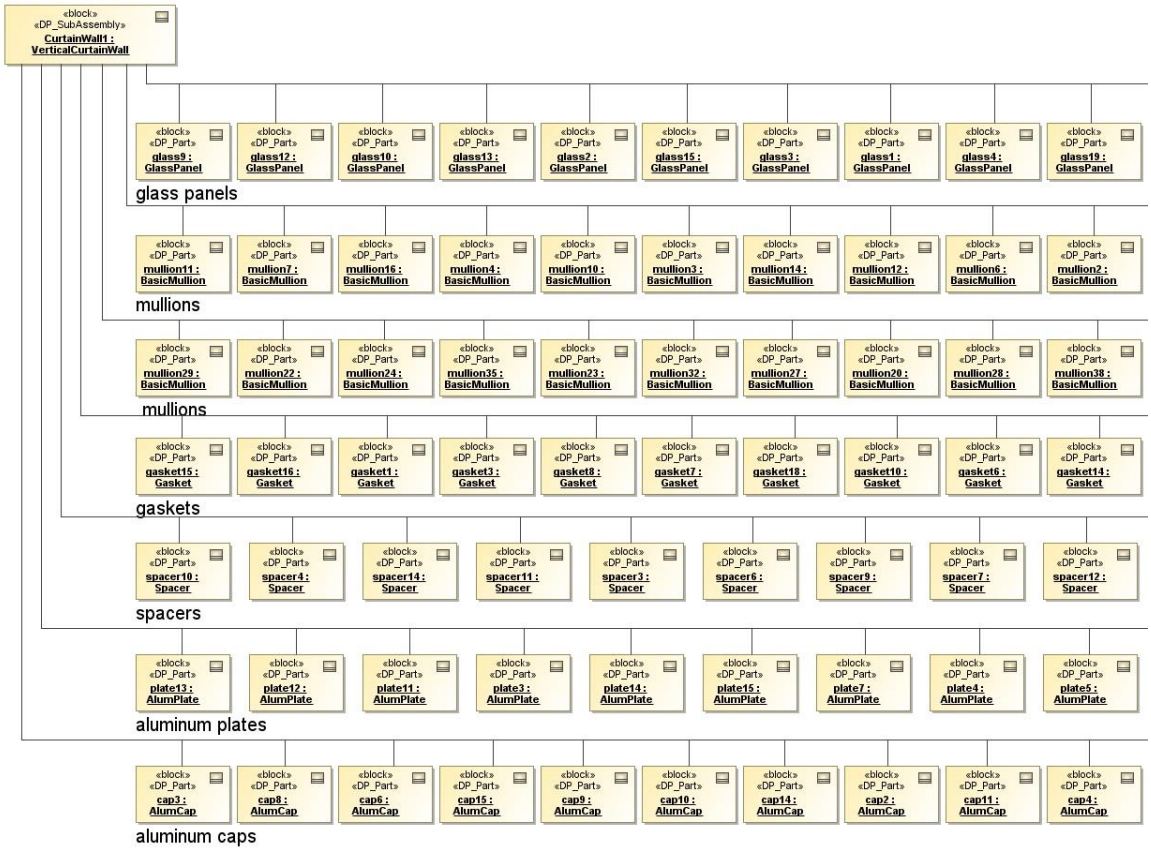


Figure 6.6. Fraction of a resulting list of instance specifications of a curtain wall façade

Geometric Representation based on Parts

Propagation of objects can be materialized thorough single parts, sub-assemblies, or conceptual structures or. This case study explores the propagation of parts that share the same reference geometry. Every part requires one, two or four points as input geometry for instantiation. To minimize the amount of parts of the actual geometric model, the technique of “inserting feature” is used. Digital Project can insert parts, copy of parts, or a very compact representation called feature, which is a single node on the binary tree of the geometric model that do not reveal its internal structure as the parts and copies do (Figure 6.7). Although the resulting arrangements of such

amount of parts can achieve distinct configurations from the topological perspective, because the parts are over-constrained, they exhibit limited flexibility to produce geometric variations (Figure 6.8). In fact, the flexibility relates to the number of components and the anatomy of the array rather than geometric variations of the arrangement of parts.

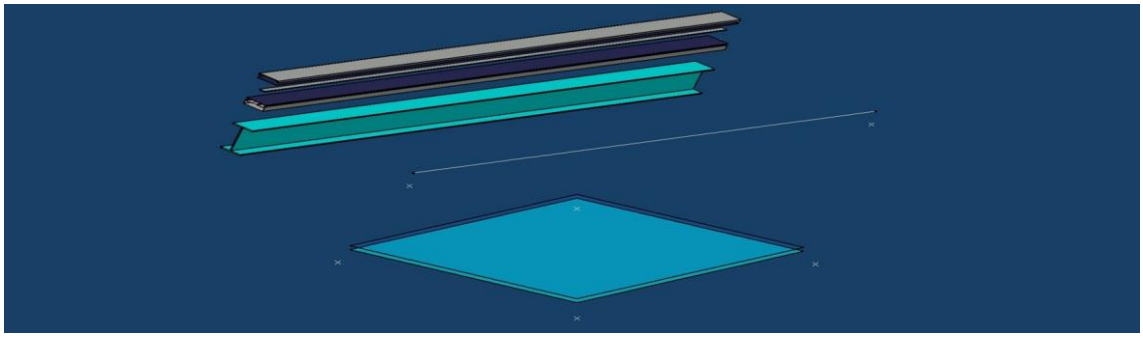


Figure 6.7. Two and four point features

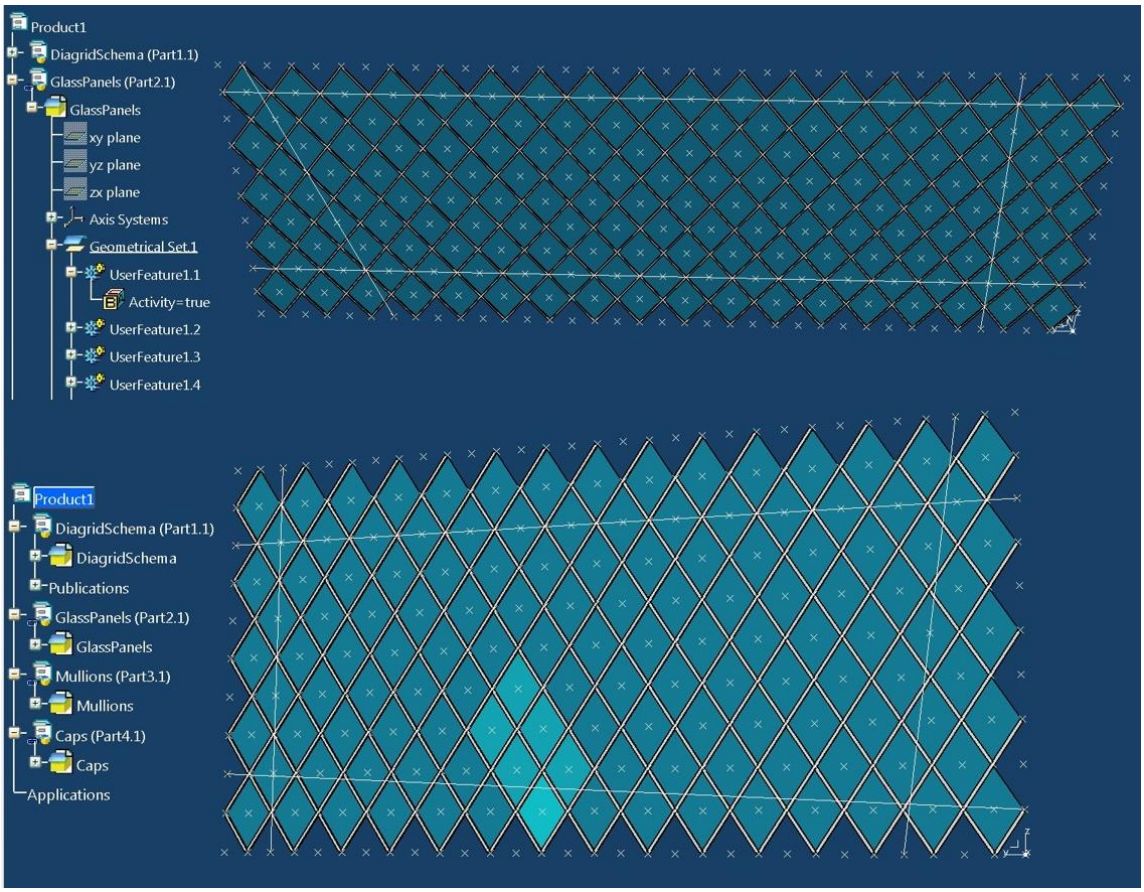


Figure 6.8, Alternative design configurations

Case Study 2: The Sequence Schema

Module-based Wireframe

This schema is based on a modular subdivision of the panel in units of 2 ft. The upper Vierendeel beam and the hanging frames of the structural frame are subdivided based on that module. The windows, brise-soleils and the rein screen panels are also modulated in the same way. The only substantial difference is the number of modules of every part.

The schema controls the number of modules that triggers the script producing the wireframe (Figure 6.9) that contains the grid of points and the parameter to control de offset of the reference planes for the rain screen, insulation, structure and interior finish.

Figure 6.9. Via Verde Wireframe

Protocol of the Configuration Specification based on Conditionals

This case study requires deeper interpretation of the schema than case one. The expert describes the parts and sub-assemblies, their composition and derived attributes. However, the rules that define the pattern of the subdivision of the mega

panel remain implicit. Nevertheless, by observing the family of panels of the building they can be deduced.

The *Design Schema* implies complementing the *wireframe* with a protocol of implementation. The protocol is a collection of distilled and deduced rules. Such deductions are tested in the generation of the specification of configurations of the mega panel. The process of defining the pattern of the subdivision of the maga panel requires the size of the module and the number of modules as inputs to create two sequences of numbers. Each number represent an amount of modules that in total must match the number of modules of the input. The reading of the sequences defines the composition of the panel. The upper sequence controls the propagation of rain screen panel that covers the Vierendeel beam, and the lower sequence determines the openings of the mega panel represented as negative numbers (Figure 6.10).

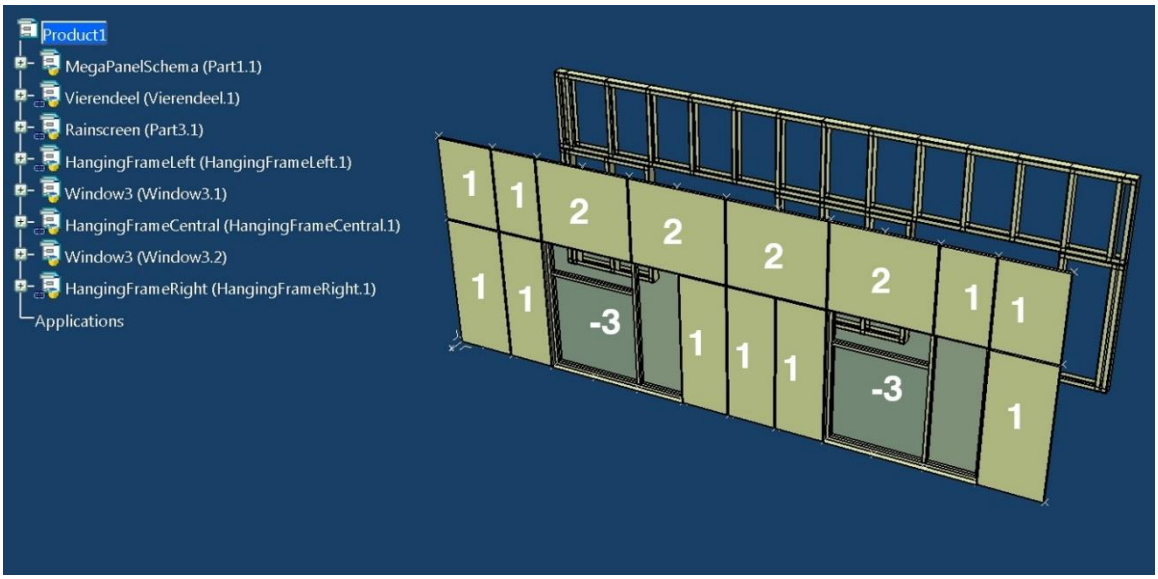


Figure 6.10. Sequence driving the modularity of the mega panel

The protocol (Figure 6.11) deduced from the transcriptions (Table 6.4) and documentation creates the sequences. These list of numbers represent the number of the pattern of the existing mega panels of the building stored I sequences such as

{1,1,2,2,2,2,1,1} and {1,1,-3,1,1,1,-3,1} or randomly create variety of patterns for exploration of alternatives.

The reading of the upper sequence leads to the instantiation of the major object of the structural frame, the Vierendel beam, modulated according the inputs values of the protocol. The panels of the rain screen, that need four corner points from the wireframe for instantiation, are created according to the intervals produced by the sequence. The reading of the lower sequence is more complex since it presents a chain of conditional statements to evaluate every number on the sequence and decide the instantiation of the hanging structural frames that have four specialization, the corresponding rain screen panels, or the tree possible sizes of windows of one, three or nine modules lengths. In addition, every window includes a similar size brise-soleil, and the insulation also match the process of instantiation of the rain screen.

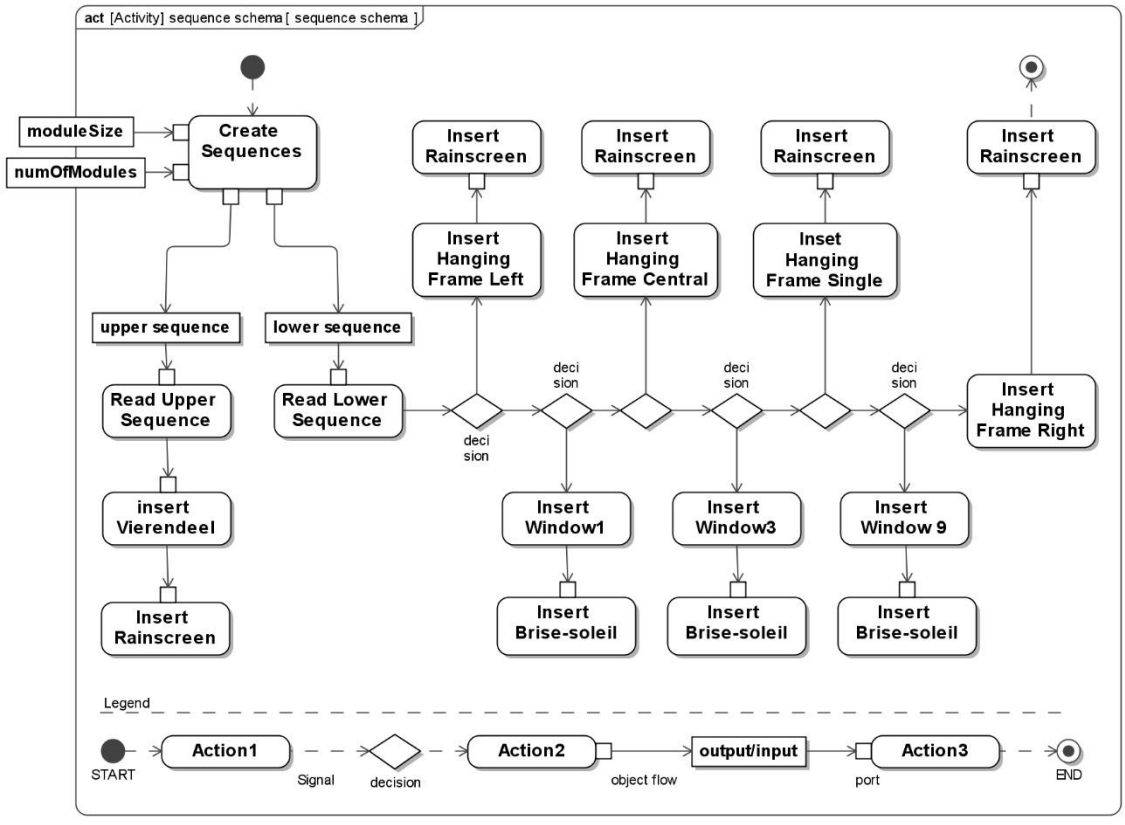


Figure 6.11. Activity diagram describing the protocol of propagation of objects

Table 6.4. Case 2, rules of the protocol interpreted from the transcriptions

Interpretation / parameters & rules	*transcription*
Center the Vierendeel beam with the edge of the slab *Move(Vieredeel)*	*The edge of the slab crosses right in the middle of the Vierendeel, where it is welded on to the slab.*
The structural frame, including the vierendeel beam and the hanging portions have a module *module = 2'*	*… the (mega) panel was made on galvanized plate cold form sheet metal.* *Every lateral portion (the hanging frames) are made by a module, a lower steel member and a hanger.*
The mega panel shows two different sequences of modules. While the upper values controls the modulation of the rainscreen, the lower values the distribution of hanging frames, windows, doors and the lower rainscreen *CreateUpperSequence(module, numOfModules1)* *for(each value)then Insert(Rainscreen, value)* *CreateLowerSequence(module, numOfModules2)* *for(each value)* *If(value < 0) then Insert(Window, value), Insert(Brise-soleil,value)* *else if(value > 0) then Insert(Frame, value), Insert(Rainscreen, value)* *else* *Insert(Door, value)*	*The entire structure of the panel is a Vierendeel beam with two hanging portions to the right and left leaving space in the center for the door (and the windows)* *…this window is kind of hanging in the space. It is not seating on the panel below* *… Normally … a crew installs the brise-soleil (on every window), and the whole panel is lifted up to the building.* *… In a mega panel the door is half the way in a one panel and half the way in the other panel. The door is actually installed after the fact.*
The R value depends on the thickness of the insulation. 4.5 R per inch of mineral wool, which defines the offset parameter of the reference plane for a target R *insulationOffset = R/4.5*	*Normally we put between 3" and 5" (of insulation). From the energy stand point you get a building 60 % opaque, and 3" gives you R13, 4.5 R per inch. If you want R16 just add one inch to the insulation.* *Code is R 13 for a wall. This building has R24 for walls. U-value for the glassing is 0.5.*
Parameter for the rainscreen reference plane *rainscreenOffset = insulationOffset + 2"*	*.. if you put the insulation outside it must be …mineral wall* *The only issue with mineral wool that if it gets wet, its U-value decreases. And the idea is you have a rain screen on front of it, so you don't have the problem of humidity degradation* *…and then you can have 2" of air plus the rain screen.*
Preliminary estimation of cost based on srft *GetEstimation(area, insulation thickness)*	*The idea is that you can say I have $64 sqft do you want to add extra insulations? … You got almost a parametric cost model that is so easy to manage …engineering fabrication, shop drawings, installation and water proofing is $94/sf for everything, including the balconies, the brise-soleil and the rain screens*
Installation tolerance *SetInstallatonTolerance(¼")*	*… We were concerned more about install tolerances, and it was plus or minus ¼" when it should be 1/8". In curtain walls it is 1/8", and you should work on plus or minus 2mm on the fab*
Set the water proofing tolerance for the stack joint *SetWaterProofingTolerance(12pounds/sqft)*	*We put the water requirement to12pounds/sqft pressure of infiltration not 50 pounds…* *… basically we are taking the studs and skinning them with aluminum extrusion to achieve a unitized curtain wall like water proofing strategy*
Set the tolerances for thermal expansion of the joint perimeter that will affect the water proofing *SetThermalExpansionTolerance(Value)*	*So, you start to see the joint perimeter. These joints are slightly larger. They are larger because they need to handle thermal expansion...*

The described protocol of creating instances from the blocks of the meta-model following a sequence of numbers, produces model of instance block or *Configuration Specification* in SysML language (Figure 6.12). The parameters of the module and number of them are combined to define most of the values of the attributes of the instance blocks such as dimensions of position within the assembly. The methods implemented in the *MetaModel* class allow retrieving the values of the attributes of the instances as well as the stereotypes that define the locations of the parametric models. While reading the instances, the methods of the Interpreter class use these values to automate the production of a VBScript file that launch Digital Project and creates, inserts, moves, and cross references the parts of the assembling during the Geometric Representation.

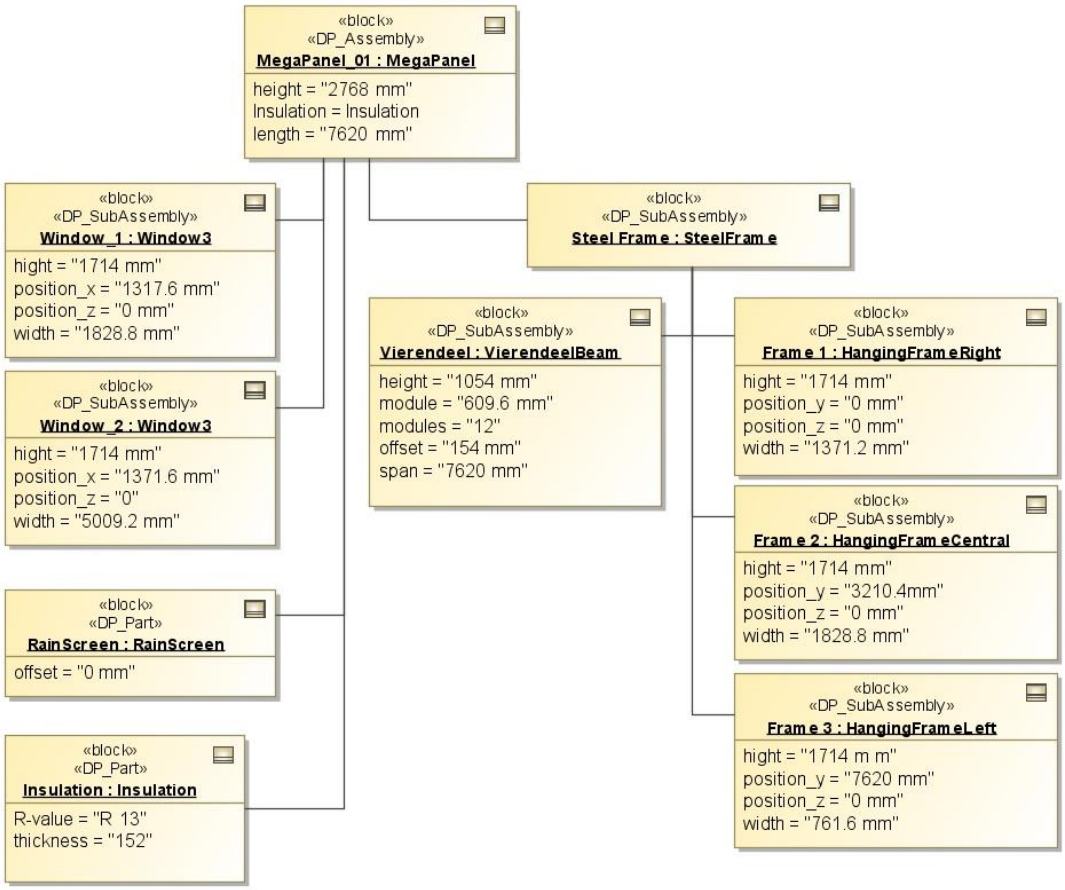


Figure 6.12. SysML instances of the Configuration Specification

Geometric Representation based on Sub-assemblies

The manipulation of the sequence-based schema creates various configurations with different topologies (Figure 6.13). These variations range from the reproduction of existing mega panels to randomly generated alternatives of various sizes, including anomalies beyond the set of rules of the protocol. These malformations show the boundaries of the validity of the protocol, and define the start point for evolution towards robustness.

The methods of the interpreter that manipulate the parameters of the parts during instantiation assure the well-formedness of the parametric structure of every configuration. Since each one represents a design space of geometric variations of the same model, the population of configurations multiply the search space for design solutions. In other words, the new design space is a combination of geometric and topological variations.

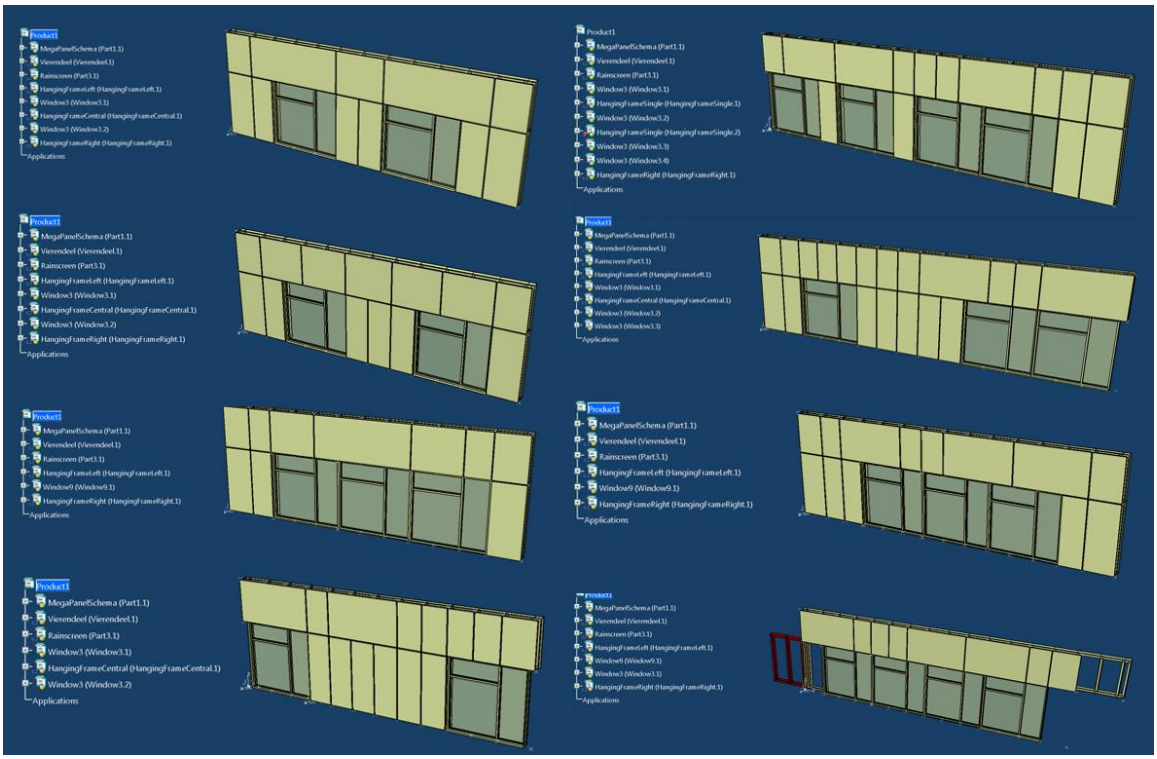


Figure 6.13. Case study 2, generation of alternatives with different configurations

Case Study 3: The Filling Schema

Cell-based Wireframe

This schema creates a wireframe that subdivides the mega-panel into 1 by 1 feet square cells defined by four corner points. The list of glass panels of various sizes that complete the collage have the same module. The resulting grid of points represents the field that the glass panels will fill. The schema determines the size of the field and the number of cells. There are 48 collage mega-panels of various sizes. Regular areas of the façade concentrate most of the repetition of the panels, and its lower levels most of the singular sizes. The glass panels are organized in 32 sizes, and the largest piece of glass is 7 by 16 feet long. The wireframe provides the four corner reference points that accommodate all of the glass panels, shown in Figure 6.14.

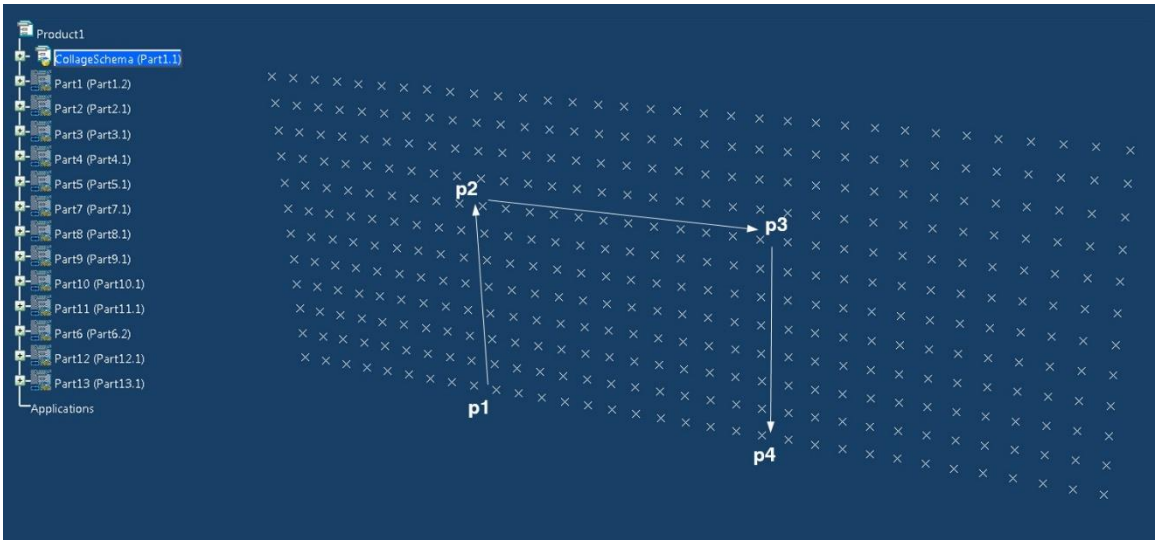


Figure 6.14. Process of inserting a glass panel into unoccupied cells

Protocol of the Configuration Specification based on External References

The design intent is the generation of a collage-like design. For such a purpose, the original approach is based on specifying a list of various sizes of glass panels that fill every collage mega-panel. Other important requirements are defining the position of the main glass panel where the view is most appreciated, followed by positioning the

operable window close to areas where natural ventilation is needed. The interpretation of these intents and requirements is a protocol of a sequential specification and insertions of the glass panels. After the main glass panel and the operable windows are defined, the remaining glass panels are inserted into the available cells until the collage is complete. Figure 6.16 shows the insertion sequence.

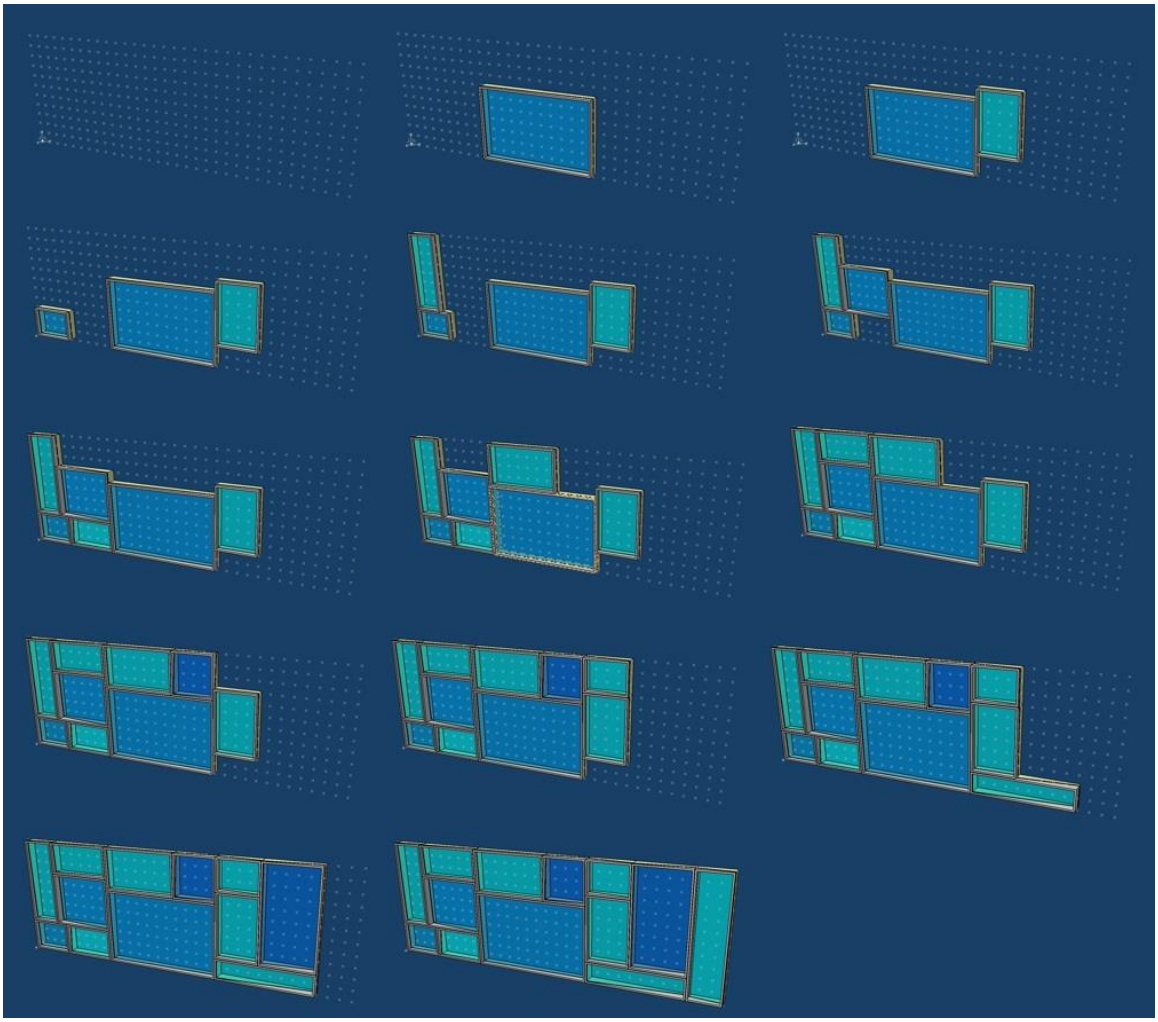


Figure 6.15. Sequence of insertion of glass panels into the field of cells

Figure 6.16 describes the protocol based on the interpretation of the transcriptions of the descriptions of the rules and parameters by the expert. The protocol is sorted set of rules that specifies the instances of the collage mega-panel before insertion. The process requires the list of panels to insert and an abstract array of

cells consistent with the wireframe. After the list of sizes are sorted, the process begins with the selection of a glass panel size, continues with the definition of the glass horizontal and vertical tilted angles ranging from 0 to 5 degrees, and ends with the specification on the glass type among four options. After every glass panel is defined, the hypothetical cells that the glass panel are supposed to use are labeled as occupied. If the current glass panel does not fit in the wireframe, the process will continue with the next one until the cells are fully occupied. Only after inserting all the panels is the steel structure specified.

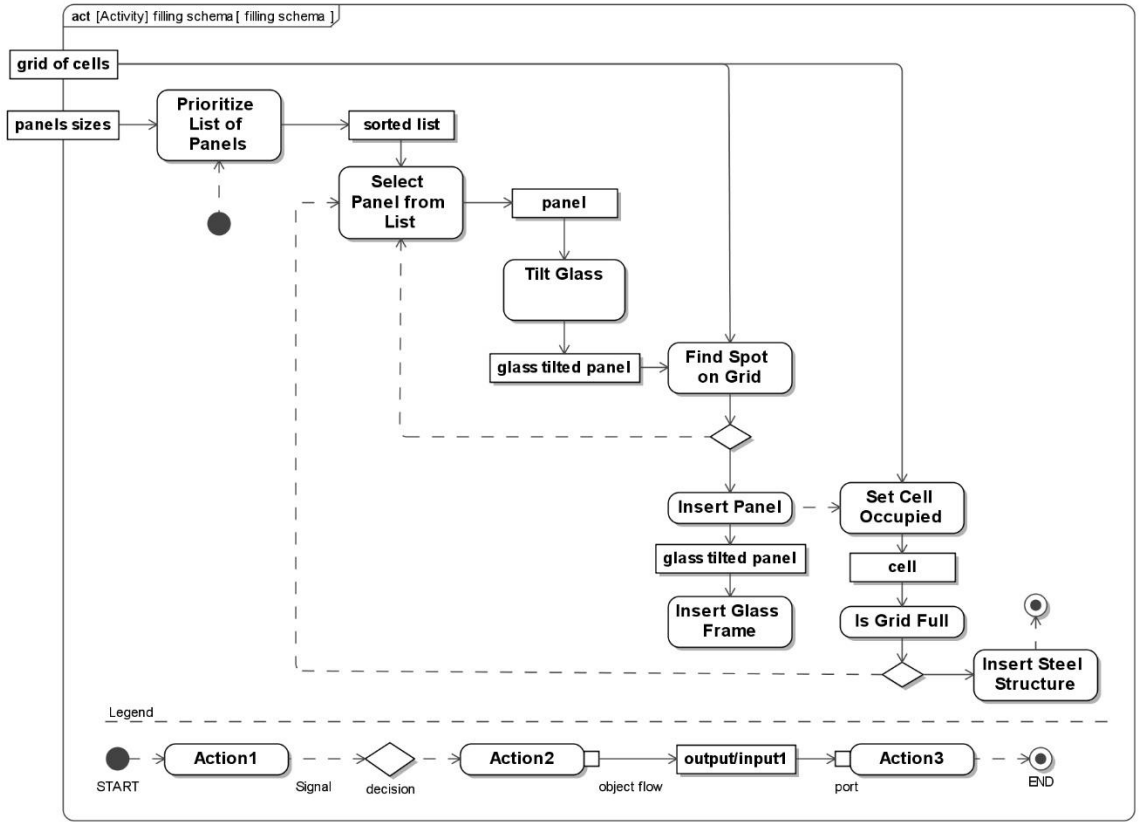


Figure 6.16. Protocol of the filling-based schema

Figure 6.17 illustrates the resulting configuration specification. Even though every glass panel entails a steel frame, the glass, and an aluminum frame, they share the same angle and dimension parameters. Therefore, for the purpose of this study they are grouped into one sub-assembly called “glassTiltedPanel”.

Table 6.5. Case 3, rules of the protocol interpreted from the transcriptions

Interpretation / parameters & rules	*transcription*
Definition of the size of the mega panels *regularHeight= 11'* *topFloorHeight= 18'* *length1= 18'; length 2= 20'; length3= 37'* *curvedPanelLength= 45'*	*We have vertical continuity. So, the whole building becomes rationalized into this mega grid, floor by floor and seven panels per floor plate. After that, everything has some variability in a kind of a crazy grid inside the panels* *The mega-panels varies from 11' x 18', x 20', x 37'*
Definition of angles of rotation of the glass *angle_Y = 0,2,3,4,o 5 degrees* *angle_Z = 0,2,3,4,o 5 degrees*	*Atelier Jean Nouvel provided a breakdown of the façade system as a composition of glass panels with four direction of rotation: tilting up, down, left and right ; four glass variations; and angles of rotation varying through 0,2,3,4, and 5 degrees*
Definition of the glass types based on color *glassType = color 1,2,3 or 4*	*There are four different type glass, they are all laminated glass*
Definition of the offset position of the main reference plane of the façade form the YX plane *planeOffset = 0*	*What is the only thing in this façade that is continuous? That is the exterior face of the structural steel. It will be a single plane across the entire façade.*
Definition of the section of the steel structure components *width = 3"* *depth1= 3"; depth2= 4"; depth3= 5"; depth4= 6"*	*The face dimension of every piece of steel on the inside is going to be limited and harmonized to 3", meaning that now we can play with the depth. And the depth was 3", 4", 5" and 6". Every single piece of steel except for the tube on top is ether 3 by 3, 3 by 4, 3 by 5, or 3 by 6. The bottom member is 6*
Derivation of the angle and dimensions of the steel plates that supports the tilted glass panels. These plates are derived from the angle of the glass.	*Steel profile protrusions at the intersections of the mullions vary in length to provide the specified angular tilt of each sub-panel. The triangular gaps between the glass panes and resulting mullions are closed with steel plates at the head and sill of each panel*
A spreader beam on the upper edge of the panel accommodates the brackets. Its length is equal to the length of the panel it belong to. *beamLength = length1, 2, or 3*	*A rectangular spreader tube beam behind the frame holds the bracket to put the panel in place on the floor*
Location of the largest glass panel and the operable window based on floor plan, next to the living room and next to the kitchen respectively *position_Y = user input* *position_Z = user input*	*How this pattern is made is actually very simple. You have a panel; let's say 11' by 37' and your kitchens close to the right and your living room close to the left. They said that we want our largest piece of glass close to the living room* *Then you put your window (close to the kitchen) and satisfy your fresh air requirement for the room…*
A list of glass panel defines a sequence of insertion. While the cells of the grid are not occupied, the panels are inserted according to the height and width of the glass. If the glass panel is inserted, then the cells must be set occupied *for(i = 0; i = rows of cell ; i++)* *for(j = 0; j = col of cells ; j++)* *for(n = i; n= glassPanelHeight; n++)* *for(m= j; m = glassPanelWidth; m++)* *isOccupied = cell[n,m].isOccupied* *if(isOccupied==false) insertGlassPanel(i,j)* *for(n = i; n= glassPanelHeight; n++)* *for(m= j; m = glassPanelWidth; m++)* *cell[n,m].isOccupied = true*	*Front's first step was to create a spreadsheet for organizing these parameters along with the glass and panel dimension. We were using excel design tables to drive the instantiation of the solid components*

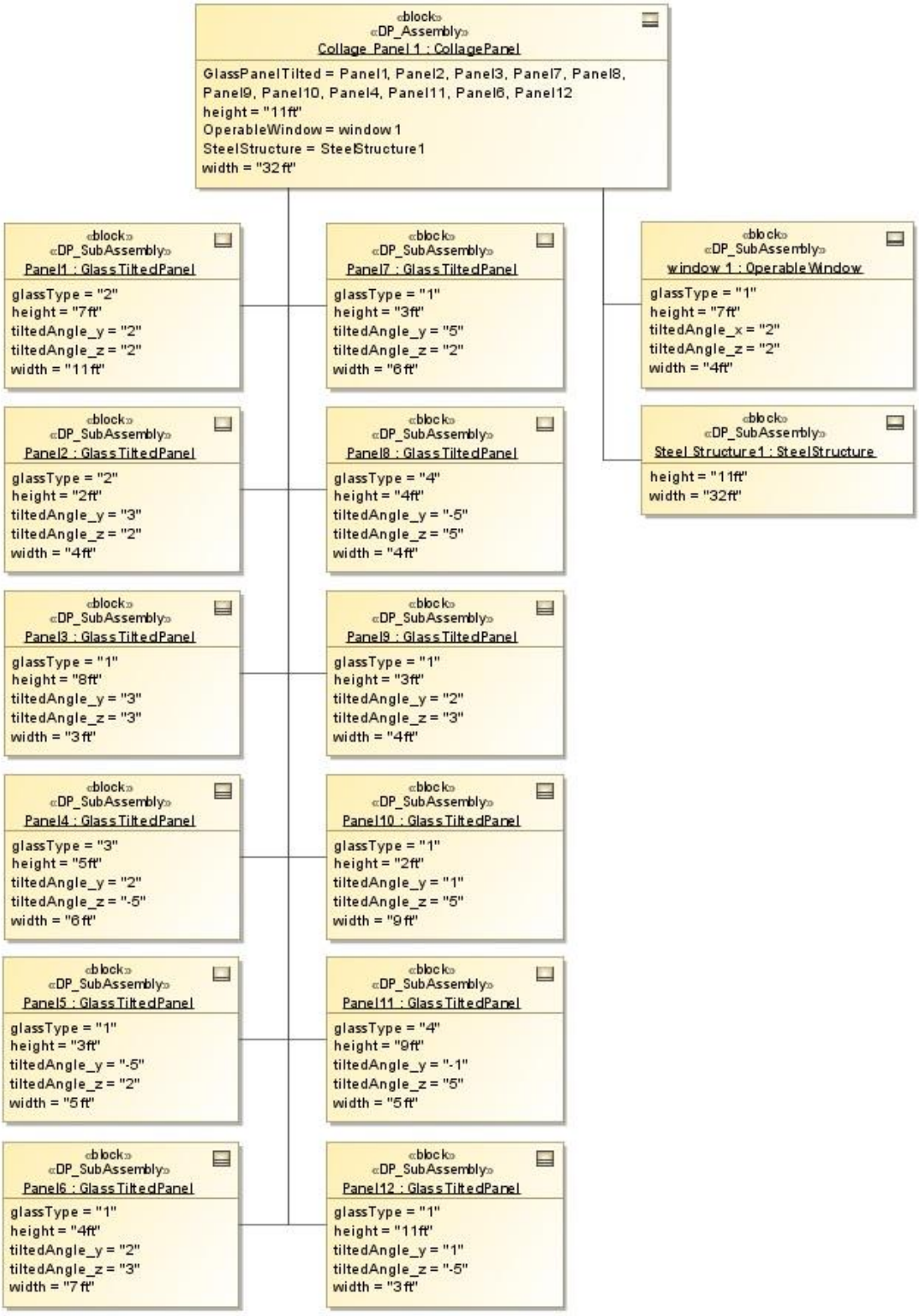


Figure 6.17. Example of configuration specification of case study 3

Geometric Representation based on Power Copies

This case study explores another method of the instantiation of parametric objects. It uses the “power copy” method of Digital Project. This method, which is not available in every parametric modeling tool, demonstrates the need for interfacing with the singularities of the means for geometric representation. It closely resembles the regular insertion of parts into the assembly. The main difference is that it requires geometric inputs for the insertion. The *Configuration Specification* provides a list of instances of the glass tilted panels and their attributes—glass type, height, width, and tilted angles—required for inserting the panel into the assembly using the wireframe as reference geometry. The protocol reads the specification and searches for the dimensions and then uses them for defining the four corners of the glass-tilted panels within the wireframe. During the insertion process of every glass panel, the protocol determines if there are enough cells available according to the size of the panel. Otherwise, it will continue traversing the wireframe until reaching a spot. One of the benefits of the propagation based on the power copy method is that all of the complexity of knowledge that relates to the definition of the part is embedded in the instance and manipulated from four points in this particular example.

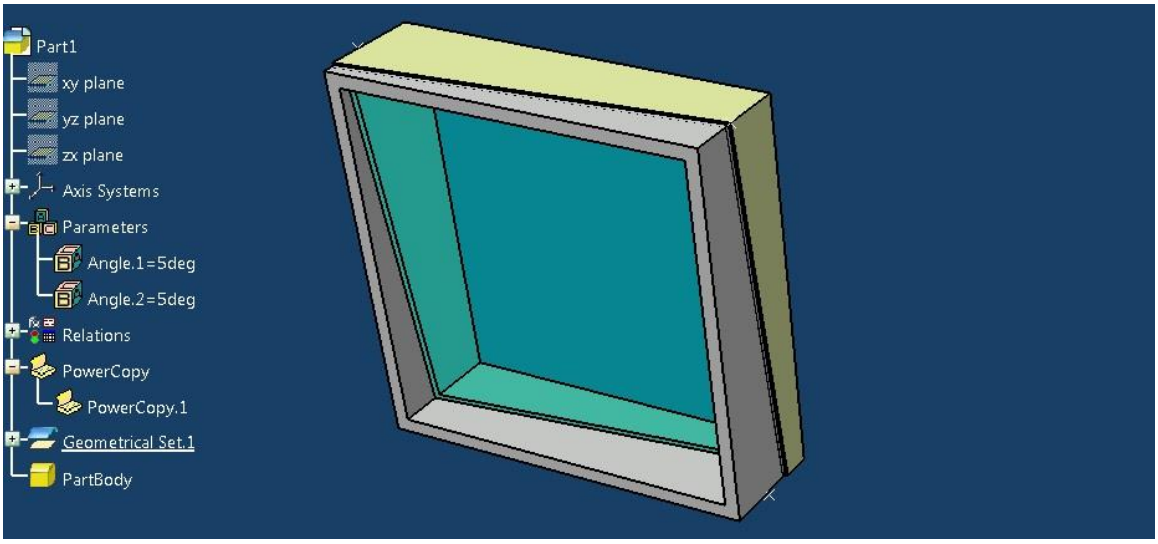


Figure 6.18. Glass tilted panel represented as power copy

The resulting configurations (Figure 6.19) range from the reproduction of the existing collage panels to speculations about more regular arrangements of glass tilted panels. Even though each configuration defines the glass panel in a fixed position with fixed dimensions, the parameters controlling the angle of the glass still allow some degree of manipulation. While reading the specification in addition to creating the product (assembly), the interpreter uses the method to add the instructions in the VBScript for inserting an object with four external reference points and to set the angle parameter values according to the specification.

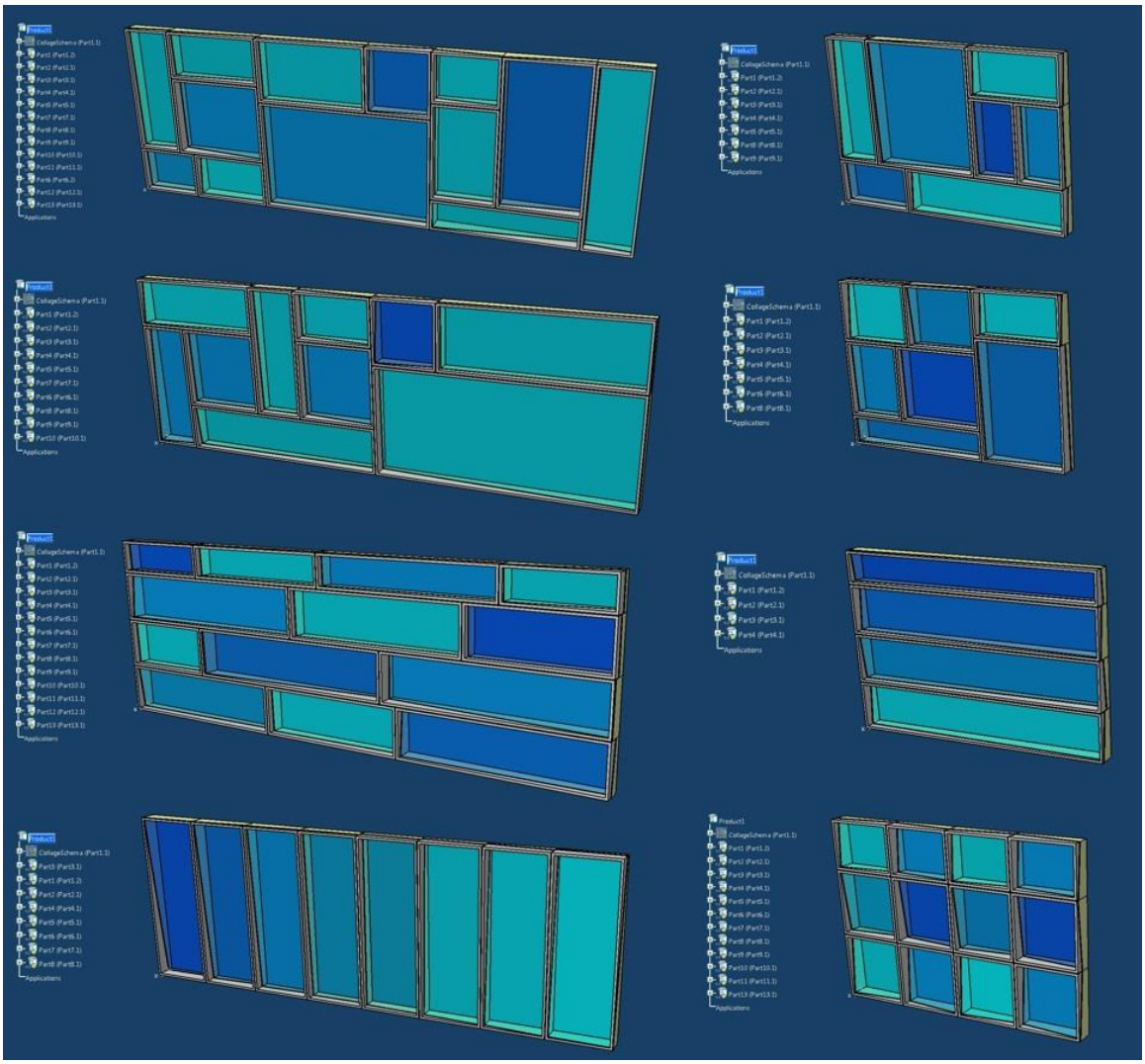


Figure 6.19. Examples of reproductions at the top and explorations at the bottom for two panel sizes

General Issues across the Case Studies

Although the Case Studies are driven by different types of schemas and the approaches for specification and geometric representation are not the same, features such as the capability of creating hybrids, the need for well-formed parametric structures and the resources for creating indicator to enable comparisons and selections emerge across the three of them.

The Hybrids

The notion of hybrids is rooted in the potential combination of parts from various case studies based on functional and instantiation compatibility. Functional compatibility refers to the type and the role of the part. Examples of exchangeable components are glass panels and different types of mullions or gaskets. Instantiation compatibility refers to the inner structure of the tri-dimensional model of the parts and the propagation mechanisms. For example, glass panels and rain screens require four input points for instantiation. However, a rain screen within a window frame does not seem to be a valid instantiation.

The following examples show the two types of hybridization. While Figure 6.20 specifies the glass tilted panels from case study 3 into the mega panel of case study 2, Figure 6.21 shows the propagation of rain screen panels over the collage schema of case study 3. The first example of hybridization verifies that the object is an actual window, the second only relies on the compatible four point instantiation mechanism. This notion of hybridization based on the two types of compatibility raises the question about how one defines the normalized standard to promote compatibility between schemas and parts. Even though these examples are limited, they demonstrate the need for further definition of the compatibility rules that allow the combinations.

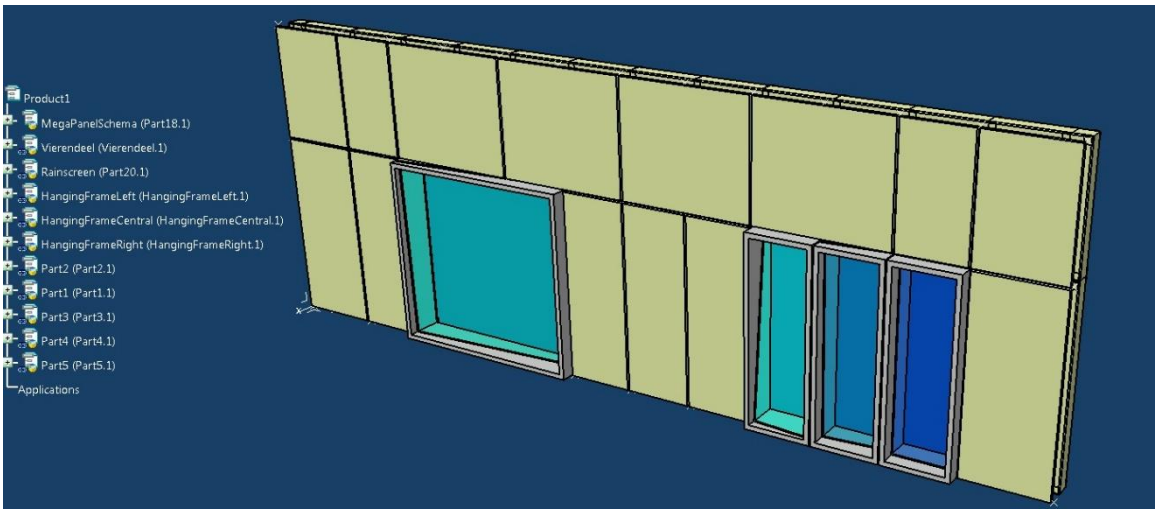


Figure 6.20. Exchange of windows based on functional compatibility

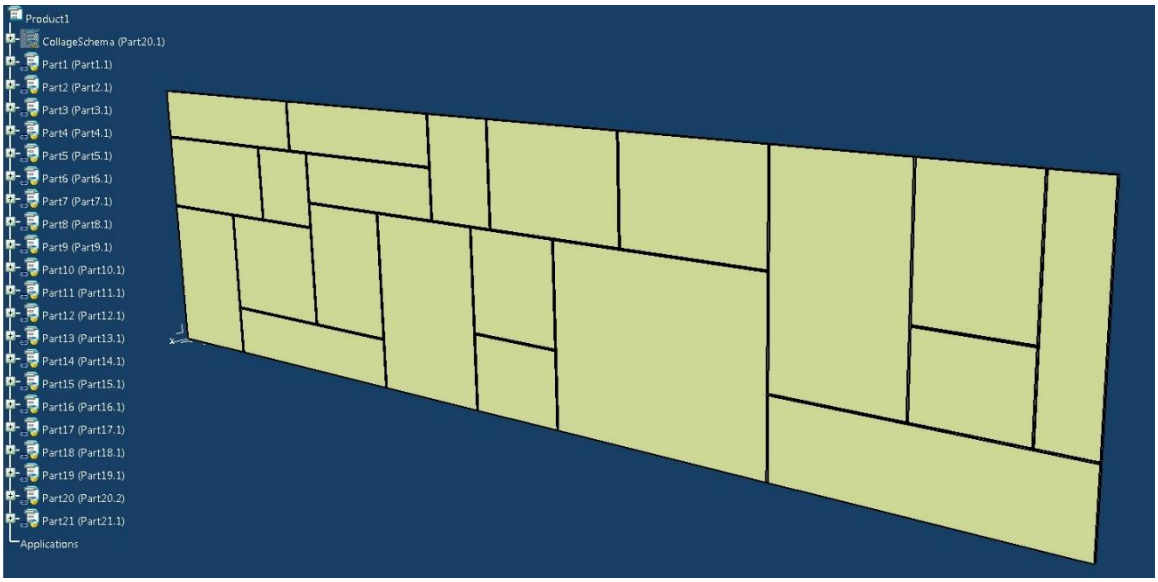


Figure 6.21. Propagation of the rain screen over collage schema base on instantiation compatibility

Well-formed Parametric Structures

The sharing parameters and reference geometry mechanisms are particularly important in the creation of well-formed parametric structures that allow the further manipulation of the parts of every configuration of the three case studies. The Interpreter class provides the methods for referencing the geometry of the schema from any part, and referencing the parameters of the parts to the driving parameters of the Schema. Figure 6.22 shows the linkage between the driving parameters of the schema

and the driven parameters of the Vierendeel beam. These parameters control the size and number of modules that define the array of studs of the beam, and the offset distance that defines the thickness of the insulation. At the bottom it shows two consistent variations of the mega panel. In the back, it shows the rain screen panels that use the points of the schema as external reference geometry for propagation.

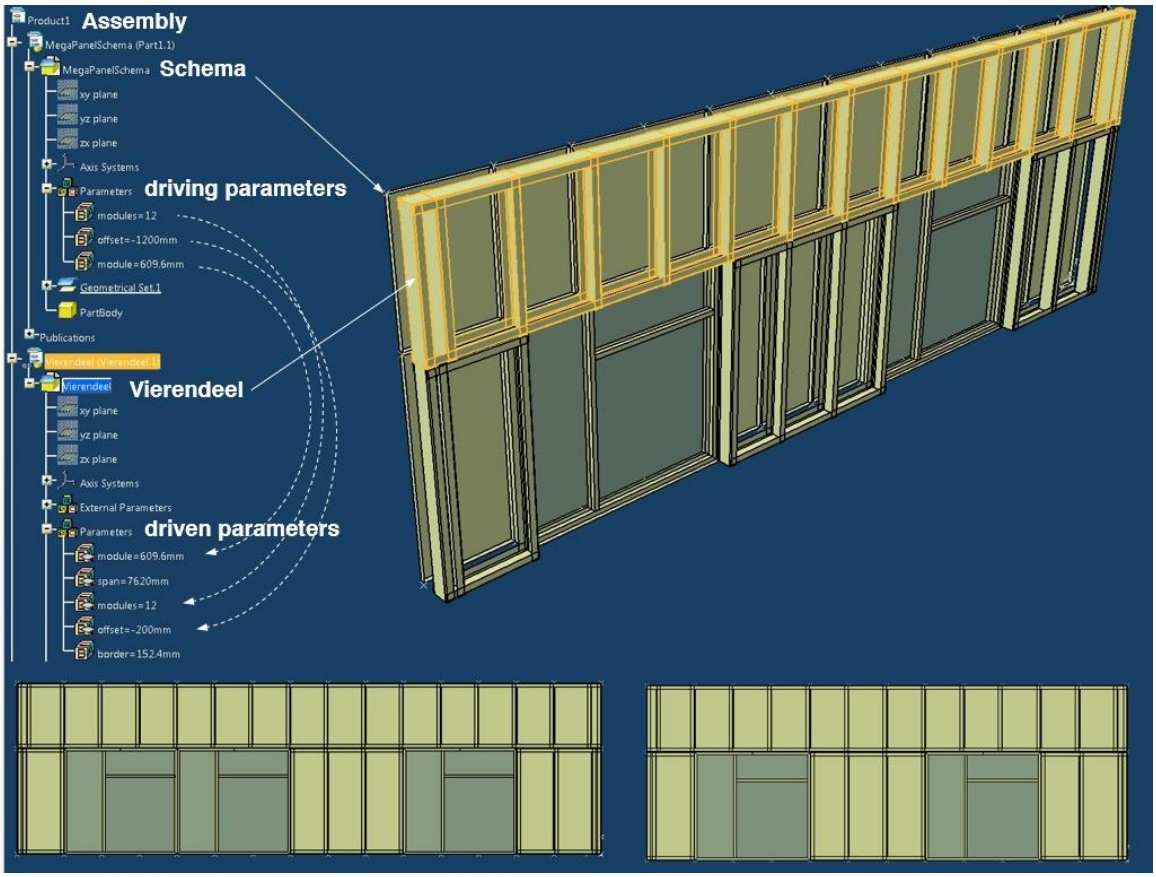


Figure 6.22. Well-formed parametric Structures

To achieve such consistency the following methods *addIntegerParameter, addDoubleParameter, setIntegerParameterValue, setDoubleParameterValue form* Table 6.2 are applied during the instantiation of the schema and parts; and the linkage between them relies on the methods *addExternalParameter* and *addExternalReferencePoints,* which builds the references to the points of the

wireframes. The following example corresponds to the method *addExternalParameter* from the Interpreter.class tool box. This method returns a CharSequence, which is basically a string that concatenates all the instructions. It takes as inputs the parameters of a source part, which in this case is a schema, and assigns them to the target part, which in this example is a Vierendeel beam. In the process, it creates the reference to the source parameters, and then consistently assigns them to the target. The resulting VBScript shows all the internals of the Digital Project tool. It implies that for every tool all these methods should be adapted for recognizing their singularities.

Example method from the Interpreter.class:

```
public static CharSequence addExternalParameter(String source, String target,
              String sourceParam, String targetParam) {
       index++;
       Script("'-----------------------------------------------------------------");
       Script("\'adding external parameters from "+source+ "to "+target);
       Script("\tSet ActSel"+index+" = oProductDoc.Selection");
       Script("\tIf ActSel"+index+".Count <> 0 Then");
       Script("\t\tActSel"+index+".Clear");
       Script("\tEnd If");
       Script("\tSet "+source+"Parameters = "+source+ ".Parameters");
       Script("\tSet param = "+source+"Parameters.Item("+"\""+sourceParam+"\""+")");
       Script("\tActSel"+index+".Add param");
       Script("\tActSel"+index+".Copy");
       Script("\tActSel"+index+".Clear");
       Script("\tActSel"+index+".Add " +target);
       Script("\tActSel"+index+".PasteSpecial "+"\""+"CATPrtResult"+"\"");
       Script("\t"+target+".Update");
       Script("\tSet relations1 = "+target+".Relations");
       Script("\tSet parameters1 = "+target+".Parameters");
       Script("\tSet length1 = parameters1.Item("+"\""+targetParam+"\""+")");
       Script("\tSet formula1 = relations1.CreateFormula("+"\""+"Formula.1"+index+"\""
+","+"\"\""+","+"length1," +"\""+"`External Parameters"+"\\"+sourceParam+"`"+"\""+")");
       Script("\t"+target+".Update");
       Script("\tIf ActSel"+index+".Count <> 0 Then");
       Script("\t\tActSel"+index+".Clear");
       Script("\tEnd If");
       return VBScript;
}
```

Resulting segment of the VBScript for Digital Project:

```
'--------------------------------------------------------------------
'adding external parameters from MegaPanelSchema to Vierendeel

	Set ActSel2 = oProductDoc.Selection
	If ActSel2.Count <> 0 Then
		ActSel2.Clear
	End If
	Set MegaPanelSchemaParameters = MegaPanelSchema.Parameters
	Set param = MegaPanelSchemaParameters.Item("modules")
	ActSel2.Add param
	ActSel2.Copy
	ActSel2.Clear
	ActSel2.Add Vierendeel
	ActSel2.PasteSpecial "CATPrtResult"
	Vierendeel.Update
	Set relations1 = Vierendeel.Relations
	Set parameters1 = Vierendeel.Parameters
	Set length1 = parameters1.Item("modules")
	Set formula1 = relations1.CreateFormula("Formula.12","",length1,"`External
Parameters\modules`")
	Vierendeel.Update
	If ActSel2.Count <> 0 Then
		ActSel2.Clear
	End If
```

Building Indicators

Even though the focus of this research is meta-modeling for generation and not decision-making for selection of solution candidates. The SysML instances of the Configuration Specification includes attribute values that allow deriving indicators for preliminary estimations or, at least, comparing options.

Table 6.6 shows how to get the attributes of an instance of the meta-model while iterating through all the instances of the meta-model.

Table 6.6. Example of UML resources to access the instances and their attributes

purpose	Example

Get attribute	`while (iterator.hasNext()){`
	`instanceClass = iterator.next();`
	`if (instanceClass.getName().equalsIgnoreCase("instance name")) {`
	`Property property = instanceClass.getAttribute("attribute");`

Counter	`if (instanceClass.getName().equalsIgnoreCase("bracket")) {` `numberOfBrackets++;`

The methods that get the attributes of the instances facilitates the creation of indicators by counting or combining them. Case Study 1 instantiates single parts instead of sub-assemblies. Figure 6.6 shows a list of instances created through the loops. Although the instantiation mechanism based on inputs points instead of parameters values hides the actual dimensions of the parts, counters can quantify the number of instances of every object for estimations. Case Study 2 instantiates sub-assemblies that allows accessing to the values of their attributes. For example the Vierendeel beam made of studs has the attributes “height”, “module” and “modules” that facilitate the task of quantifying the total linear feet of studs, the area of the rain screen, insulation, or the size of the windows. The offset parameter that controls the thickness of the insulation defines the indicator for the resulting R value of the mega panel since according to the expert:

…3” (of mineral wool) gives you R13, 4.5 R per inch. If you want R16 just add one inch to the insulation…

Case Study 3 instantiates objects based on copies. Similar that case 3, these copies have all the necessary attributes for quantifications. Although more research is needed to define the specific criteria to compare options, the meta-modeling process demonstrates enough flexibility to build indicators from the available information.

6.4. Validation

Validating the methodology of meta-modeling a design domain to produce a variety of design configurations is a process of building confidence in the usefulness of the set of methods that depends on their effectiveness and efficiency. Effectiveness is the capability of the methodology to correctly generate design alternatives, and efficiency is the capability to do so with acceptable operational effort. While an evaluation of the effectiveness of a methodology is expressed in qualitative terms, an evaluation of efficiency requires a quantitative approach.

The adopted method is the Validation Square (Pedersen et al., 2000; Seepersad et al., 2006), which proposes a combination of qualitative and quantitative approaches to evaluating design methodologies. An evaluation is organized around four notions of validity: structural, performance, theoretical, and empirical. *Structural Validation* refers to the qualitative evaluation of the effectiveness of the methodology, the roots in the literature, the logical structure and internal consistency of the methods in terms of information flow, and the appropriateness of the case studies for testing whether or not they represent the fundamental problem that motivates the development of the methodology. The second notion is *Performance Validation,* a quantitative evaluation of the efficiency of the methodology as it produces useful results. It implies that the methodology is consistent with its original purpose (generating a distinct design configuration), it is useful within an acceptable range for some key examples, and the improvements evident in the resulting design are the results of the application of the methods. The third notion, *Theoretical Validation,* evaluates the validity of the methods for general problems beyond the case studies. Finally, *Empirical Validation* addresses the validity of the selected case studies. From a combination of these four notions, four types of validity (Figure 6.23) of the evaluation of the proposed design methodology in qualitative and quantitative terms can be derived. The validation process of the proposed method implies a sequential fulfillment of the four types of validity based on individual criteria described

Table 6.7. Type of validity

Validation	Description
Theoretical Structural Validity	The proposed design methodology proposes a valid internal structure according with the general research problem
Empirical Structural Validity	The appropriateness of the chosen example problems or case studies for testing the usefulness of the proposed design methodology
Empirical Performance Validity:	The ability to produce useful results for the chosen example problems or case studies
Theoretical Performance Validity	The ability to produce useful results beyond the chosen example problems.

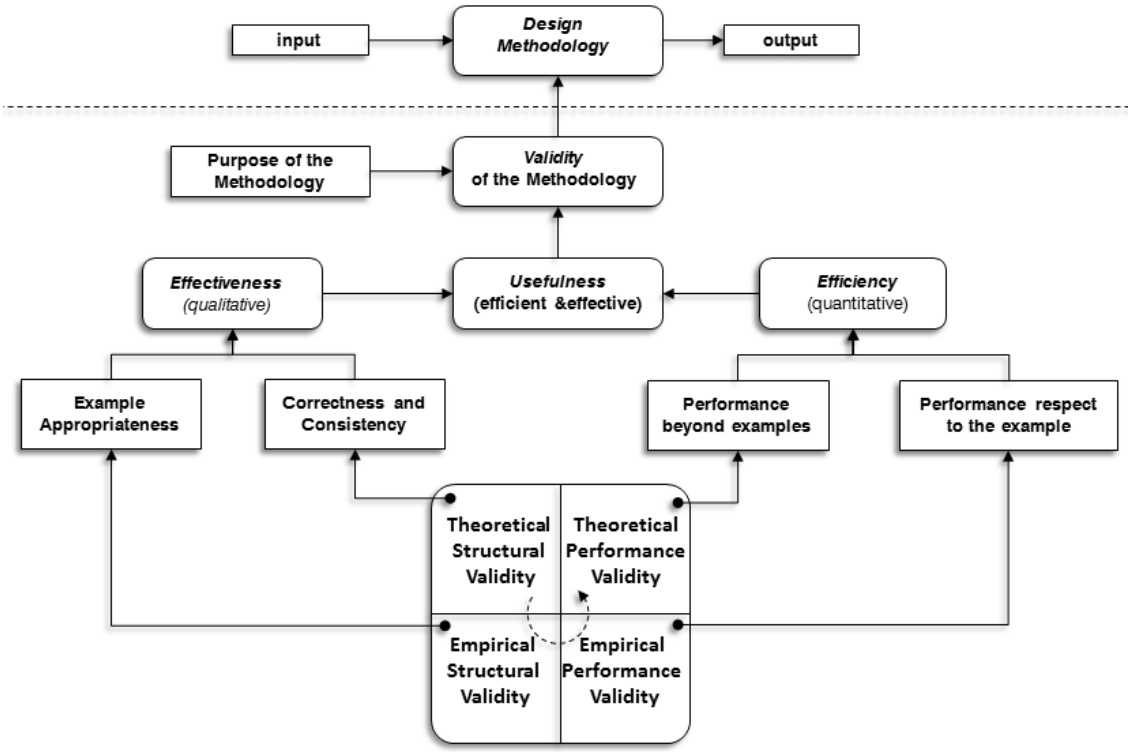


Figure 6.23 Validation Square

Consistency of Internal Logic

Internal logical consistency corresponds to the theoretical structural validity (TSV) of the proposed meta-modeling approach adopted from the MBSE process. It involves the validation of individual components and the coherence of the overall process of the methodology. Regarding the individual and overall aspects and in addition to the well-documented foundation in the MBSE and parametric modeling fields, the meta-modeling process of a design domain in its own semantic terminology, the representation of the objects of the domain in classes and their related attributes and associations, the capability of Stereotypes to map blocks of the meta-model with parametric models, the capability of the Interpreter to communicate instructions to the parametric tool, and the capability of BIM tools to represent geometry have demonstrated their effectiveness at performing individual tasks and consistently sharing

information. At this point of the research, the tests have demonstrated the reliability of the functionality of the process.

Appropriateness of the Case Studies

Case study appropriateness corresponds to Empirical Structural Validity (ESV). The chosen domain in which the proposed process is tested is the field of custom facades systems. As it is highly specialized, facade systems deal with the same set of phenomena, and it involves a limited number of parts. In addition, collaboration with the expert designer provides a necessary source of information upon which the meta-model of the domain is built and that is based on the already discussed case studies. Cases in the same field provide the context for distilling a general design framework and the opportunity for specialization and sharing objects.

Usefulness of the Results

This validation corresponds to the Empirical Performance Validity (ESV), or the ability to produce useful results for the chosen cases. Criteria for evaluating the usefulness are first, the capability of reproducing reproduce the case studies*;* second, the capability of manipulating patterns for producing configurations beyond the existing case studies*;* and third, the capability of producing hybridization among the case studies. The implementation of the Design Schema as a wireframe that drives the propagation of objects through many iterations gradually refines the process of recalling and adapting physical parts from a repository for either producing or exploring new configurations. The iteration also allows the normalization of parametric modeling methods that promote compatibility among the parts of different case studies. Such compatibility enables the generation of hybrid designs.

Scope of Usefulness

The scope of usefulness corresponds to Theoretical Performance Validity (TPV). Building confidence about the general usefulness of the design methodology beyond

the chosen example problems is based on the three previous stages. As long as the process has internal integrity, the case studies allow one to explore the boundaries of the problem, and the process effectively produces results. We can infer that the methodology is valid at least for design cases in the same domain. Furthermore, the results show an emergent framework that describes the interaction among physical components and patterns of organization, and the normalization of the parametric modeling process for reutilization. Both components of the process can extend the applications of the process to other domains by extending the range of design schemas and including industry standards in the development of the repository of parts.

6.5. The Implementation and User Perspectives

Capturing design expertise is not a one-step task. On the contrary, it is a process in continuous growth. It means that the start point, like in any endeavor, requires significant effort. This process has two perspectives: that of the designer with computational skills who implements the computational infrastructure and that of the user who grows, refines, and reuses the meta-model.

From the implementation standpoint, several steps must be completed to reach an acceptable functionality. Since this approach automates the creation of the meta-model using SysML, an extension of UML, the first step is to import to the Java programming environment, the Eclipse Modeling Framework, and UML resources for implementing the MetaModel Java class, which contains all the methods for creating and manipulating objects. The user can also manually create the meta-model using the Magic Draw editor. However, assuring the integrity of the working meta-model for the purposes of this research, it was automatically generated taking advantage of the previously mentioned imports. The second step is the implementation of the Profile and the Stereotypes that map the parametric models of the parts. Having applied these stereotypes to the SysML blocks that capture the object of the domain, the following

step is implementation of the Interpreter. The Interpreter Java class has a collection of methods that automatically create and concatenate scripts with instructions in the native scripting language of the parametric tool. These methods replicate the generic operations of the tools (i.e. create product, insert part, move part, set parameter value, share parameter) based on the reading of the Configuration Specification.

From the user perspective, not addressed in the user interface, the following steps represent what is expected from a user. Having a working meta-modeling computational infrastructure, the next step is the creation or edition of the parametric models of the parts and assemblies of the domain according to normalized guidelines that enable the compatibility of objects during insertion. The location of these files in the repository of parts needs to be mapped to the corresponding SysML blocks. If needed, a new block can be added to the meta-model. The hypothetical user must implement the function that creates the wireframe and the protocol that produces the Configuration Specification based on the values of the Design Schema to which they belong. After executing the protocol that creates all the instances, the interpreter produces the script with the instructions for Geometric Representation.

Even though this process involves several steps, the most laborious ones belong to the implementation phase, which builds the functional infrastructure. For a hypothetical user, the steps require more simple tasks such as parametric modeling of the parts, mapping of the parts and blocks, and implementation of key functions. All of these tasks can be executed by any designer with computational skills.

6.6. Discussion: Generation of Design Alternatives

The process of generation of specification and further representation demands the automation of the production of well-formed parametric structure, demonstrates the capability of extending the design space for searching solutions, and based on the compatibility of components supports hybridizations across the elements to f the three case studies.

Automaton of Parametric Structures

Three classical conditions of parametric models that describe their range of flexibility in supporting geometric variations: One is under-constrained, which means that not all the necessary parameters that execute a consistent change are included in the hierarchical tree of associations. Another is over-constrained, which implies that conflicting parameters control the same feature, resulting in frozen geometry. The third condition is well-constrained, that is, the desirable association of parameters that control geometric changes according to the design intent (Eastman et al., 2011).

The proposed process can effectively automate the creation of the well-constrained parametric structures of the assemblies of parts for every design alternative to effectively enable the automatic or manual control of the features of the design configuration. This augmentation is based on creating cross references among the parameters of the various parts of the assembly driven by the design schema or wireframe.

Extending the Design Space

Expert designers have the ability to identify many chunks of constraints in advance, which helps them frame and reduce the Design Space to the scope of feasible alternatives (Flager et al., 2009) by weighting variables and applying heuristics derived from professional experience (Kleijnen, 1997; Shan & Wang, 2010). While experts intuitively attempt to reduce the set of options, efforts focused on Computational Design

Optimization (CDO) (Barg, Flager, & Fischer, 2015) attempt to expand them. They operate under the assumption of variation of a given configuration that often means an evaluation of the geometric variation of the same parametric model. Various approaches to searching based on optimization algorithms take advantage of such a range of variation by iteratively executing changes in the input values to create new versions for evaluation. Extending the Design Space by generating several design configurations rather than geometric variations can better contribute to the reduction mechanisms that experts apply while traversing the space of options.

Towards Hybridization

Hybridization of case studies rely on compatible parts that perform similar tasks or behave in similar way during instantiation. We can understand hybridization from both their functional and instantiation compatibility. Therefore, the domain requires the development of a set of verification mechanisms along with the standardization of parameters and instantiation mechanisms that propagate the object within an assembly. Otherwise, it will produce what some theorists call “malformations” or “pathological variations” (Najle, 2013). Even though some of these pathological variations are still open to interpretation and can trigger unexpected readings, they are not valid models.

At least three fundamental conditions are required for hybridization or a flexible combination of parts. First, the interface of parameters that drive the parts should be normalized in such a way that similar objects share similar parameters with the schemas. Second, the anatomy of the schemas (e.g., grid, sequence, filling) should be normalized in such a way that it matches the characteristics of the instantiation mechanisms of compatible objects. Third, and the most laborious, functional compatibility should be verified. However, more research that focuses on developing methods for verifying the validity of the propagation of objects is needed. After all, even if the user is able to manipulate the schema, the computer can verify the process only by instantiation compatibility, not by functional compatibility.

CHAPTER 7

CONCLUSION

Unlike other design fields, design in architecture is characterized by a vast amount of tacit knowledge embedded in the patterns of design organization, complicating the declaration of elements of a domain. The literature in the design cognition field largely describes the effectiveness of the notion of patterns of design organization. From Alexander (1964) to Lawson (2012), we can find research efforts that elucidate and explicitly declare the logic behind this aspect of the design tasks. However, architectural design lacks of a formalization of its own process to take advantage of the efficiency of the reutilization of the design patterns. On the other hand, research efforts for a scientific and systematic approach to elucidate and formalize the engineering design (Hubka & Eder, 1987; Pahl et al., 1996), and more recently, the developments of languages that enable a Model-Based System Engineering process (MBSE) (Reichwein & Paredis, 2011) had succeed in the formalization that enable the representation and manipulation of the explicit knowledge it includes. While architectural design shows a tendency of imposing design patterns early on to drive the arrangement of physical parts, engineering design relies on explicit declarations of requirements, constraints, attributes, association and working principles to systematically synthetize design solutions through an iterative process.

The focus of interest of this research is an expansion of our understanding of the knowledge that experts embed in the patterns of organization of their designs to develop both external knowledge representations and computational methods that enable the reutilization and the manipulation of such knowledge to produce design alternatives. This research implements a prototypical methodology that supports the reutilization of such patterns by taking advantage of the meta-modeling process adopted from MBSE, which contributes to building a representation of the diversity of

objects of a design domain from abstract concepts to physical components. The meta-modeling process allows identifying the fundamental framework for reutilization of design knowledge, manipulation of the design patterns to specify and represent distinct design configurations, augmenting the design space in the search for design alternatives, providing mechanisms that support various external representations, and envisioning the migration form Parametric to Topological Modeling in Design as the next step of this line of research.

7.1. The Adoption of the Meta-modeling Process for Specifying Different Design Configurations

The adoption of the meta-modeling process acknowledges the limitation of the current parametric modeling technology to support changes at the design configuration levels. Even though it is well known the success of paramedic modeling for capturing best practices in custom reusable adaptable objects (Eastman et al., 2011), it has several limitations for supporting changes of the topological structure of the design configuration. Although the hierarchical data structures that computer programs use to represent geometry and its attributes such as Boundary Representation (B-Rep) or Constructive Solid Geometry (CSG) (Eastman, 1999; Kalay, 1989; Mantyla, 1988) facilitate parametric modeling and geometric changes, we can find limitations to supporting changes beyond the scope of the dimensional variations of geometric models. The topological structure of these models rely on sharing parameters and reference geometry in a binary tree structure. It means that a node in the tree is created for every cross-relationship between objects. Therefore, changing the configuration by deleting or adding objects during a hypothetical topological transformation either eliminates input parameters of geometry for the remaining objects, or add object not integrated to the structure of the model affecting the behavior of the model during further geometric variations.

To address such limitation and extend the range of configuration options for design exploration the meta-modeling process relies on the separation of the design *Configuration Specification* from the *Geometric Representation*. This differentiation provides the flexibility of specifying the configurations avoiding the complexity of editing the binary trees or topological relationships of the geometric models. The process of instantiation discussed in Chapter 6 specifies instances of the blocks representing objects, their attributes and association without building the parametric models. It implies that the associations across objects are discrete instructions and not actual nodes in a binary three. The interpretation of the *Configuration Specification,* intended as a list of instructions specifies in the parametric tool scripting language, builds well-formed geometric parametric models according to the advantages and limitations of the chosen tool. Another implication of separating the specification from the representation is that it facilitates preliminary estimations by accessing the data of the models before the geometric tri-dimensional representation takes place.

7.2. Framework for Design Expertise Reutilization

During the process of distilling design knowledge form the case studies and building the meta-model, several constructs common to all three case studies emerged through the process of generalization. We need to differentiate the constructs that specifically relate to the case studies in the field of custom façade design from those constitute the general design framework for reutilization. While the first group corresponds to the general categories that classify components of the façade systems into generic object types such as mullion, glass or gasket; the second group represents the taxonomy of objects that interact while creating a new configuration. These objects constitute the framework within which a domain can be continuously extended that are the focus of interest of this research.

The distilled and proposed *General Design Framework* (Figure 7.1) consists of two main types of objects that have been recurrently identified in the literature: *Physical Components*, which are *Parts* or *Assemblies* of them driven by *Pattern of Organization*, which corresponds to *Chunk of Constraints* (Gobet et al., 2001) that capture generic local restrictions, *Conceptual Structures* (Lawson & Dorst, 2009) which are abstract spatial or functional constructs made of parts or assemblies that locally organize the design, and the *Design Schema* (Lawson, 2004) that is the overall pattern of organization also called "schemata" or "design parti". Those three constructs, at different level of granularity, determine the organization of parts, from already known general restrictions to the overall schema that represent the design intent. The latter, in terms of actual implementation, uses a protocol of sorted *Design Rules*, which represents behavioral aspects of the design activity, and a *Wireframe* that all the *Parts* and *Assemblies* will use as reference geometry.

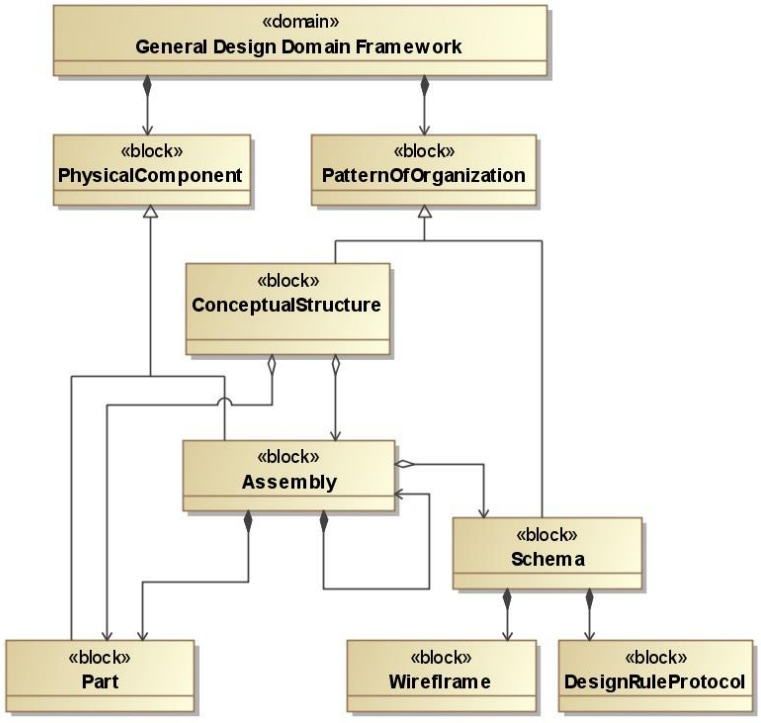


Figure 7.1. General design framework for reutilization

Results show that the influence of the *Chunk of Constraints* appears to be limited to the characterization of *Parts* and *Assemblies*, while the *Conceptual Structures* and *Design Schemas* operate on a larger scale. The first groups and provides meaning to large number of objects that constitute the main functional elements of the design, and the second to the overall logic of the assembly or final product. This particular notion of *Design Schema* that is supposed to represent the design intent has the attributes and links to trigger the *Wireframe* and the *Protocol of Design Rules* that actually implement the schema and aloe the propagation of *Parts*, *Assemblies* and *Conceptual Structures*. The first component is what the expert contributing to this research calls the *Wireframe*. These tri-dimensional models of auxiliary geometry define the reference input geometry for the entire arrangement of *Physical Components*. They store the main driving parameters, define the reference planes, directions, limits, intervals, grids of points, and any other relevant geometrical features that organize the propagation of objects. The Design Schema can either call a function to automate the generation of the *Wireframe* or a parametric file. The second object that is called by the Design Schema is the *Protocol* of rules that is a collection of distilled and deduced rules derived from the interpretation of the transcriptions of the description of the case study by the expert designer. The *Protocol* is the sorted set of rules that specifies the instances of the major objects or single parts. The combination of both is the basis for automation of the propagation of parts since the design knowledge resides not only the parts but also their assembly. While the *Protocol* execute step by step the abstract specification, the *Wireframe* provides the reference geometry to consistent insert or create *Parts* and *Assemblies* for creating a well-formed parametric product.

The efforts of this research have been devoted to expanding our understanding and manipulating the underlying logic that organizes the designs. The abstract constructs that participate in the process require explicit declarations that frequently are not available in conceptual design stages (Gross, 1996) and we still have problems

implementing representations of abstract concepts or ambiguous definitions (Dreyfus, 1992) in CAD tools. Extending the boundaries of the MBSE to supporting architectural design enables the construction of human-readable and computer-interpretable models that contain and structure a diverse collection of objects ranging from a single physical part to very abstract design conceptuality that express qualitative aspects of the design that rely on group of parts or sub-assemblies or abstract expression such as requirements or constraints. Meta-modeling stimulates progress in the integration of studies pertaining to the cognitive aspects of the design task and developments in computational design by fostering the implantation of several abstract concepts presented in design studies through the proposed General Design Framework for reutilization.

7.3. Parametric Modeling for Reutilization

The separation between the design *Configuration Specification* and the *Geometric Representation* determines the parallel development of an entire repository of parametric models that gather all the *Parts* and *Assemblies* of the domain. These parametric models, mapped to blocks of the meta-model, must follow general principles for reutilization since they are constantly recalled by the same generic commands such as insert part, move part, add reference from external geometry, link the parameters of other parts, and set the internal parameter values. These generic commands, found in any BIM tool and triggered by the interpretation of the *Configuration Specification*, allow both the parametric control of the overall assembly and the internal control of the parts.

To enable the reutilization of such *Parts* and *Assemblies* guidelines and normalizations of the models are required to assure the consistence between the *Configuration Specification* and the *Geometric Model*. Parametric modeling for reutilization requires identifying shared parameters across the *Assembly* and a normalization of the parameters that drive the *Parts* and *Sub-assemblies*, enabling the

generation of well-formed parametric relationships of the resulting product. Woodbury, Aish, and Kilian (2007) faced similar issue while studying pattern propagation techniques for a prototypical parametric modeling tools ("Generative Components ", 2016). While the primary purpose of the parametric structure of a model is to enable geometric variations, the automation of the propagation of Parts requires identifying the Driving and Driven Parameters. These *Driving* parameters are defined in the *Wireframe*, and they will be shared with the *Parts* and *Assemblies* of the final product. The *Driven* parameters determines the geometric variations of the parts, and they are controlled or "driven" by the first ones. This distinction is a key step for enable the control of the entire assembly from the *Driving* parameters of the *Wireframe*. After this fundamental distinction the parameters and reference geometry need to be classified according to the following categories: the parameters that will be controlled by another part, the internal hidden driven parameters of the parts, reference geometry from another part, and constant values across the overall *Assembly*.

The normalization also determines the need for a classification of the parameters from the perspective of the geometry instantiation process, which connects the internal complexity of object-related knowledge with high-level operations driven by assembly-related knowledge. Furthermore, because not all tools have the same capabilities, the chosen tool for representation determines the range of possible operations each time, and this interface of parameters must be robust enough to recognize a variety of tools.

This research distilled, represented and manipulated three different types of Design Schemas: *Grid*, *Sequence*, and *Filling,* Even though their anatomy is totally different, they demand similar normalization of the parameter and reference geometry. These complementary normalizations entail consistently naming parameters of attributes such dimensions, coordinates for positioning, thicknesses for example of insulation, reference planes or arrays representing either number of rows and columns of grid of points, or number of objects. Regarding the reference geometry, the three

schemas provide the inputs for objects that use one, two or four points, rail lines, or start and end reference planes for instantiation, which is independent of their functionality. Finally, the stereotypes also need normalize the attributes that link the abstract definitions of the SysML blocks with the files of the parametric objects form the repository.

In summary, the following conditions are required to recall and combine parametric parts. First, the interface of parameters that drive the parts should be normalized in such a way that similar objects share the same parameters with the wireframes. Second, the anatomy of new schemas added to the repository should be compatible with the characteristics of the instantiation mechanisms of objects to support combination of objects. Third, the functional compatibility should be verified before insertion, since parts with different functionality can share the same instantiation mechanism.

7.4. Multiplicity of External Representations

Although design knowledge is internally represented in the mind of the expert, the external representation of the same knowledge can adopt multiple file formats. In architecture, this is achieved essentially via geometric representation. Every time these representations are reused or adapted for a new or similar problem, some aspects of the solution are assumed to be embedded in them.

The adoption of the meta-modeling process, rooted in the MBSE field, contributes to structuring the design expertise regardless of the tool. Modeling the design domain independent of any tool or means of graphic representation to more effectively declare the knowledge of specific design-domain terminology precludes any reference to the complexity of the geometric representation. The meta-modeling process, provides methods, languages and the flexibility for creating a comprehensive taxonomy of the objects of the domain in abstract terms and mapping them to different

tools, either through automatic routines or the manual use of a graphic editor. While the high level aspects of the object such as their normalized parameters and reference geometry for instantiation are included in the abstract SysML blocks, their internal parametric relationships remain hidden in the parametric files. This separations facilitate the linkage of the meta-model with various repositories based on different tools by applying proper stereotypes that link domain-specific definitions with internal requirements of the tool.

The need for interpretation acknowledges that every tool represents the same object in its own way. The implementation of the *Interpreter* provides methods of translating the abstract *Configuration Specification* into software-readable instructions. Although the interpreter for this particular study produces specific files in the interpreted VBScript language for the Digital Project parametric tool, its methods address the fundamental requirement for interfacing with any other tool as a means of representation and production of well-formed parametric models, intended as models that can consistently support geometric variation and manual editions if necessary. These methods includes resources for creating the overall assembly, create or insert parts, move or rotate parts, add and set parameter values, link external parameters and reference geometry, and create instances from parts using variety of reference geometry as insertion inputs. This approach based on mapping existing files and interpretation of abstracts specifications, by taking advantage of SysML resources, potentially produces a multiplicity of external geometrical representations of the same definition by implementing the already listed fundamental methods of assembling products that most BIM and CAD tools support.

7.5. Apparent Incompleteness and Continuous Growth

The apparent incompleteness of the meta-model is related to its efficiency and its level of detail that range from single *Parts*, several levels of *Sub-assemblies* and the

overall *Assembly*. That is, its detailed descriptions of either of low-level *Parts*, or high-level abstract *Conceptual Structures* are defined by the need of capturing and reusing rather than the registering every single part. The expert designer does not describe every single aspect of a design. On the contrary, the designer seems to prefer high-level *Conceptual Structures* and major *Sub-assemblies* to describe the composition of the design. To support this argument, Chapter 4 shows unbalanced branches in graph-like diagrams. However, they appear to be sufficient for modeling and recalls. In fact, the designer resolves the descriptions by reutilization of the objects of the meta-model while interacting with CAD and BIM tools.

The results of the implementation of the meta-model show that even though the process effectively contributes to integrating abstract objects into the models, the challenge of consistently assigning constraints and requirements to the parts and assemblies in such a way that they produce an organized arrangement of restrictions that shape the problem requires great deal of previous human interpretation to enable computers read the embedded designer's intentions.

Table 4.6 from Chapter 4 identifies three different level of specification of target-oriented and the failure-preventive requirements: determined, which cannot be avoided; under-determined, which require interpretation; and un-determined, which leave room for the preferences of the designer. Consistently, along the three case studies the requirements mostly belong to the under and un-determined categories. Although the SysML used to create the meta-model of the domain provides resources to capture computable expressions and attach requirements to parts and assemblies, these abstract notions of requirements are highly ambiguous yet. In addition, unlike the engineering design field, in which requirements are essential aspects of the design task, in architectural design, they appear to co-evolve with the solution during the design process. Although case studies show that experts already understand fragments of design problems, the meta-model does not show a consistent hierarchical structure that

emerges from the collection of requirements. On the contrary, they seems to establish local relationships with the parts. Thus, most of them are subject to human interpretation and require refinement and rationalization if they are to become computer-interpretable information.

Nevertheless, this process of building a meta-model is based on iterations and refinements that could lead to stronger rationalization of the application of the requirements. Thus, a continuous process is critical, as it allows gradually adding subcategories as well generalizations that to the meta-model improve the level of detail of the specification of the requirements by supporting the process of migrating from designer's preferences, free interpretation to clear specifications. The specifications should identify the metric to assess the degree of fulfillment of the requirement and the objects that are affected.

7.6. Augmenting the Design Space of Alternatives

Expert designers generate not only one possible solution but several of them. This small population represents possible trends in the development of a design and facilitates our understanding of the design problem when they are viewed along with the preliminary solutions. From the *Design Cognition* perspective, the *Design Space* represents the context in which the designer explores and searches for design alternatives (Goldschmidt, 2006). That is, *Design Space* supports a wide range of representations that designer can explore. From this perspective, the *Instance Specification*, which also contains the attribute values of the object, and the *Geometric Representation* of the resulting parametric *Assembly* belong to the same *Design Space*. Throughout such space of alternatives, the designer is an explorer that builds a search path and evaluates various design states. One of the benefits of this approach is that the non-deterministic definition of the representations supports shifts among design arguments and between states while exploration occurs.

From a *Computational* perspective the results of the application the process of *Configuration Specification* and *Geometric Representation* in the three case studies has proven effectiveness in producing well-formed parametric models of design alternatives amplifying the designer's options. Table 7.1 shows the different level of details of the recall, the inputs that require from the Wireframe, the efficiency of the configuration specification and the resulting flexibility of the geometric models in every case study. While recalling Sub-assemblies and Conceptual Structures, also constituted by Sub-assemblies, produces compact human-readable specifications with a reduced number of instances and creates well-constrained parametric model with some degree of geometric control, the instantiation of Parts produces large lists of instance specifications that complicate the human interpretation and creates over-constrained geometric models. Nevertheless, this last level of detail could be suitable for large assemblies that do not require further optimization.

Table 7.1 Level of recall compared with specification efficiency and parametric flexibility

	level of detail	***Input type***	***specification efficiency***	***parametric flexibility***
Case 1	Single Parts	Reference geometry	Over-populated non-human readable	Over-constrained
Case 2	Conceptual Structures	Shared Parameters, Reference geometry	Compact and human-readable	Well-constrained, thickness and general dimensions control
Case 3	Sub-assemblies	Reference geometry, Shared parameters	Compact and human-readable	Well-constrained, internal sub-assemblies control. Over-constrained, sub-assemblies dimensions

In the specific practice of parametric modeling in architecture, the set of possible geometric variations of a model embeds a Design Space rather than a final solution (R. Woodbury & Burrow, 2006) since every resulting configuration is a parametric structure subject of geometric changes. Having alternatives configurations implies additional spaces per configuration. Therefore, finding a solution for a given problem is not only a

searching process through a single *Design Space* determined by a unique configuration, but also an evaluation of different topological configurations.

7.7. Towards Topological Modeling in Design

Parametric modeling is the dominant technological paradigm in design. Variety of computational approaches that support design tasks rely on this technology such as BIM, genetic algorithms or various optimization procedures. While we have one single configuration we are not questioning the structure of the configuration, but the attributes values that determine current geometry. Producing via meta-modeling techniques a population of topologically different configurations establishes a trade-off between changing the structure versus changing the geometry in exploratory design tasks. In other words, the flexibility of the meta-modeling process enable introducing the notion of topological modeling as a long term research effort derived from this study.

Progress in such a direction of this research are: the normalization of the interface of shared parameters and reference geometry among objects that facilitate the compatibility with various design schemas sharing common instantiation mechanisms; the identification of the fundamental methods required to interpret a *Configuration Specification* for creating a *Geometric Model*; and , even though it requires more research, the identification of the need for methods to evaluate the functional compatibility between parts and schemas. More research is needed to elaborate indicators to facilitate comparing options either topologically or geometrically different by taking advantage of the availability of information of the attributes in the specification.

Finally, the current progress of this research point out towards topological modeling in design and the development of methodologies for configuration optimization as the next steps.

APPENDIX A: Seattle Central Public Library

(Courtesy of © Marc Simmons, 2015)

Architect: OMA / LMN Architects

Structural Engineer: Magnusson Klemencic / Arup

Façade Consultant: Front Inc.

General Contractor: Hoffman Construction of Washington

Façade Contractor: Seele USA - Germany

Seattle, WA, 2004

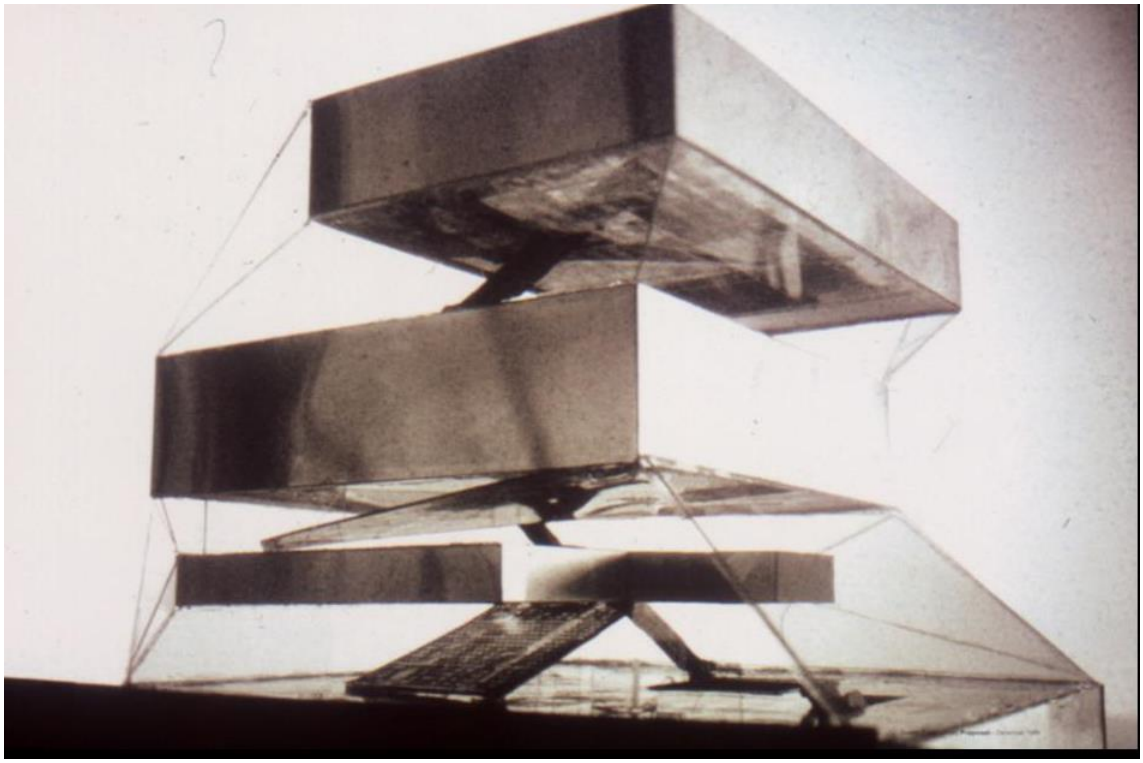


Figure A.1. Seattle Central Public Library conceptual model by OMA (Courtesy of © Marc Simmons, 2015)

[Concept design]

Conceptually wrapping the diagonal grid over the vertical surfaces was very rough, because you didn't need to. Vertical surfaces are conventional column and bean structures, there no actual diagonal structure behind that. Diagonal structure only exist in the interstitial zones, and that was also actually a kind of late development, it wasn't really clear that those elements will have a primary structural function. They (OMA and LMN) wanted the skin elements between the open boxes to be the structure. They just wanted to be that, no columns, all clear span interiors (Figure A.1). The real problem of that is once you take all of that steel and size it for lateral and gravity it has to be fire protected, and if you are going to fire protect all that steel it is 4.5 million dollars, 5 % percent of the job… and of course the density of the steel could be much larger than it is... So what came out of this was the recognition that the building had to have columns, and once you put all the columns there and you start analyzing them and trying to find a

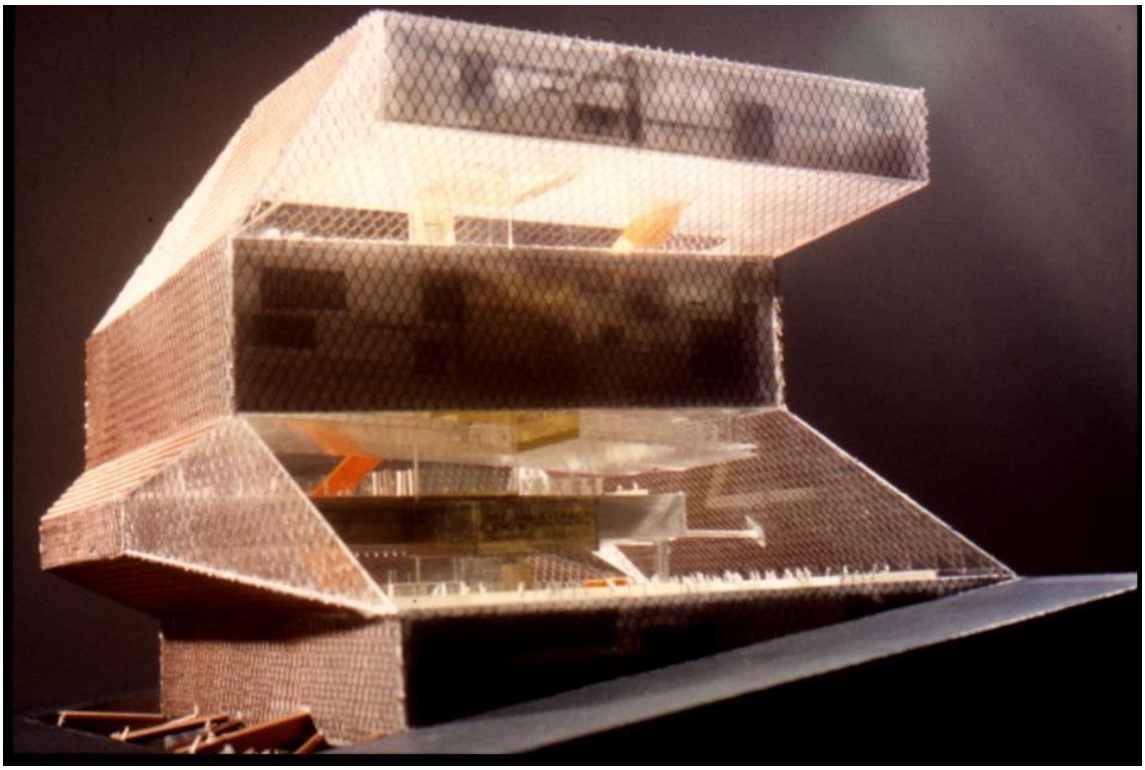


Figure A.2. Diagonal grid façade model (Courtesy of © Marc Simmons, 2015)

lateral size of the system to restrain the building, the columns become monumental, gigantic sections besides massive moment connections or huge gigantic cross braces… So, the question was then, and it took long time ride the way. I don't have images, but just to give you an example, during schematic design the entire skin of this building was a tension system. For about six weeks was obsessively done tension system. … It took a little while to work up that scheme and get the pre-stress on the cable to support fabric over the 134' spans, and you still have snow loads in Seattle. What do you get? You get basically a pre-stressed force on the building that is equal to 40 % of the seismic loads on a permanent loading basis… and it also entails 5 million dollar in steel complexity. Tension structures always come with a cost since perimeter conditions must be very robust. So, that was killed from the cost stand point of view.

The idea that the exterior diagonal grid then come into the picture as a diagonal shear grid that would serve as lateral system to the building (Figure A.2) while there

Figure A.3. Steel façade taking lateral loads (Courtesy of © Marc Simmons, 2015)

were steel column grid system in the building was actually a very late development. Imagine developing it in the last part of the schematic design, when we were beginning to consider tender the façade... The timing was quite tight. Do you see the uncertainty in the process? The main structure must be fire protected in the connections with the slabs. The boxes themselves, it was easy to brace the frames, you have a couple mega breakers, gigantic diagonal steel columns specially located, and then we have this diagonal grid steel. The Diagonal grid steel is doing double labor, Imagine we don't have the grid, but I still need to put cladding over 134' span. It needs a massive backup structure, huge, which is actually equal to the size of the bracing structure that would have to be. Putting the bracing structure and using it structurally for the lateral stability of the building (Figure A.3) puts tons of steel inside of the rest of the structure.

Figure A.4. Seattle Central Public Library natural lighting (Courtesy of © Marc Simmons, 2015)

Then, the façade is important, but actually it is for free. So, using the building geometry to extract tonnage, put it into the skin, and then it is the wind load resistant structure for the skin. Then I can pay 35 dollars per sf. without having to think in the extra 75 dollar per sf. I would have to put for the secondary steel backup structure... Another obvious factor as well is that if you see the building you will notice there are no blinds. Most of the libraries like which have this percent of the glass on it will all have one hundred percent of motorized controlled blinds in there... In Seattle they are not there, to do diagonal intention trapezoidal blinds could cost $35 sf. just for the blind for the interior. The fact that they are not there is attributable to the density of the steel (Figure A.4). Also the oblique grid becomes opaque and that lateral opacity, oblique opacity, combined with the presence of the adjacent towers is the reason why we don't have blinds. There are so few times a year in so few locations in the building where you

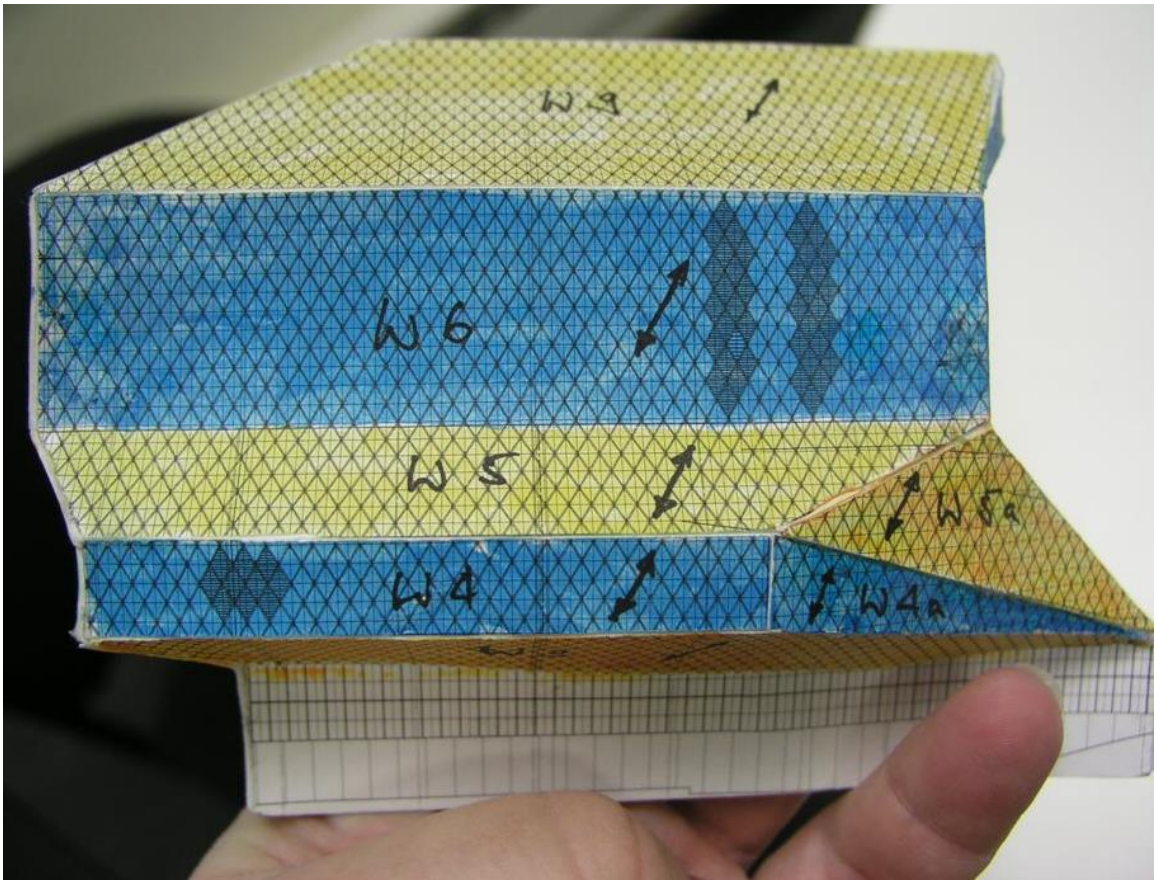


Figure A.5. Façade dominant directionality (Courtesy of © Marc Simmons, 2015)

can also be static when you actually feel uncomfortable, and if you are uncomfortable it is just a small portion of the façade and it very reasonable percentage compare to 5 million dollars blinds.

[Building Surface Geometry]

The black arrows (Figure A.5) are really important; this is the dominant directionality of the façade. Steel in one axis is linear and the infill is stitched in, the mullion is linear in one axis and has an interstitial transit that interlays the other axis. The brackets only happen in the continuous mullion, the brackets only seat on the continuous steel.

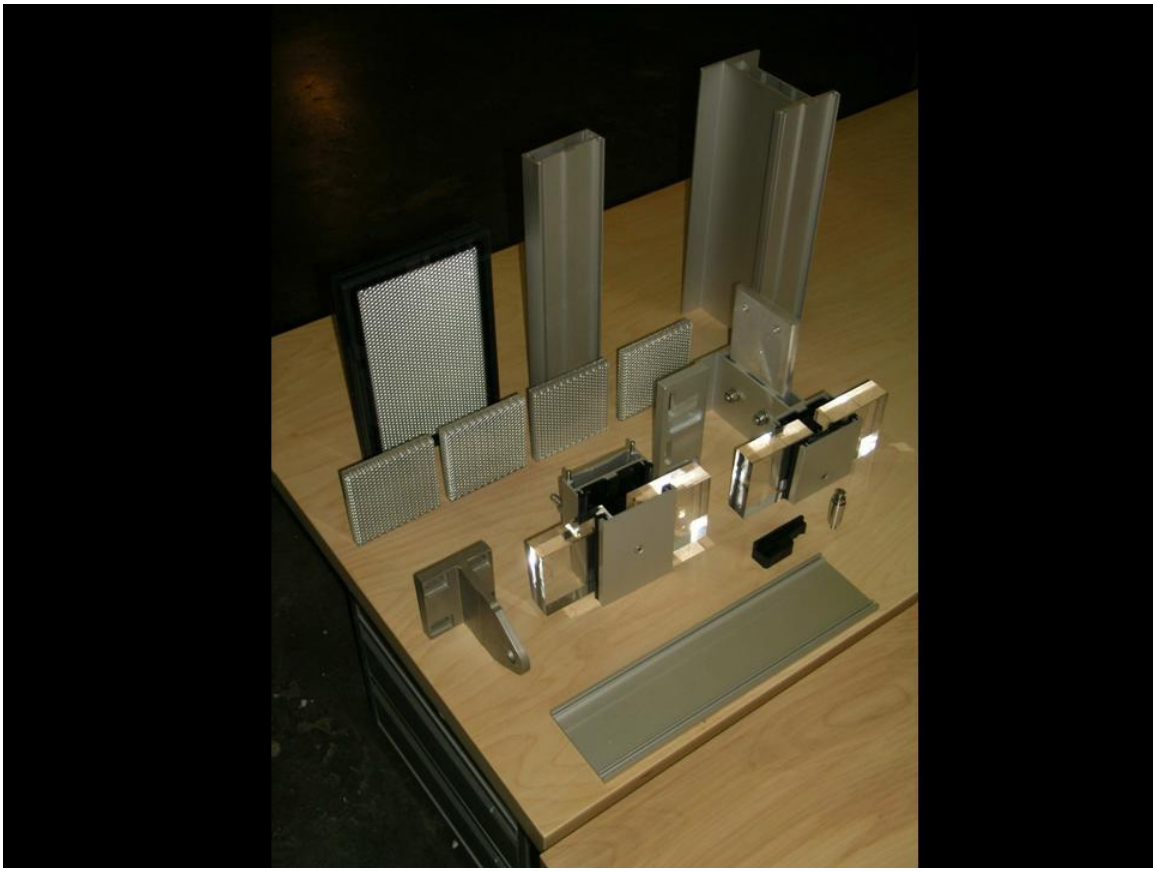


Figure A.6. Façade components (Courtesy of © Marc Simmons, 2015)

[Façade components]

(The) basic mullion extrusion is 1" ½ thick that goes on the slope areas on the building. It only has 4' spanning capacity. Every 4' there is a bolted connection back to the flange of the steel, which is 4" wide... Between this (mullion) and the steel there is approximately ¾". The front parts of the two types of extrusion are identical. So, everything from that forwards is identical for both curtain walls. But because we don't have diagonal steel behind our vertical curtain wall we need to span floor to floor, and in this case we are spanning on the diagonal. Diagonal span from floor to floor is 17', which is quite long. The "I" shape box mullion that has been engineered to span 17' on the clad. This could have been a box, it could have been a "T". Obviously, it was chosen an "I" shape because it is conceptually similar to the steel (Figure A.6). Even

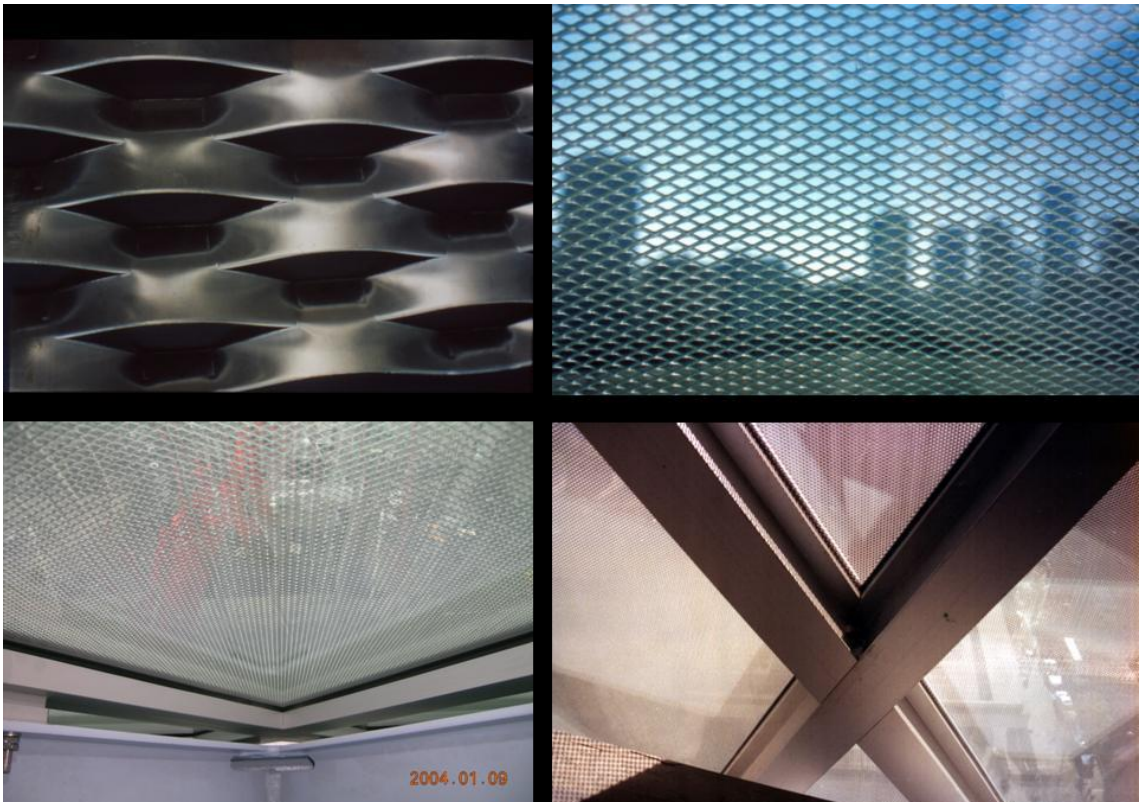


Figure A.7. Glass metal mesh (Courtesy of © Marc Simmons, 2015)

though it is a self-supporting façade, it is not really the same than the slope curtain wall. Aesthetically there was a decision that the way light plays on the "I" shape piece of metal is beautiful, as you will see. And it was designed consistent rather if it is vertical or horizontal. The slope were paint steel and the vertical anodized aluminum

[Metal mesh]

Industrial stretched metal (Figure A.7). We visited the factory… we asked them could they adjust the rate of holes. So once you punch it, the degree of pull governs the degree of aperture in the mesh. We wanted certain meshes to be very tight and other ones to be quite open. We were, actually, modulating it through micro folding it. The aluminum mesh is encapsulated within the glass. That is a natural anodized finish.

Figure A.8. Façade face cap holding the glass (Courtesy of © Marc Simmons, 2015)

[The face cap]

This extrusion is the face cap (Figure A.8). What is very different about this curtain wall is that the face cap is that it is actually holding the glass on the building. Most often is a sub trait cap which has a snap cap over it. So, the only way that this cap can holds the glass on is with a screw that goes through this piece of material. So, this piece of extrusion is actually milled with cumbersome holes at very precise locations. The screws are exposed outside of the facade. Every single screw is CNC precision located according to the design specification. The reason why this was done is not just for technical reason, because it is more expensive to do this than just put the cap after the fact; it is because it gives much greater precision. Every single hole in the extrusion (the mullion) has to absolutely match with the drilled hole on this extrusion (the cap).

Figure A.9. Connection blade that supports the cleaning crew (Courtesy of © Marc Simmons, 2015)

[Blade]

So those loads have to go back to the primary steel, meaning, it is penetrating through every layer you see here (Figure A.9). Essentially, that blade has to go all the way back to the extrusion through every single layer between the glass and come out the face of the cap. Every penetration of the cap must be water proofing.

[Anodizing]

If you are doing all this drilling and drilling on the extrusions (Figure A.10) the cheap way to do this is anodizing first and not afterwards... In this case they did the drilling first, sending it to a third party to anodize, getting all anodized and getting all the drills completely protected and then bringing back in. Logistically organizing every single component directly, doing all the sub assembling and getting it to the site.

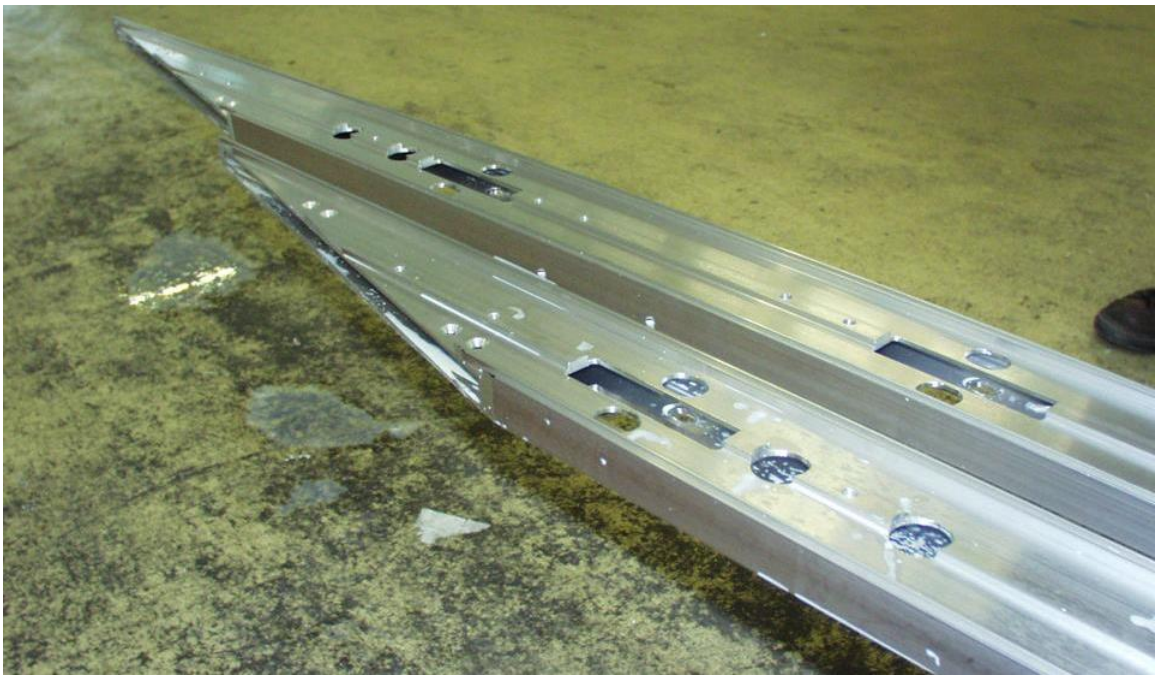


Figure A.10. Drilled aluminum extrusions (Courtesy of © Marc Simmons, 2015)

[Glass customization]

I should mention that the glass has both Argon and Krypton gas depending on the location on the building... If you are going to create a custom piece of glass, how are you going to get your certification? The testing is not that sophisticated to do the completely assembly, it has to be component based, so what they do testing individual layers, that information goes to a data base and a software integrates customized layers built up with all those properties and basically evaluate the aggregate's performance.

[Mesh distribution]

Only the 50% of the building is covered with glass with mesh integrated, which is only in Faces that face South and West all the North, West and down wards faces are actually low-E coatings with pure glass. There are large regions of the building that not have meshes, the mesh is only there where it is needed. Basically with the low-E coatings we started with 0.3 – 0.4 SHGC and the mesh got it down to 0.17. That 0.17 was targeted to the areas with larger Solar Heat Gain.

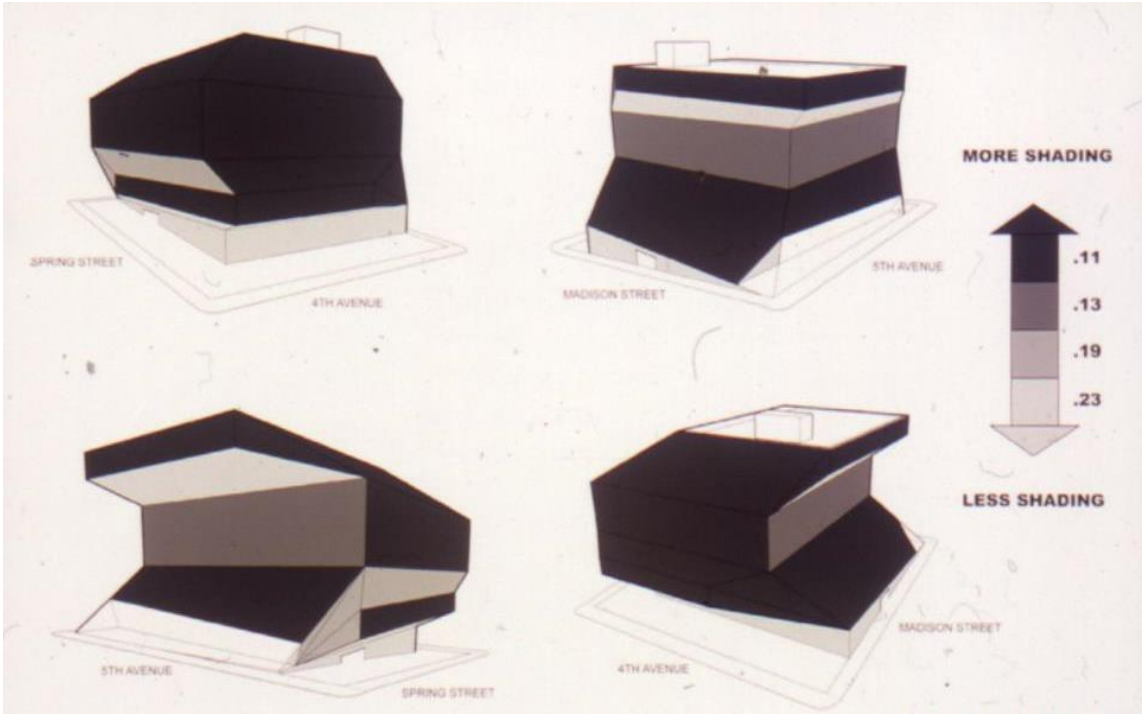


Figure A.11. Metal mesh distribution according to orientation and slope (Courtesy of © Marc Simmons, 2015)

It is actually a triple glassed unit in these locations, the other two unit of glass are only 2 mm spaced where the mesh is, and there is another interior space with gas and low-E coating. So, the metal mesh performs as a brise-soleil and is opaque, only certain amount of energy is going through no matter what (Figure A.11). The mesh is either reflecting or absorbing the energy… Because it has a relatively lower thermal mass it has high thermal expansion, but this material is so white it is also rejecting a huge amount of heat. So it is actually not expanding very much. What we are getting is 35% cut from benchmark immediately... The performance of the glass is calculated in 90 degrees normal to the surface... , but give the geometry of the mesh curving, when it moves slightly it becomes completely opaque and when it get more lower in the sky, of course, it becomes more open. But when you look at the solar intensity in Seattle occurs only in the summer mid-day when you get this very sun. In which case the building has much better SHGC than is in the energy model, and I've been told that it is actually over performing.

Any heat that is absorbed by the mesh, thermally, what happen is you get re-radiation, part of this radiation goes in part goes out. With the low-E coating in the interior plus the gas, basically mitigate the radiation, and it pushes the radiation coming-in back out again. So, almost all the energy gained by the mesh goes outside of the building. That is why it performs so well. And then the solar radiation getting though the apertures in the mesh is getting though the low-E coating itself, which already has a SHGC of 0.3 or 0.5 which cuts 65% of what is getting through. That is why we are getting down to 0.17 effective SHGC in 90 degrees to the glass, which is so high performance and is also so beautiful.

Because of the tridimensional nature of the mesh and the non-standard geometry of the mesh the PVD can't just be a single sheet of 1.7 mm in either side of the mesh. (It is going to be) micro laminates which basically have the same thickness but use a really thin sheet of PVD. Essentially they were more pliable and were melted into the surface of the mesh. … It made the job viable with the mesh. The Seattle Library said "OK, this is possible." We also agreed it is beautiful; the quality of light is beautiful. The kind of diffuse effect you get into the building is also beautiful... The light quality in the building is much better... Back to this tinted glass…, tinted glass filters the color, it changes the color in the interior of the building. So, the filter of the micromesh, of course is not filtering the color of the light... We get it aesthetically, technically, let's pursue it. However it cost 3 million dollars more... Could the design team be authorized to rise funding? If you didn't have all that certainty somebody could say can we look other solution? But because it was so well documented the team was allowed to do that. It was based on this notion of selling the quality of light in the building, which is really awesome because you are not saying you (will) get put your name on the lecture hall, you are not buying a piece of the building. It appeals to the people in Seattle to be buying something that was actually performance and experience oriented. It is really cool.

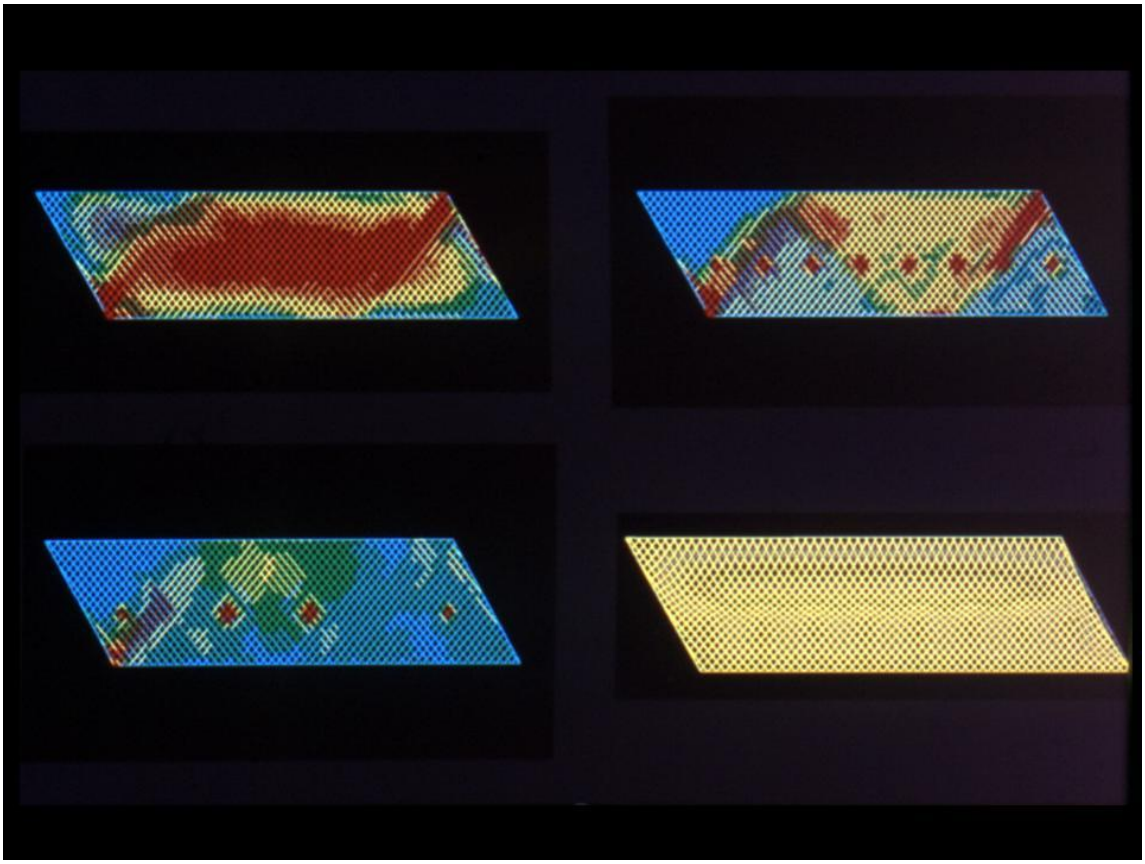


Figure A.12. Structural analysis diagrams (Courtesy of © Marc Simmons, 2015)

[Steel Design and Construction]

Of course, we were looking at rectangular piece of glass nested into a diagonal grid. Eventually the diamond gird came in and it looks more beautiful. These are just these typical rhetorical stress diagrams from structural analysis software, which resulted in those amoebas as we call them (Figure A.12)... And you will see why it matters, because in this area, under dead and wind loads we needed double steel depth in these areas to 24", and that is what we get here basically we used that shape to add double depth steel to the back of the existing grid… Steel lattice was welded up into frames approximately 10' wide by 50' long. Every 3rd of 4th piece of steel there is a bolted space connection which also has directionality. Those lateral frames basically expand from top to bottom. Came in a flat bed and came into the building through cranes, and they are all basically welded.

Figure A.13. Façade differentiation according to structural requirements (Courtesy of © Marc Simmons, 2015)

There is another area I want to highlight. You don't have the equal density of the mesh of the steel (Figure A.13). We got three diamonds long among supporting brace steel. The reason is this surface is not contributing to the lateral stability of the building. Certain surfaces are lateral stability and when they are not, we take as much steel as we need. This is just a secondary steel structure assumable for wind load over the cladding, not for the structure of the building. So, this is building structure, this is not. And that become totally legible the building. Actually, you can walk through the building and start reading what is doing what, why. Certain areas are doubling up because they have heavier forces, other areas are single, and in other areas the steel has been subtracted. And this is why is important that the curtain wall is also directional. The brackets are all going the direction of the steel continuity, and the other cross spans don't have a bracket in that other span because sometime there is no steel to connect it.

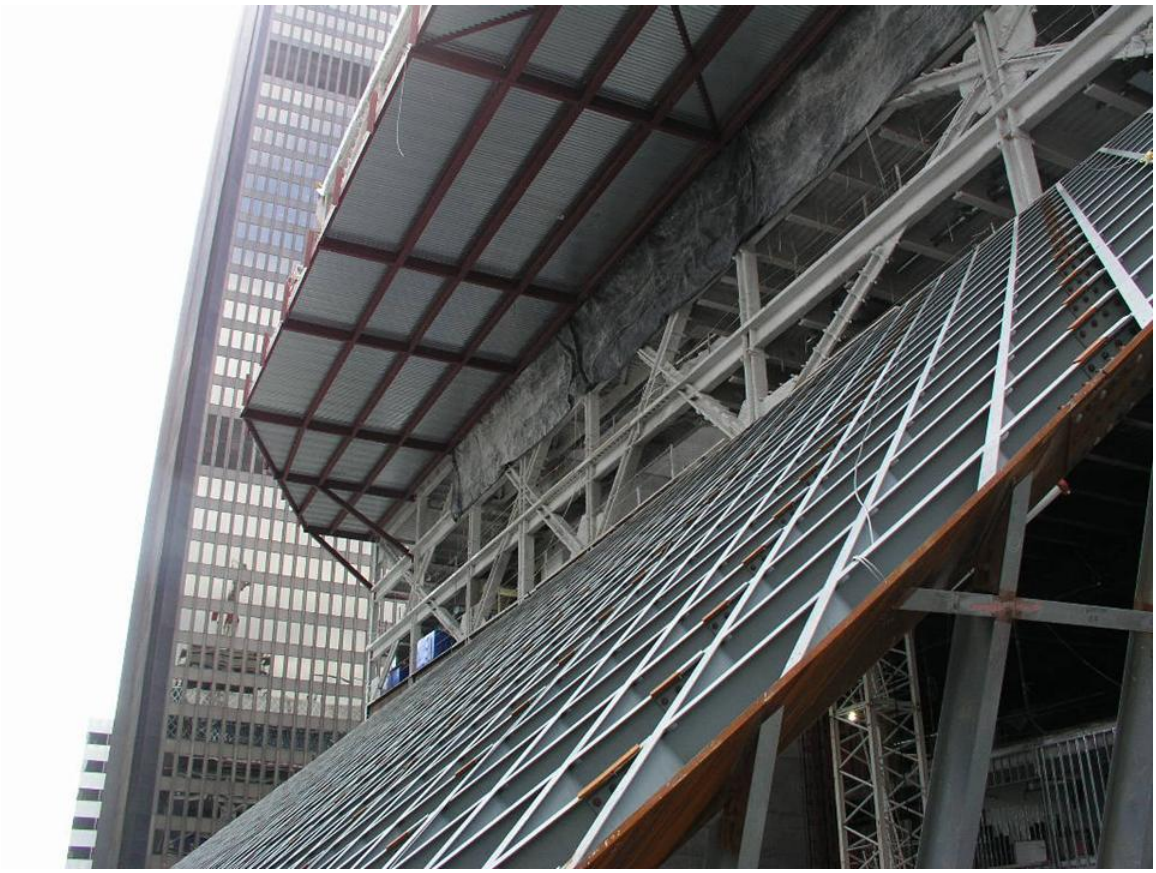


Figure A.14. Façade ruled surface (Courtesy of © Marc Simmons, 2015)

30% of the entire surface areas of the building are twisted (Figure A.14), and actually 30% of the panels are cold form curved glass panels. Because every time we have a diamond (the upper vertex of the panel) is out of plane by 3/8", and we use the cap to pull it down and induce the curvature of the glass. It is actually very similar than Frank Gehry's cold form to twist his panels. The reason it comes about though is that because of the boxes are offset geometrically in two axes, the one in the middle is also a parallelogram. So (the bottom) surface is actually sloping out of plane. And if we try to connect the points below, by definition is a twisted surface. So, the steel was always organized in one axis, which also correlates to the dominant directionality of those black arrows.

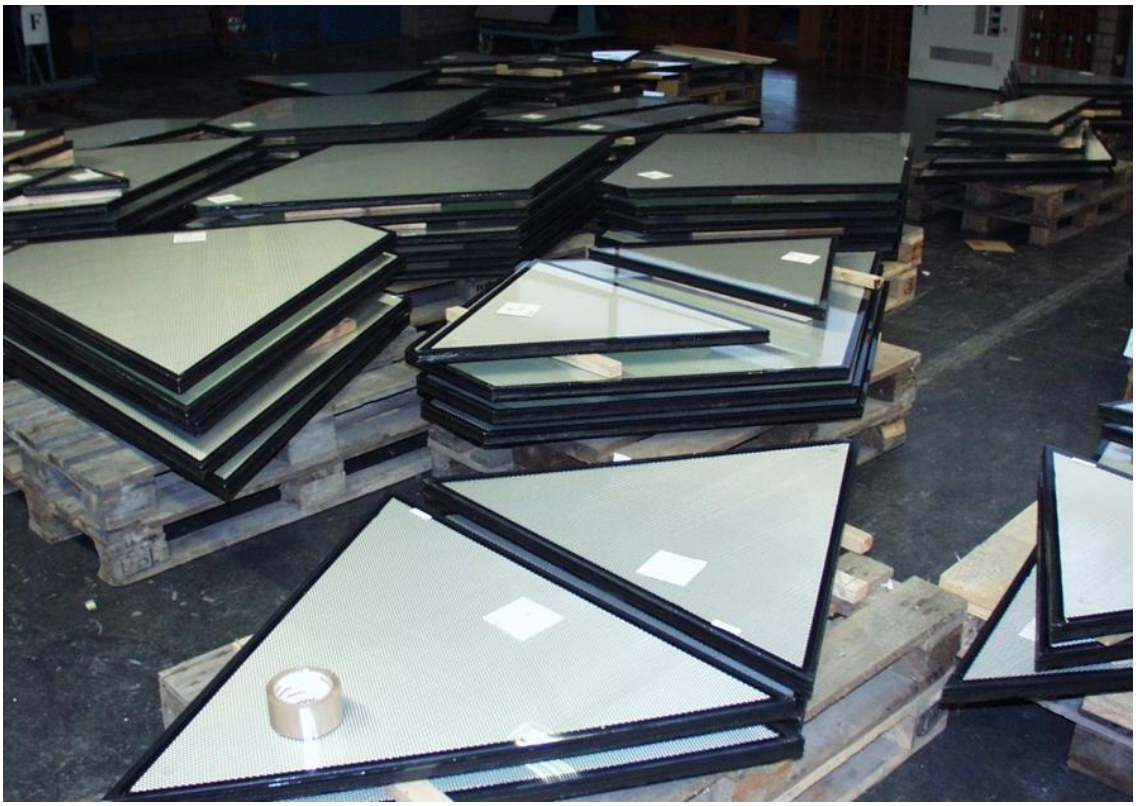


Figure A.15. Custom glass panels (Courtesy of © Marc Simmons, 2015)

[Glass Panels]

65% of the glass panels on the Seattle library are actually non-standard. Even though the infill panel is forced to the scale of 6x7 panels across the entire facade, the number of edge trimming and small panels is actually huge. You get all kind of bizarre triangles (Figure A.15). Imagine the logistics that has to ensure that the piece of laser cut aluminum mesh is always oriented vertically. It always must be cut in the correct orientation and always located in the panel in perfect relationship with that geometry.

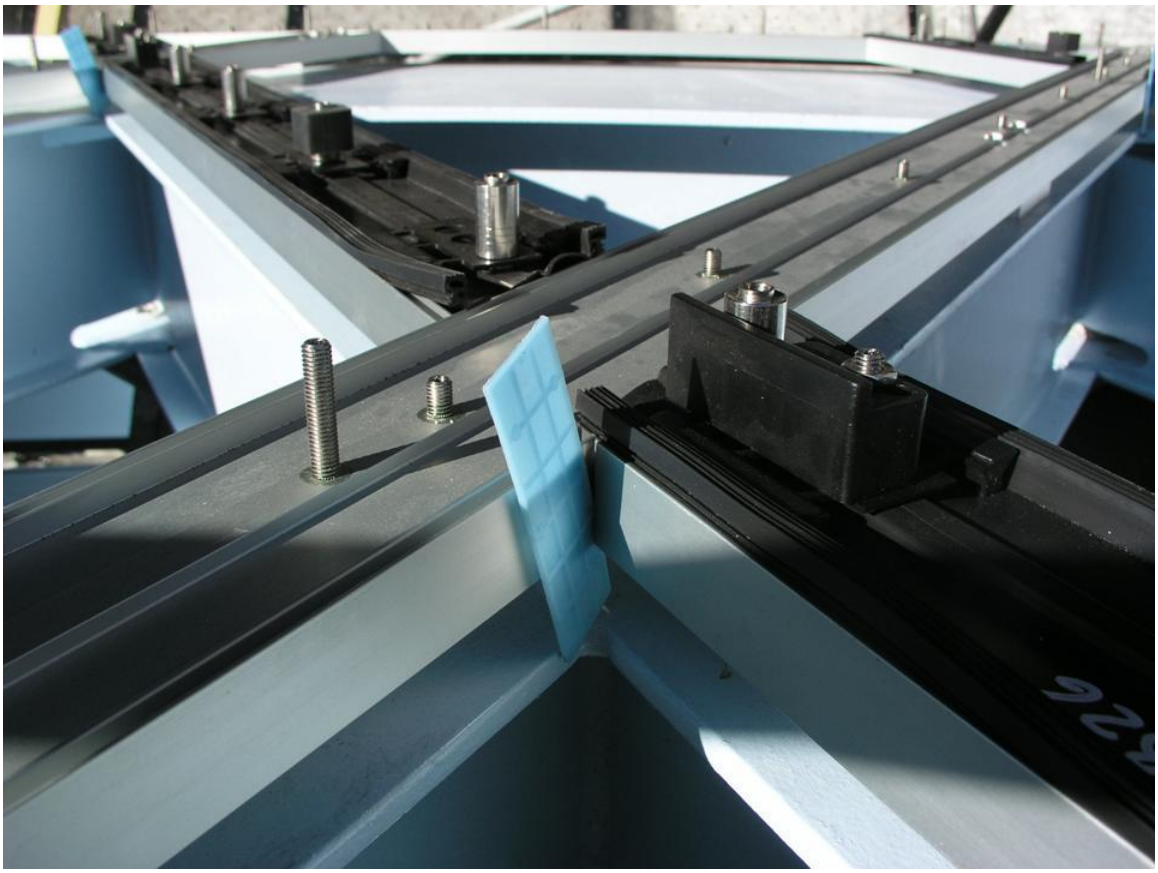


Figure A.16. Spacers and gaskets (Courtesy of © Marc Simmons, 2015)

[Spacers]

This is the panel spacer. There are plastic spacers, aluminum spacers, stainless steel spacers (Figure A.16). Stainless steel has one third of the thermal conductivity than aluminum. It is better thermally, but it is not as good as plastic spacers, but the plastic spacers are subject of long term deterioration. So, stainless steel spacers are the most reliable long term solution. The pre-assembling and indexing of all the components is incredible. The gaskets are actually going on to the extrusions. There these things... Plastic blocks, and little plastic gaskets.

Figure A.17. Façade surface cap holding glass panels (Courtesy of © Marc Simmons, 2015)

These diamond shape panels of glass are aligned correctly. There are two little plastic sides blocks and in the top there is little pin bumpers that have a soft rubber that hold them in place. The glass is allowed to rotate within the frame, and when the frame distorted by the movement of the building, the glass compresses the soft rubber and gets back to its position and the rubber finds its own geometry. A piece of hard plastic is perfectly indexed to the distance between the aluminum and the glass. What happen in normal caps is that the guys of the field are screwing down the screws into the screw chase. Different screws are actually with different levels of engagement. In the gasket that is been compressed there is differential levels of compression in every screw (Figure A.17). What you get very often, almost always, is a gasket with geometry which looks a little bit bumpy. If you put a snap cap on top, there is an induce curvature. If you at the reflection on that cap, it is discontinuous because there is non-standard geometry

Figure A.18. Snow guards (Courtesy of © Marc Simmons, 2015)

on that cap. If you look some other buildings that are a little bit cheaper, you often see those kind of lines which don't have perfect lines of reflectivity.

[Snow accumulation]

There is snow accumulation to a certain point. The snow changes the U-value of the assembly, because the snow is insulating. Then, the melt point migrates, the heat inside of the building goes to heat the top part of the glass and then all the snow start to melt, all get lubricated and then in one point snow cap basically goes. So we have those gigantic snow guards (Figure A.18), the snow basically hits those snow guards, stops, and falls into those massive snow melters which have four lines of heat tracing that melt the snow as it is falling in.

Figure A.19, Glass installation (Courtesy of © Marc Simmons, 2015)

[Glass installation]

12” steel, all painted, ¾” gap, 1 ½” aluminum extrusion, the gaskets, everything, drilling, is preset in the shop. There is not bracket along the discontinuous aluminum extrusions. Only the continuous pieces of aluminum have brackets... They put the glass on, and then they put temporary caps (Figure A.19). The way the systems works, the eventually take them off (two temporary caps), they hold the glass in two opposite sides, put the continuous aluminum cap extrusions that run in the dominant axis, and then that holds the glass in temporary two sides, and then they can put the stich piece and then they can take off those temporary restrains safely.

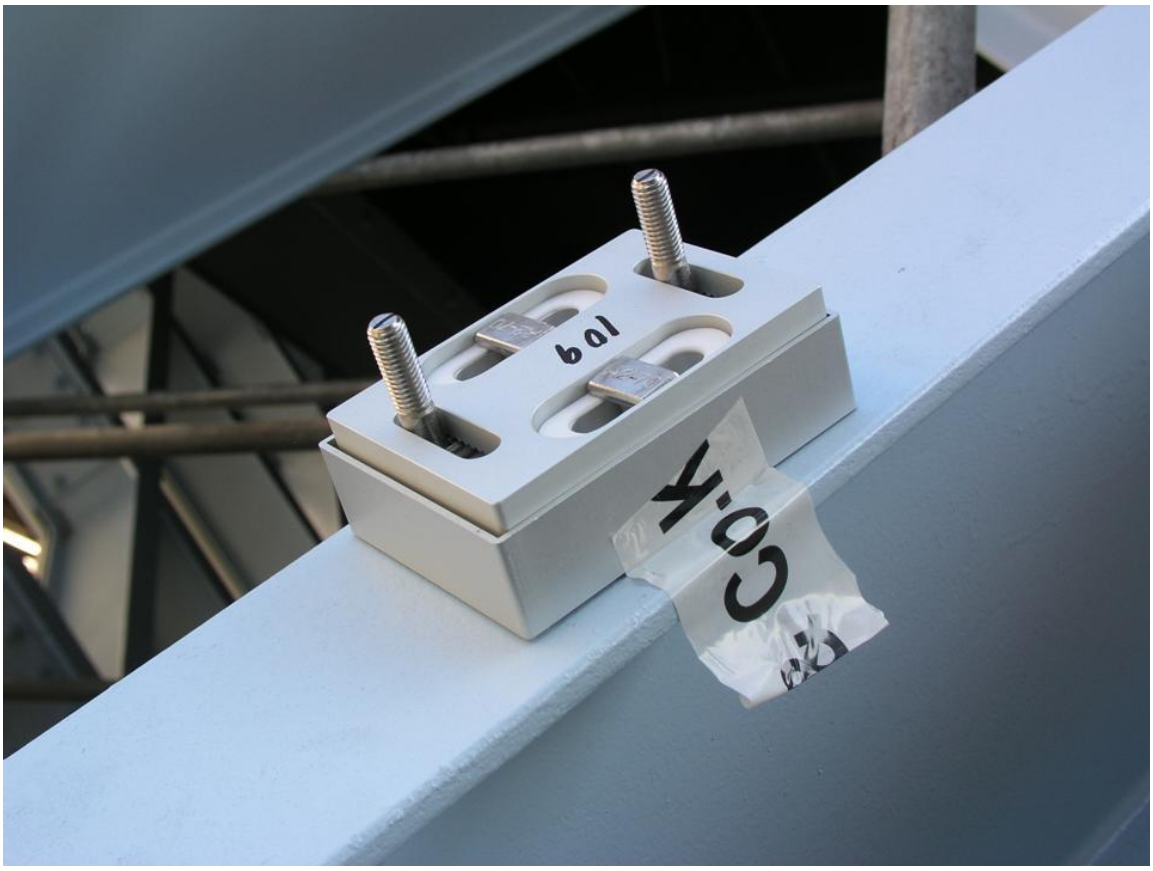


Figure A.20. Curtain wall brackets (Courtesy of © Marc Simmons, 2015)

[Curtain wall brackets]

Here is the bracket. It is basically a block of aluminum (Figure A.20). It has linear slider holes plus minus ¾" adjustments. The contractor offered post drilling all the holes into the steel on site. The owners said no, they preferred CNC machining. Then what happened in the other axis we have similar arrangement than this, but in the underside, where this two tee bolts go up and basically pin up the extrusions. It provides ¾" of tolerance in both axes.

Figure A.21. Vertical curtain wall upper bracket (Courtesy of © Marc Simmons, 2015)

[Vertical curtain wall upper brackets]

These brackets only provide perpendicular wind load resistance, but they are laterally and vertically flexible (Figure A.21). They support the upper portion of the vertical curtain wall. They have to handle the vertical translation… It is a rigid connection that just moves up and down with the thermal expansion of the curtain wall.

Figure A.22. Floor closure (Courtesy of © Marc Simmons, 2015)

[Floor closure]

We still need to provide floor closure and a partition, and we don't actually break the glass. We have little extrusions that have basically a gasket that goes into the glass (Figure A.22). They have one slider hole with this little reveal on the side. And that accommodates the geometrical change on this curtain wall. That is what you have to do, as soon as you start setting those rules and system logic, how do you detail them? Otherwise those infill extrusions will constraint the whole façade actually locking it up preventing its natural movement.

Figure A.23. Aluminum tape, primary water proofing barrier (Courtesy of © Marc Simmons, 2015)

[Water proofing]

The primary water proofing barriers of this façade is a tape, aluminum reinforce tape (Figure A.23). That stainless steel platen is there to support the tape... This aluminum reinforced tape unrolled onto the glass and just forms this continuous barrier. And the tape is flexible and handles all the size difference in the glass. So the façade has three layers of water proofing, the face cap has silicon sealing which is really there for cosmetics. It has the gaskets which are compressed and then it has the silicon at the corners between extrusions. Then it has the aluminum tape below it, which is really protected by the cap. Then has a secondary draining channel inside which is a continuous extrusion in the gasket. The gasket has a channel through the entire facade that slopes out and drains into the gutter…that has its own secondary drain build into it. It has three lines of defense inside of the 2" curtain wall.

Figure A.24. Vertical curtain wall bottom bracket (Courtesy of © Marc Simmons, 2015)

[Vertical curtain wall bottom bracket]

This is the base bracket for the vertical curtain wall (Figure A.24). There is a flexible bracket above, the massive deadwood bracket. There is a secondary chunk of secondary steel tied into the base building structure... The curtain wall bracket is locked to two mullions at the same time. That happened at the base of every vertical curtain wall.

Figure A.25. Slope Curtain wall bracket (Courtesy of © Marc Simmons, 2015)

[Slope Curtain wall bracket]

This is a secondary steel connection to primary steel (Figure A.25). Every steel diagonal has a vertical slider connection to allow them adjusted to each other

[Installation Sequence]

It is a top down installation. The corner panels are very interesting (Figure A.26). The resolving looks totally non-standard... We set up a series of rules where the vertical edge lines have an inset reveals, and the horizontal are simple bends, and there is an offset dimension to the first cap. They produce a very funky pieces of prefabrication. You couldn't invent this stuff. They have to come up from a rule based module. These pieces are kind of resolving all that. Eventually you get this welded corner aluminum… They are anodized welded aluminum sheet metal... You can see the quality, is pretty good looking, considering the scale of the building.

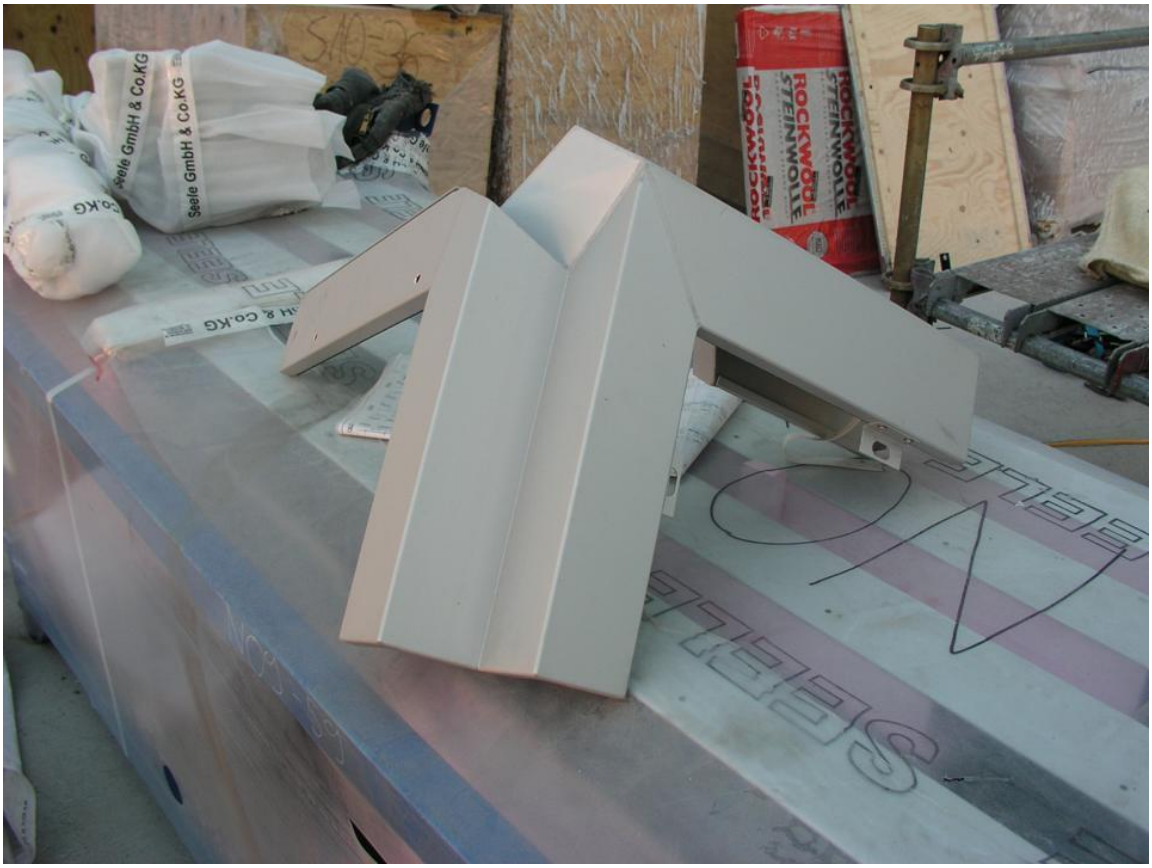


Figure A.26. Custom corner panels (Courtesy of © Marc Simmons, 2015)

[Water proofing tests]

There is a theoretical requirement to test 100% of the building. But we started at the top incrementally decreasing the testing requirement, and it passed 100% which pretty amazing. And then they (the inspectors) say water test the corners, you don't need to test the under slopes. They actually release us to test the whole building.

[Louvers]

They (the louver rhomboidal panels) were basically inputs…

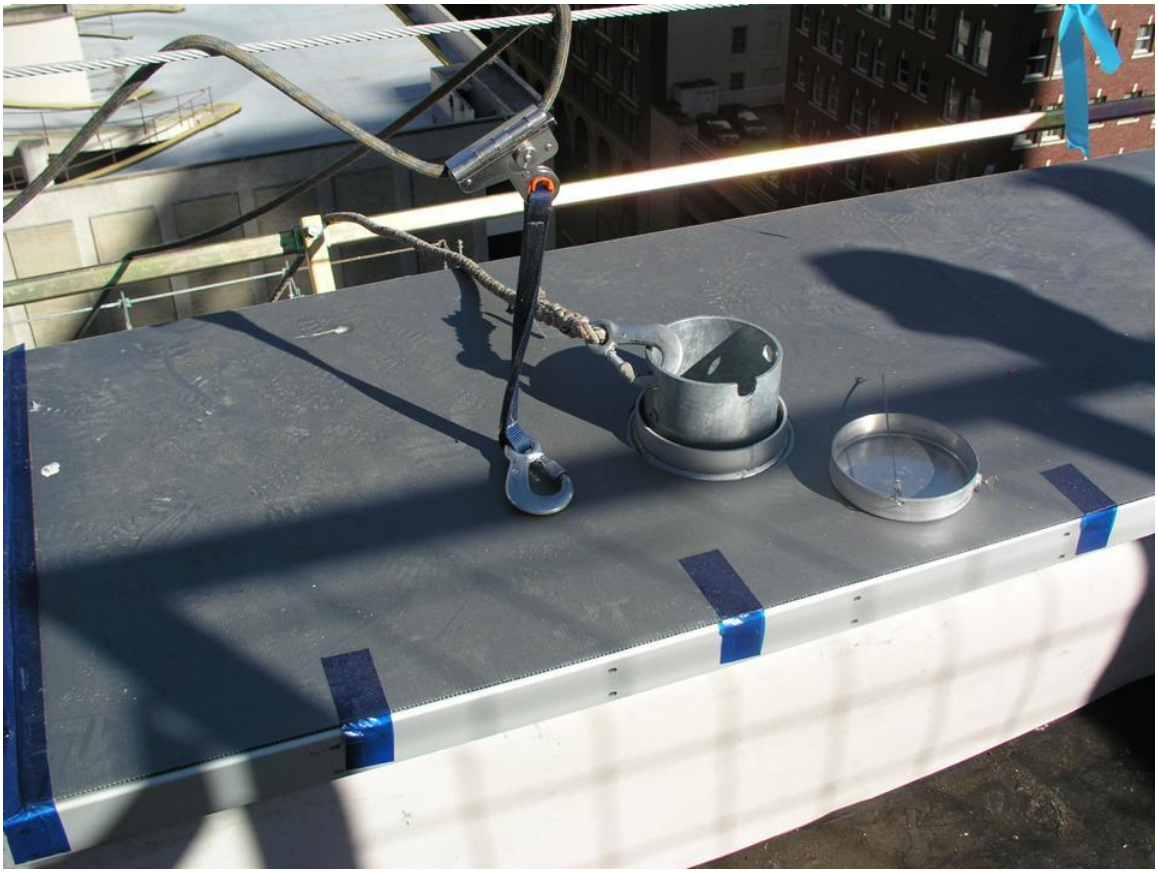
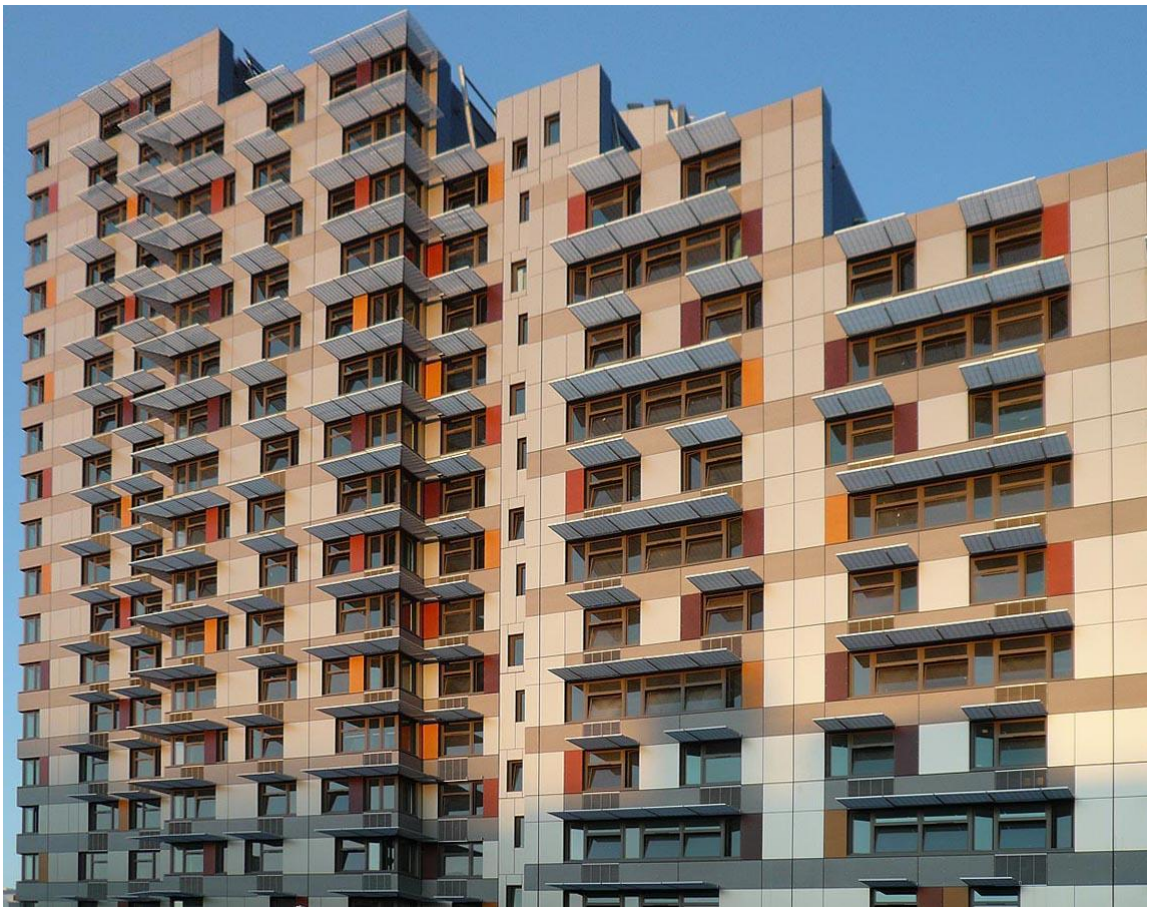


Figure A.27. Cleaning anchor points (Courtesy of © Marc Simmons, 2015)

[Maintenance]

Cylindrical tie up point on the roof and a hook provide access for maintenance (Figure A.27).

APPENDIX B: Via Verde Residential Building



(Courtesy of © Marc Simmons, 2015)

Architect: Grimshaw Architects / Dattner Architects

Structural Engineer: Robert Silman Associates

Façade Consultant: Front Inc.

General Contractor: Lettire Construction Corp

New York, NY, 2012

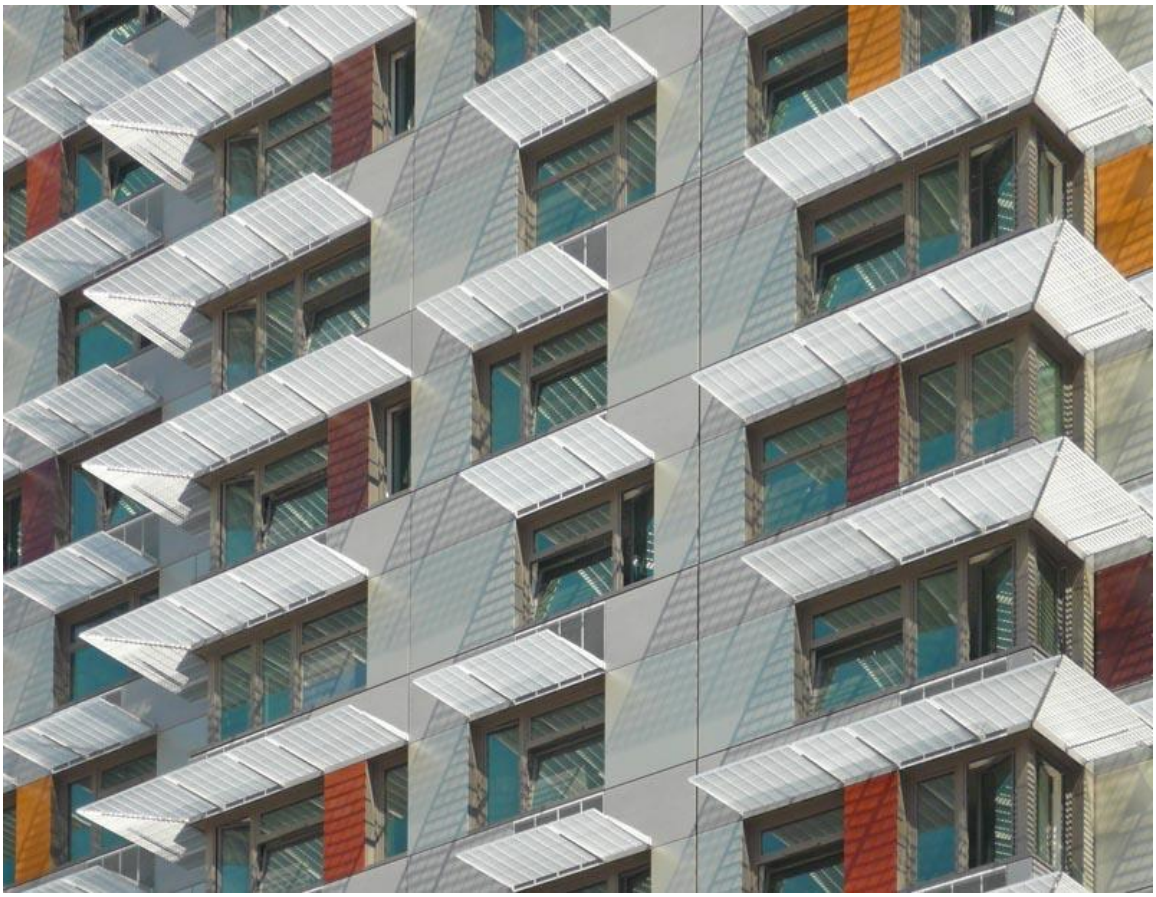


Figure B.1. Façade mega panels (Courtesy of © Marc Simmons, 2015)

[Mega panel]

Everything is a single mega panel, and brise-soleils are bolted in and the balconies are integrated at the site. There are a limited number of panels that can be stuck in a truck. Normally what we do is just bring panel s out and as soon as you pick them, a crew installs the brise-soleil, and the whole panel is lifted up to the building. So, you start to see the joint perimeter. These joints are slightly larger. They are larger because they need to handle thermal expansion... What panels do, because they are so large, they do expand.

In a mega panel the door is half the way in a one panel and half the way in the other panel. The door actually install after the fact. That was a kind of macro decision that had to be made. The reason is that the head of the window is below the floor slab.

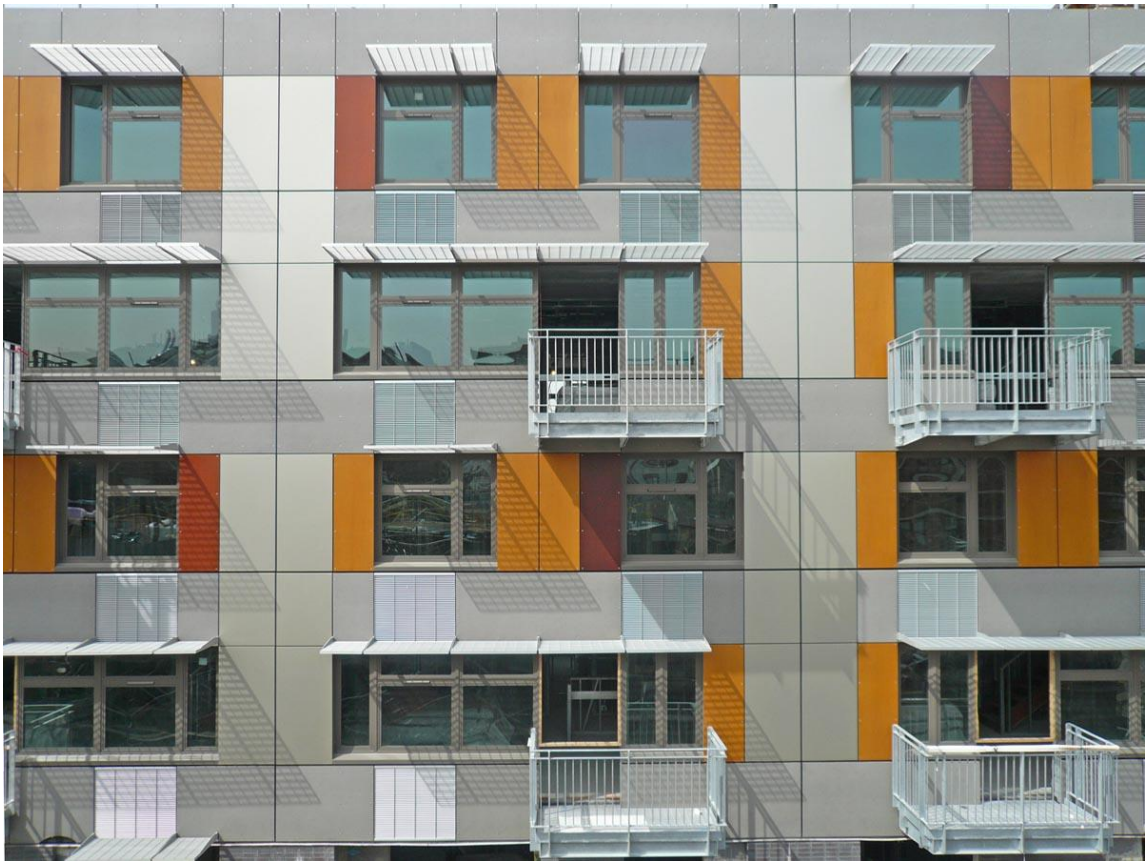


Figure B.2. Balcony doors between mega panels (Courtesy of © Marc Simmons, 2015)

The high you hang the panel from the floor slab which actually doesn't go up to the next floor slab. If the panel stops before the slab you can wreck the lower panel, you have to put a crazy temporary brace, bringing the next panel to lock it in the previous one and then build the next one with temporary braces doesn't make any sense from the construction sequence stand point. What you really need is your first panel hanging from the slab and the next one seat into that with interlocking pins and you lock it back in (the upper slab), and you are good to go. And when you leave work at night, you can leave the panels at work because they are stable, and you come back the next days to continue the sequence. The fact that the door runs across the stack joint which is half way of the room was actually a tradeoff (Figure B.2). We had to say that that as a technical detail we had to figure out in the macro picture of the project.

Figure B.3. Window supporting stud (Courtesy of © Marc Simmons, 2015)

[Window]

The window has to be supported by some kind of beam, right? But this window is kind of hanging in the space. It is not seating on the panel below (Figure B.3). There is actually a completely floating movement joint between those two panels (upper and lower panel). There is, actually a very slender tubes steel beam at the bottom of that window with the hanger rod that hangs over here (between the door and the window), and the right part and the left part of the panel are actually two independent structures that don't have lateral continuity across (the door). And this part up here (the horizontal upper part), because this is an aperture it is built as Vierendeel beam. So, there is actually a steel tube header and vertical steel subdivisions that can't be braced where there is an aperture, and can (be braced) where there is not.

Figure B.4. Panel structure based on studs (Courtesy of © Marc Simmons, 2015)

[Panel Structure]

The entire structure of the panel is a Vierendeel beam with two hanging portions to the right and left leaving space in the center for the door. Every lateral portions are made by a module, a lower steel member and a hanger. The edge of the slab crosses right in the middle of the Vierendeel, where it is welded on to the slab.

(The structure) it is complicated, but really it is not. We would really say I have to have windows, and there constructability reasons, and accepting those constraints yields panels like that. But that panel was made on galvanized plate cold form sheet metal (Figure B.4). You can make the diagram (of the panel) in five seconds, cut that out in thirty minutes and actually do structural analyses in half a day, and then do some details on it, and go through and discover all the problems. But now we know that this is the approach we (want to) take... The entire installed cost for complete design,

Figure B.5. Mega panels brise-soleils (Courtesy of © Marc Simmons, 2015)

engineering fabrication, shop drawings, installation and water proofing is $94/sf for everything, including the balconies, the brise-soleil and the rain screens (Figure B.5).

The regular approach to layering the panel, was infilling the structure with the insulation, interior finish and exterior water barrier But this kind of building would not be legal, because the U value is not being achieved. Placing the insulation in the outer face of the frame structure opens things up. The city is trying to mitigate plumbs impacts. Then, if you put the insulation outside it must be class zero insulation, which means mineral wall, because if you put foam polystyrene outside of the building it just burns, you can't do it. You can put a class A material on the outside of the building if it is compartmentalized behind a cladding panel that is not open ,that has a minimum thickness as is specified by code depending on its material. (For example) if it is zinc it

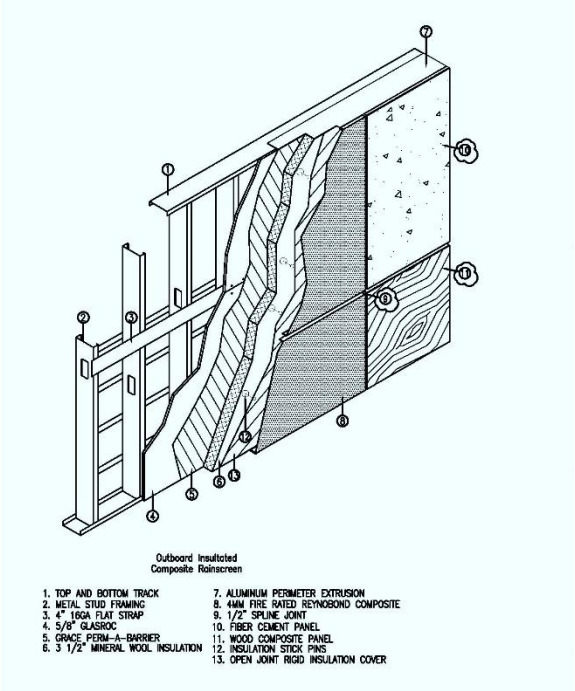


Figure B.6. Mega panel layering (Courtesy of © Marc Simmons, 2015)

(must be) 3 mm or something like that. A class “A” material could be polyiso, it has limitation on its flame spread and smoke creation. So, we always specify mineral wall, it is class zero, problems free. The only issue with mineral wool that if it gets wet, its U value decreases. And the idea is you have a rain screen on front of it, so you don’t have the problem of humidity degradation (Figure B.6).

Normally we put between 3” and 5”. From the energy stand point you get a building 60 % opaque, and 3” gives you R13, 4 and a half R per inch. If you want R16 just add one inch to the insulation, what is very easy. You get a very low cost panel. So, what the contractor is complaining about that is if we have a rain screen outside we need something to join the rain screen back to the structure. So, that is a little bit of thermal bridge, but if you mitigate the number of penetrations and you engineer the connection (between the rain screen and the frame) to be strong and a horizontal rail that carries most of the lateral loads outside of the insulation. And then you can have 2” of air plus the rain screen.

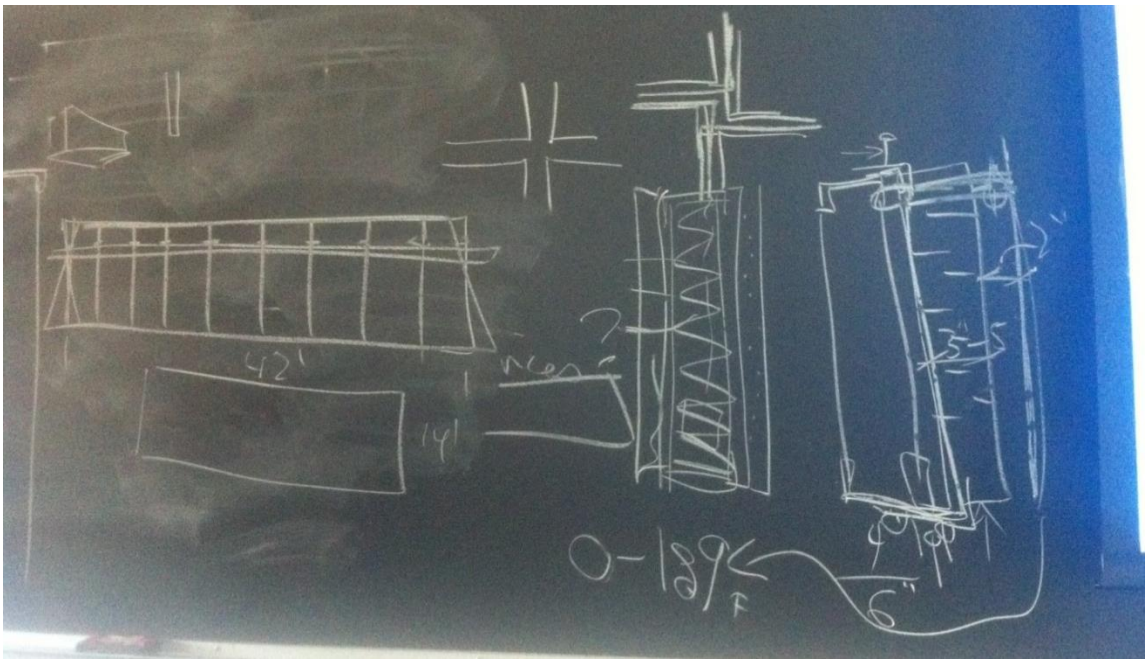


Figure B.7. Structure studs diagrams (Courtesy of © Marc Simmons, 2015)

Now you are in a probable 4” stud, which could be 6” stud, (3” of insulation, 2” of air, and 1” of rain screen), 12” long, not bad. One thing is also very good is there is nothing between studs (Figure B.7). Electrician will have impunity; there is no worries about cutting. They don’t have to wait for the carpenters to come in to finish the water proofing, because as soon the panel goes up with water barrier in place, the building is tied. This panel system could have interlocking legs, like a regular curtain wall, but it can also do this old school double coat joint detail that is more like precast, but is more labor on site and quality control. But if they do that, we were looking for a cheaper approach. It was a big compressible gasket and then a single coat joint on the other side. And the idea is if these panels were just installed, and you had this initial compression seal between the gaskets, just were the panels were in, could that (single coat joint) resist a reasonable level of water infiltration? For most building in NY we could say 12 pounds sqft pressure under water pressure. We can also say 15 pounds, but those are big storms. Then, could the compressible gasket satisfy 8 -10 pounds of pressure differential? And it could. As soon as you hang your panel water is tied, not full

water tied, the single coat is what give you the final water tied. So you got this panel now… it seems that it cost a little bit more because (the) rain screen is more expensive than the stock one… The rain screen is coming as a sort of architectural bench mark, the insulation is absolutely parametric depending entirely on the needs. Then the stand out is engineer to handle the moment generated by the weight of the rain screen relative to the 5-6" distance. This panel approach has become all standard in NY. We actually manage influence, and change the industry which is in alignment with its broader goals.

[Installation]

The only way to install it is hand labor… you get extraordinarily well trained guys in the top of the line. The crane operator, the trucking guys, the glaziers are absolutely top. Because in NY they deal with so much construction of some many high rise stuff and so many logistics challenges, they are practiced, they are very experienced. And on some of our friends rely on them religiously.

[Parametric cost model]

The idea is that you can say I have $64 sqft do you want to add extra insulations? Do you want to do this kind of cladding of that kind of cladding? You got almost a parametric cost model that is so easy to manage like a pre-constriction deign assist bases.

[Panel tolerances]

Thermal expansion, installation access, you certainty have fabrication tolerances, installations tolerances. You have a theoretical location of the edges and four corners of your panels, and what happen if they are not aligned? This is a combination of fabrication tolerances and installation tolerances... And then the thermal expansion gives you overlapping and you get a problem, right? The real risk is when you have thermal expansion at the point where a panel actually touches the other

panel... You must be aware that thermal forces are the most powerful forces in the planet... The idea that two panels are expanding into each other generates in plane forces that will shear the welds of the brackets.

Then you end with the crappy geometry there, then your thermal stress is coming, the whole thing expands, and you get a little problem. We map though all those instances... It comes down to geometrical analyses; you really have to draw it. Draw al the panel in their theoretical perfect conditions, and then you start working through all the materials to say what really my fabrication tolerance is. If the fabrication only can do 1/8" plus/minus, you draw all the panel minus 1/8", everything is bigger you draw it plus 1/8", and you assume that it is not accumulative, if one is under 1/8" the other is over. In terms of fab tolerances, it is more logical to say that it will be a systemic tolerance and everything will be uniformly less 1/8" rather than randomly distributed. To map that up, this is my own logic sizing in one direction, so you have to close unit joint and open it up the joint. Then the next thing to say is what is the installation unit tolerance and are they actually compensate in some degree? Installation tolerances are our biggest problems in panel lost, as you can imagine. ... The survey is not always perfect; they (panels) could be off like a bit, and been off actually matters. We were concerned more about install tolerances, and it was plus or minus ¼" when it should be 1/8". In curtain walls it is 1/8", and you should work on plus or minus 2mm on the fab … In thermal expansion it is more critical than curtain walls, because you have panels up to 15' horizontal mega panels. If you add all these tolerances together is unreasonable, basically there is a subjective artful judgment about which one to cut it off against each other. If you are too conservative, people will say that you are not serious, if my consultant is talking that I have to add 2" to all the joints in everywhere, doesn't help. You have to agree that the fabricator will do things to compensate where he needs to compensate. Sometimes goes to exceptional measures

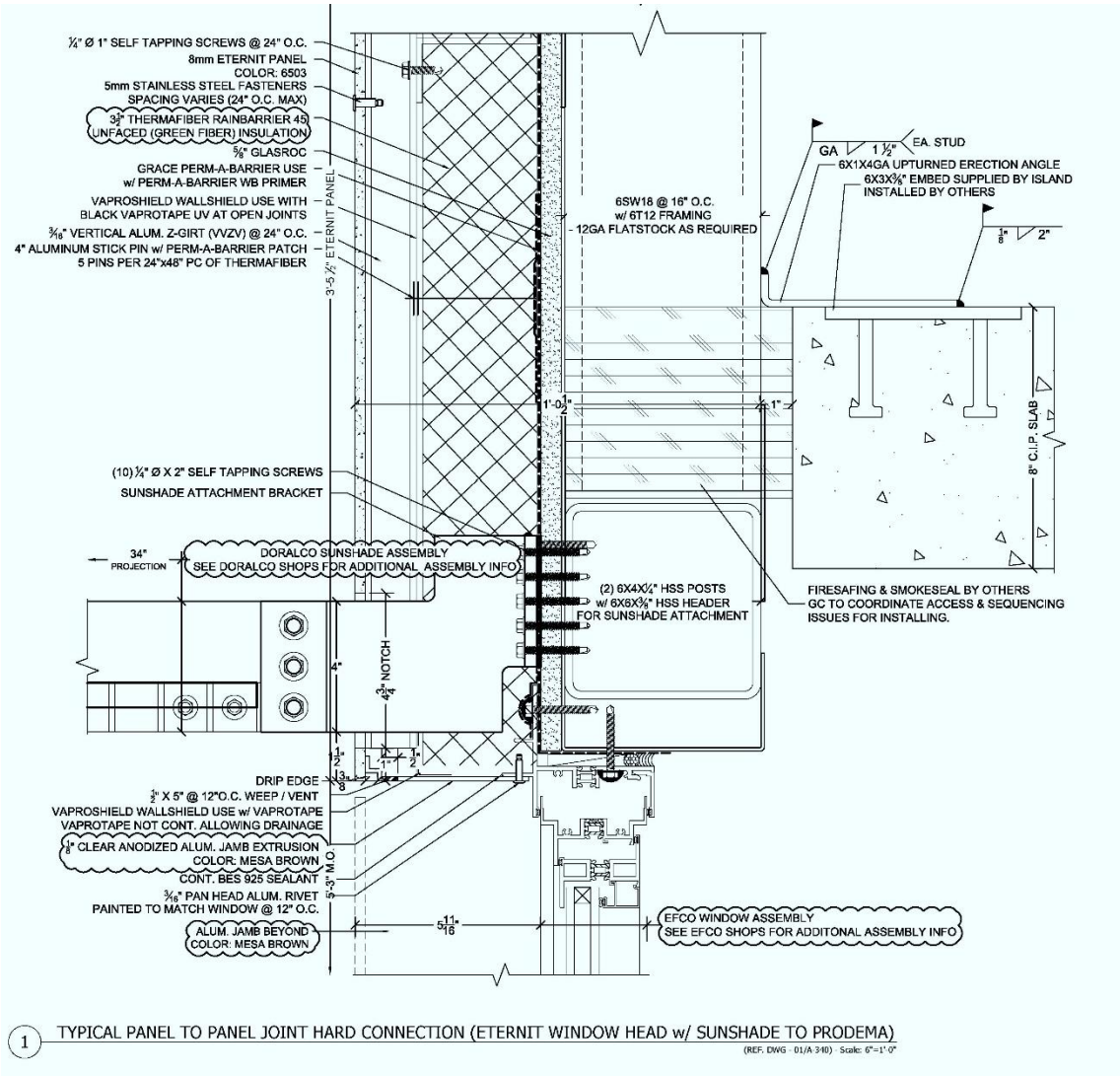


Figure B.8. Detail joint between the mega panel and the slab (Courtesy of © Marc Simmons, 2015)

[Brackets]

Here is a problem, if you have a 15' panel. They have a detail which is a continuous cast in angle… you only do this if you are going to weld your brackets. You are not going to use those tee bolts with connections, U-shape extrusions, and set screws and all that, which is what we love from curtain walls world. But the panel world is in a different play. They love welders. When you actually have your metal stud wall system here, they literally do the simple L-shape bend for the piece of mount steel strapping, and they just weld it (Figure B.8).

They have the slab, and vertical studs welded to the edge… It means they are hanging and the nothing restraining them at all. So, if you have a big temperature swing, this panel changes geometrically into that (trapezoidal shape), because this (the joint to the slab) is constrained. It puts in-plane stresses in all the welded brackets, and induces stresses in the brackets at that point, and we need to design for that. But the bottom is free to expand. So, here comes another aspect of this business which is that the (stud) metal is all behind 3”-4” of insulation.

[Thermal expansion]

All the thermal variant is happening outside of the wall. The rain screen is re-radiating a lot of heat; the insulation is all behind it. Dimensionally (the stud system) it should be quite stable, because it is not seen too much temperature variation, because it is actually in the inside of the building, and it is connected to the structure, which is inside of the building. So, it is all in the same temperature range. And the building goes from 65°F to 74°F seasonal. But if there is a black out... which usually happen at peak thermal moments, you are going to have maximum expansion. What is going to happen is your entire panel wall systems here and the inside of the building are going to ambient (temperature). If it is 0°F outside, it won’t take more than a day that the inside of the building reaches 0°F. Now you start having to say that the stud system is contracting. If it is 104°F outside, it is going to get even higher inside because of the thermal absorption and the re-radiation into the building. The building could conceive to get up to 120°F and the metal outside, just for reference the design temperature range is 0°F to 189°F. That is what we design in our specs... The interior of the building can get really hot. We have to say to our clients that if you want a joint that is 2” wide you have to accept that your water proofing may be compromise in an extreme thermal event. So we specify the joints so that the panels never fail structurally, and don’t induce in-plane load so strong, but the silicon could be pushed back to the point where we (and the contractor) wouldn’t guarantee or defend it. This is one of the intrinsic

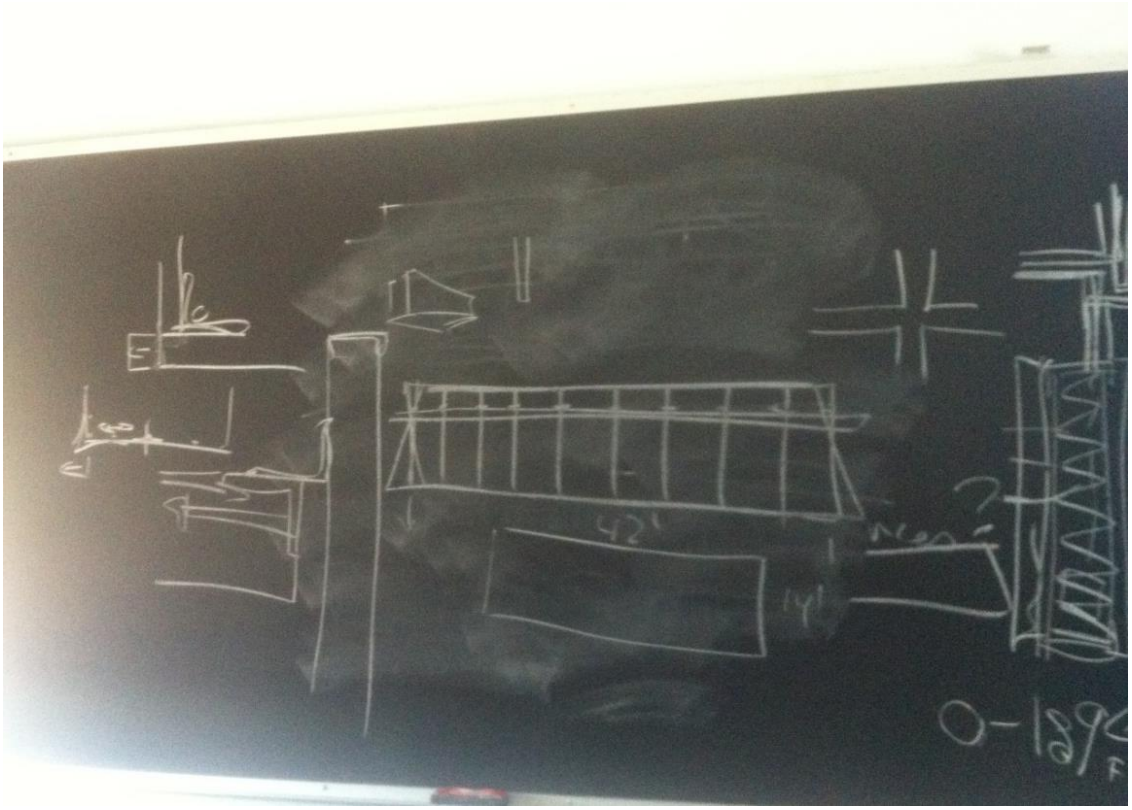


Figure B.9. Mega panel thermal expansion diagram. Corners and trapezoidal deformations (Courtesy of © Marc Simmons, 2015)

problems associated with mega panels. They accumulate thermal forces in very specific locations. Ideally ones want the joint absolutely uniform along the building. At the scale of the building joints hierarchies will be quite dense. The corner panels are jointed together, so that when they expand they keep the integrity of the edge (Figure B.9).

[Deflection]

We are designing to say ¾" deflection of the over the mullion high. We don't care about what the deflections are during peak loads of a category five hurricane. All we care is about its structural integrity, staying on the building and doesn't fall. The deflection is derived from stability requirement and perceptual issue. If the wind comes and the glass deflects ¼" we don't care... If the is a massive 15 years storm and the glass deflects 1" who cares. We don't punish ourselves putting more metal on the wall just to satisfy perception.

[Water infiltration]

We put the water requirement to12pounds/sqft pressure of infiltration not 50 pounds. Because of pressure patterns wind loading wind has a scale, and the code recognizes that. Small elements are designed for higher wind loads statically than larger elements where wind loads are distributed over that panel. If you have a piece of glass of 10' x 40' is going to be designed for higher wind loads, because that whole panel sees theoretical maximum wind load that is a uniformly distributed wind load across that panel. The wind moves across like what you see in water. The reality that any part of a building (is going to) see anything longer than 3 sec gust it is not enough time for the maximum wind pressure to sustain a pressure differential for the inside-outside to help water to move from the outside to the inside. So, 12 pounds/sqft wind pressure is considered more a sustained pressure that you can see in a large storm that is going to yield a pressure differential that will help water move across the seals. That is why the water threshold is much lower than the structural threshold. Also these loads are in seconds, so 3 sec load is a contractual reality. If someone says that you are designing for a 10 sec gust at 50 ps/f. Even by increasing your structure by a significant period, and also the statistical return period of the wind in pretty contractual. So, most buildings are designed for 50 years... many more curtain walls for a100 years, and we have done something for 300 years. Remember that is statistical. You can see s 50 years event, maybe two. We have seen two in the last two years in NY, those numbers are getting weird.

[Seals]

They are the major structural elements, the based and edge beam. That is the backend kind of infill back. These are the aluminum extrusion. This is another thing that we encourage to the fabricator to change. What we wanted to do was basically put an aluminum extrusion that is the same than a unitized curtain wall extrusion over a 40' long panel, and we wanted to put the same extrusion and the top receptor of that. So it

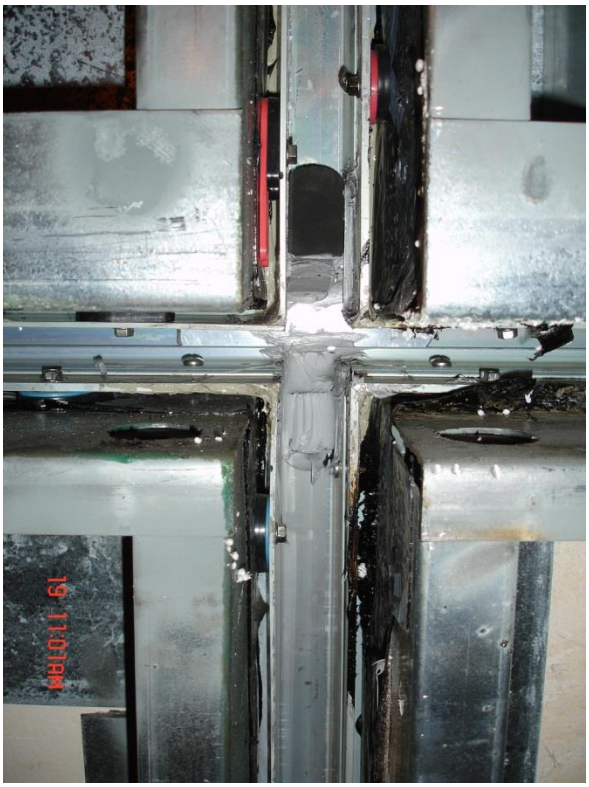


Figure B.10. Seals among mega panels(Courtesy of © Marc Simmons, 2015)

becomes a regular drain joint in a stack joint. So, basically we are taking the studs and skinning them with aluminum extrusion to achieve a unitized curtain wall like water proofing strategy... It got install in the job and if you need a secondary seal you can do it on the field. If you have a problem you can get up from the inside.

Because of this offset the aluminum was design to a much more precise tolerance dimensions from the fabrication stand point, and we gave them enough flap in the design to make sure that the aluminum extrusion could hang off a little bit. The aluminum extrusions are defining the true edge of the panel… and we did the same thing with the windows (Figure B.10). The steel stays crude to keep it cheap. We don't penalize ourselves to asking to the still something it doesn't (want to) do from the cost stand point.

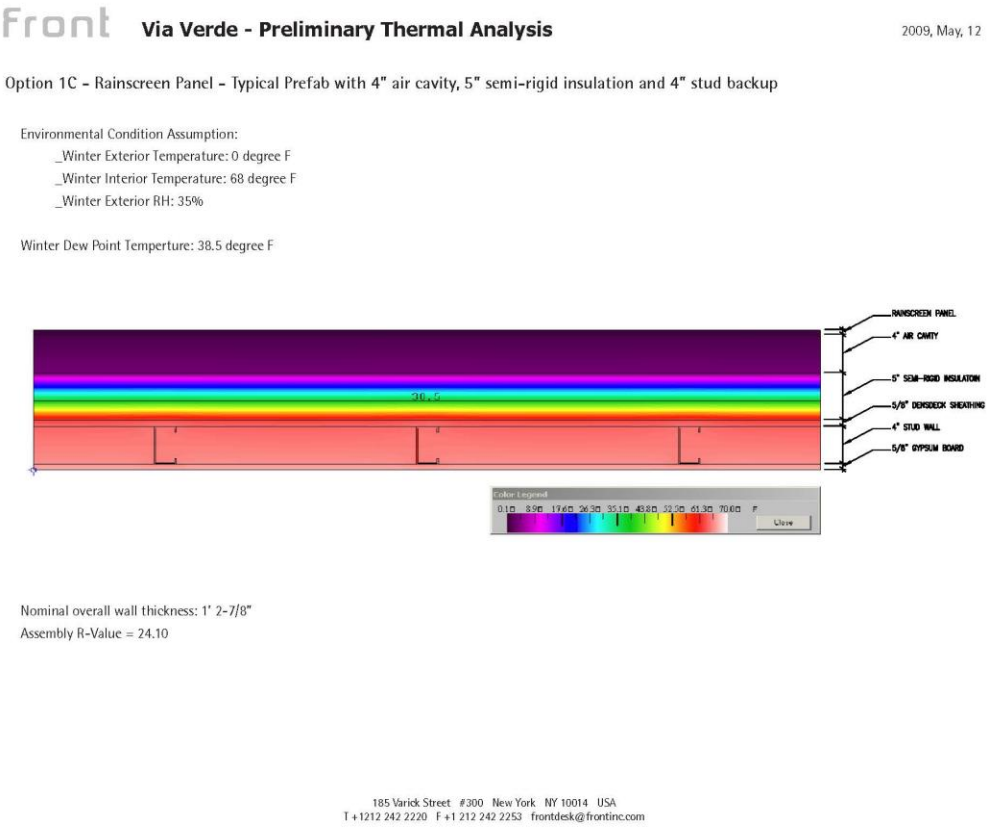


Figure B.11. Thermal analysis plot (Courtesy of © Marc Simmons, 2015)

[Thermal analysis]

When we put those C-studs in there we can see a little local degradation, but it is not enough to cause a problem. Now the 38.5°F (dew point) is far from the sheeting. It is in the middle of the insulation, where we want it to be. 38.5°F is relative to the interior temperature, not the exterior temperature. Code is R 13 for a wall. This building has R24 for walls. U-value for the glassing is 0.5. Several options are evaluated with different combinations of material thicknesses to satisfy the requirements (Figure B.11).

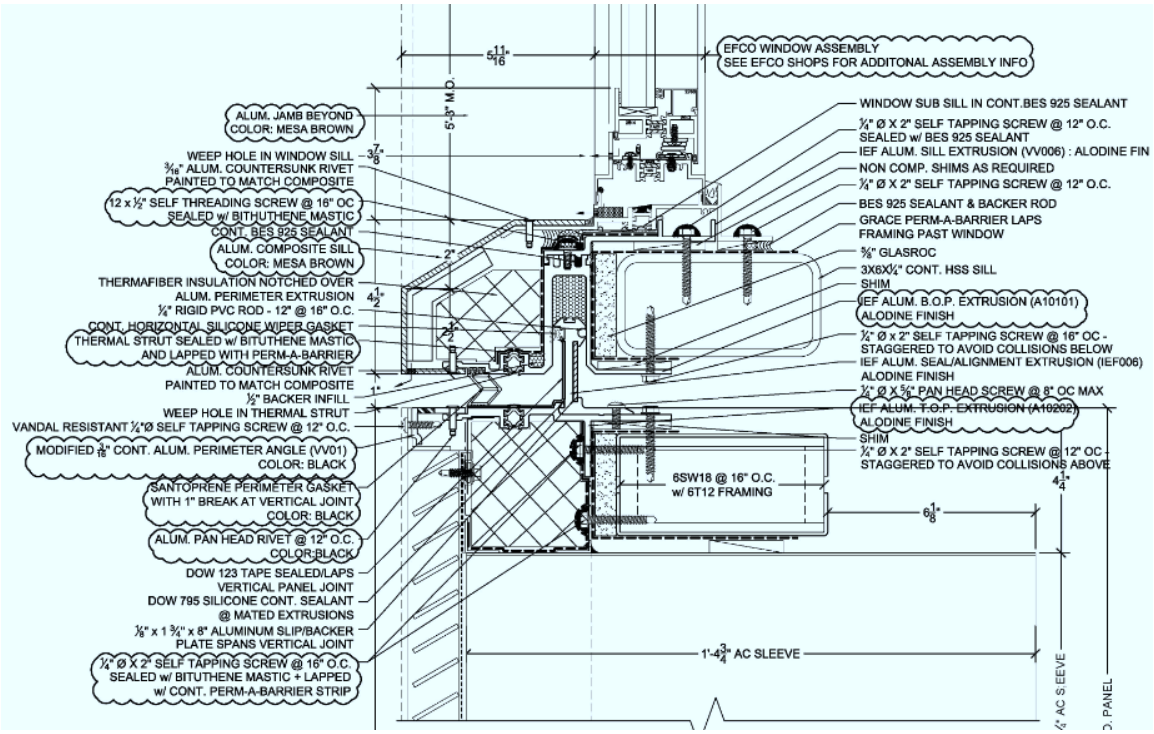


Figure B.12. Detail of connection between mega panels (Courtesy of © Marc Simmons, 2015)

[Window aluminum extrusions]

This where we have real degradation. Because we put the stack joint in bottom of the window joint zone (Figure B.12), there is nothing pulling of the windows. We had to put it in tube steel sections which go over the gap of the steel frame. So we have our standard extrusion, the stack joint, which comes in in this sort of customized window detail, and then right the window frame. So, the window is internally operable, dash drain accident. Basically, water can get in this water drain, it falls in to the receptor… and …a sponge drains it out. All the aluminum extrusions are prefab. We have an extra seal here (between the aluminum frame and the rectangular frame) and a water proofing barrier around the metal frame. But we have a thermal bridge between the bottom of the window and the metal frame. Although all the metals are isolated we can see condensation (in the interior faces of the metal frame). We did condensation analyses and it is marginal. We had to accept this configuration because the contractor

insisted in putting the stack joint close to the metal frame. But the degradation of the U-value of this localized area of the system is balanced by the whole.

The reason why we put little sponge blocks is that it controls air passing though it and dirt. It prevents dirt accumulating of front of it, so the water can get absorbed by the sponge. If it is a lot of water it goes through the sponge, if it a little water it evaporates.

APPENDIX C: 100 & 10th Avenue Residential Building

(Courtesy of © Marc Simmons, 2015)

Architect: Ateliers Jean Nouvel

Structural Engineer: DeSimone Consulting Engineers, PLLC

Façade Consultant: Front Inc.

General Contractor: Gotham Construction Company, LLC

Façade Contractor: CCA Façade Technologies, LLC

New York, NY, 2005 - 2010

Figure C.1. Concept 1, precast concrete (Courtesy of © Marc Simmons, 2015)

[Concept design]

We maximized the perimeter and organized the building with the core on the back, basically, putting all bedrooms and open spaces in the front and the service areas in the back. It is essentially 7 room along the while façade, (and) one prefab panel per façade.

Concept 1: After detail concept review this will be more expensive than concept 2. This is a precast architectural insulated concrete (Figure C.1), and the whole (beam) projects out. Then this infill is a gigantic massive sheet of glass, looking at panels of 35' - 45' long 8' tall single sheet of glass. The developer said if I can't have a monumental panel of glass, it is useless … But the schema did not only come with the mega glass, it had these gigantic kinetic motorized blinds... What you see here is this idea of gigantic perforated metal sheets which are basically lifted up. So, there is couple of things to work here, obviously the kinetic devices, glass sheeting, and outside scaled mullions… We did the numbers of that, the brise-soleil and the glass, and it didn't work. Doing a curve folding kinetic structure just cost too much and also add the geometrical complexity.

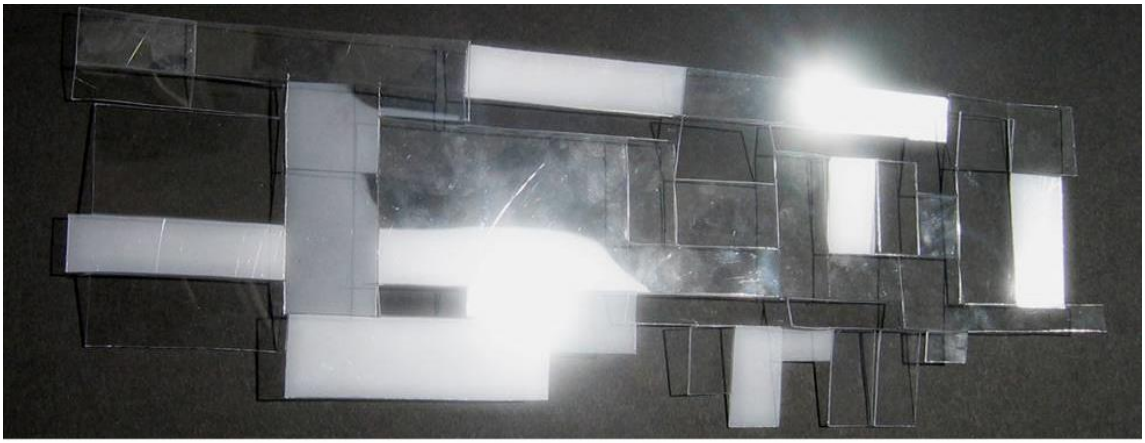


Figure C.2. Concept 2, steel prefab panels (Courtesy of © Marc Simmons, 2015)

Concept 2: Their first design intent was a collage like organization of panels, but more importantly every single piece of glass was intended different than the adjacent panels. These actually don't overlap, they are contiguous, but they are all twisted. Every panel is tilted in four directions (up, down, left and right) in one to five degrees.

The panels are right in line with the horizontal floor planes. The joints are very subtle. We have vertical continuity. So, the whole building becomes rationalized into this mega grid, floor by floor and seven panels per floor plate. After that, everything has some variability in a kind of a crazy grid inside the panels. The collage is based on the specular reflection of the sun on the water… the question was, can we imagine a façade which has such reflection. There are three different type glass, they are all laminated glass. Originally, they didn't want a regular grind on the façade... On top of that, the nonlinear load paths floor to floor add an extra complexity sin it they need to go around the frames. If you are going to do nonlinear load paths with aluminum box mullion (Figure C.3) you need to reinforce them with sheet metal aluminum requires connections detailed as moment connections with large fasteners and exposed bolts... It is, actually, very difficult. Also you need to consider the resulting size of the members. We engineered that to show them. We basically said it is not possible.

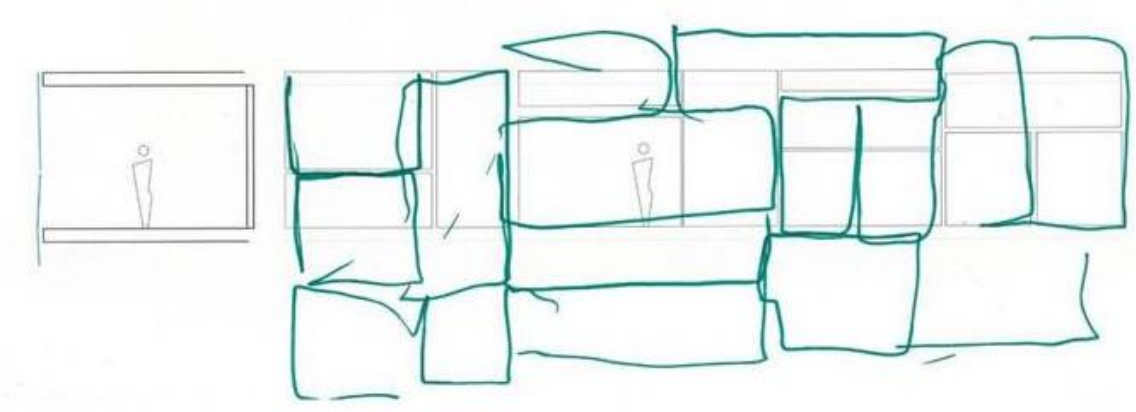


Figure C.3. Diagram of the nonlinear load paths floor to floor (Courtesy of © Marc Simmons, 2015)

The building has this glass floating aesthetics. They wanted to have another layer. The metal frames include these kinds of triangle shadows on the façade. If you look at the face it became no longer a glass façade. If you look this building from the sidewalks… obliquely it collapses into the metal. You see only this insane mesh of metal at certain angles. Once we understood the budget of the building, we realized that it was a reasonable average. It was really clear that the façade of the building could be an expensive metal-glass façade.

[Cost]

The operable windows should be installed in tilted angles. Because the façade represent 40 % of the surface of the building it became 25% (of the total cost). The typical cost is around 12% - 15%

[Street wall]

This is actually quite curious. About 21 stories fit in the site. But if you are going to use the maximum gross square feet ratio of the building, it will reduce slightly the floor plate and stretching the building right tangent to the site will result in a thinner

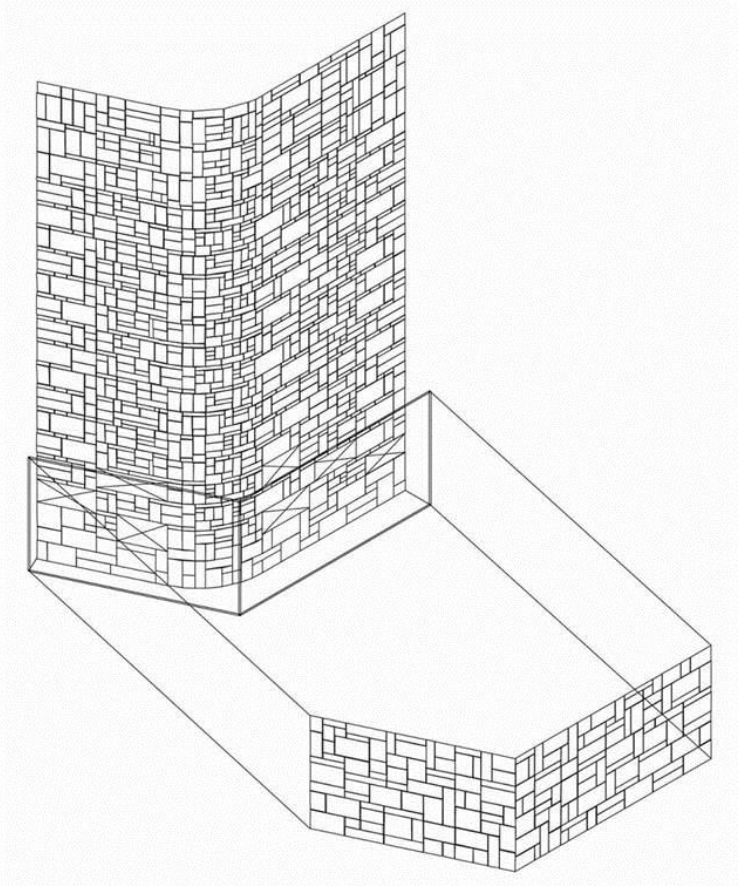


Figure C.4. Continuous side walk façade section (Courtesy of © Marc Simmons, 2015)

building that is not really deep enough to accommodate the depth of the units. So, the fact that the building is settled in bring this opportunity. There is another rule that says that the first 60' high of the building need to maintain the street wall on the sidewalk (Figure C.4). If you push your building back you won't have this continuous façade, and 60' is similar to say five or six stories of the building.

You have the face of your apartment, and you got this glass protected wind screen that is sealing your apartment and terrace. Then it gets more complex since the still has to support this street façade. The landscape designers started to specify those trees boxes in this tridimensional lattice (Figure C.5).

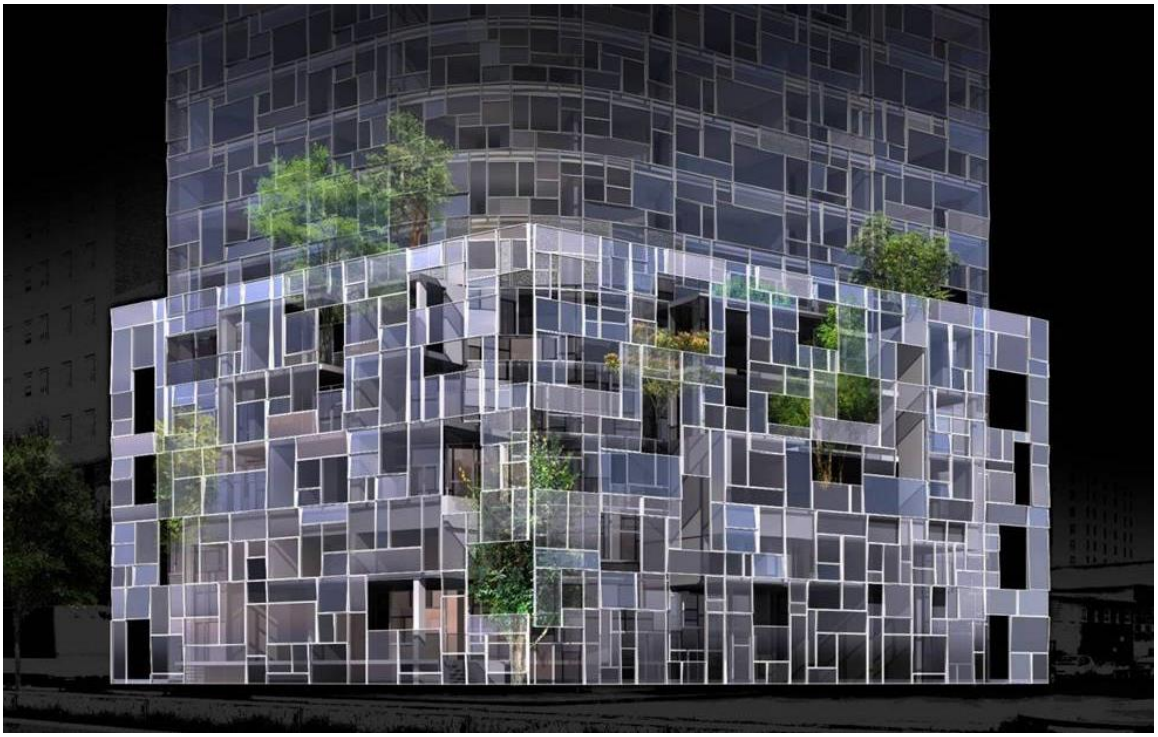


Figure C.5. Tridimensional lattice of the continuous side walk façade section (Courtesy of © Marc Simmons, 2015)

[Inner layout]

There are four apartments per floor. The kitchen and living rooms are aggregated in a big space. The structure is also integrated. Every other column is moved backwards from the plane of the facade and integrated with the subdivision walls. Those columns are also specifically allocated in some small areas where the apartments are divided permanently. So it allows to the entire apartment have full circulation along the façade. And there is a door (in the bedroom) that slides back that can be open. So, you can actually see the entire panorama (from the bedroom), and the whole façade which is (in the corner) 40-45' long.

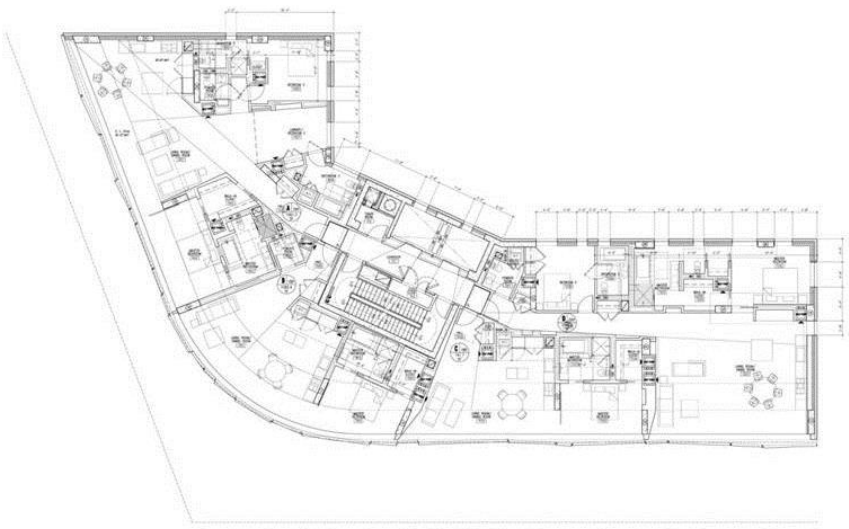


Figure C.6. Typical floor plan layout (Courtesy of © Marc Simmons, 2015)

What is also nice is that the ceilings of the spaces are all plaster concrete, basically directly onto the concrete finish. The whole zone (right behind of the façade) is continuous (Figure C.6).

The lower floors have obstructed views, because of the Chelsea piers across the street, until the fifth or sixth floor. So the idea was how to add some interest, architectural, to those floors. And that is why the kind of game adding terraces and extension came up. Once we saddle on this scheme we start to see how to making it to work. Can the mullions actually all be tapered, tilted and offset from each other and structurally one? No. That brought the suggestion can we fully prefabricate frames as individual frames and just weld them together to just build this complex assembly?... But it proved not to be very adequate. It is difficult for water proofing and insulating.

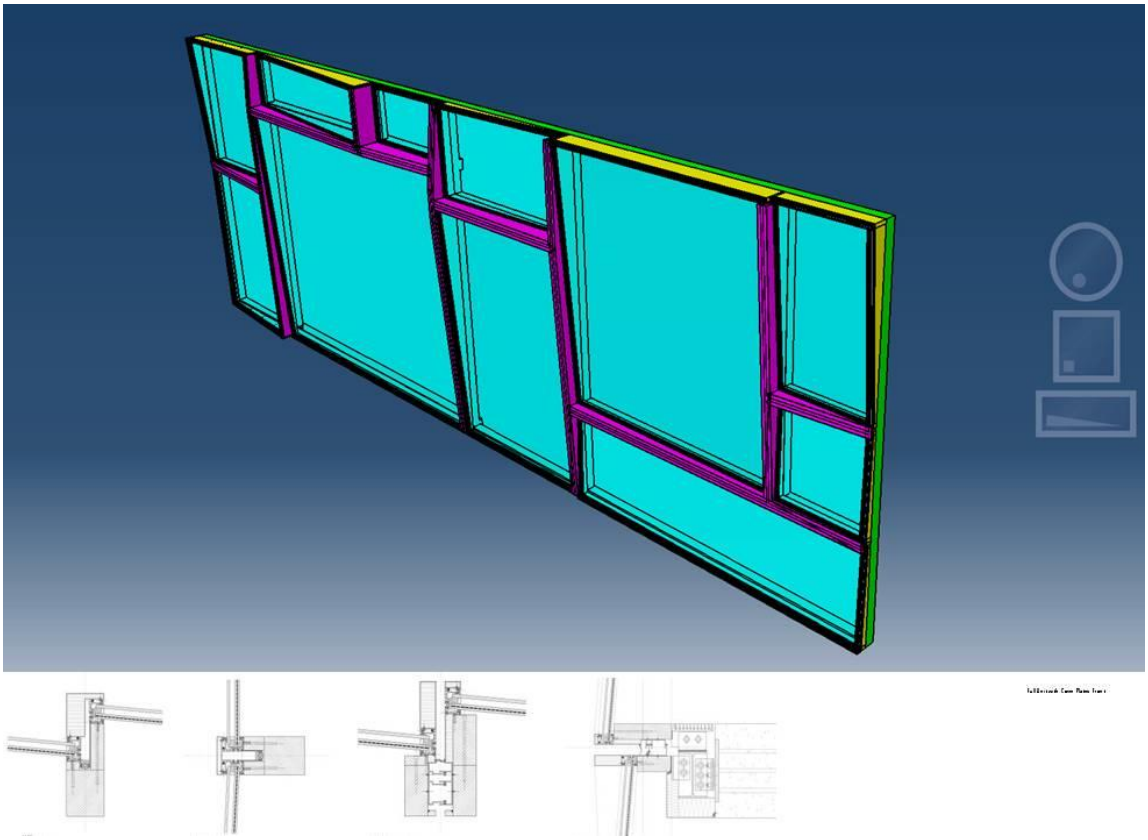


Figure C.7. Panel parametric model (Courtesy of © Marc Simmons, 2015)

[Rationalization of the panel]

What was the goal? What was the strategy? If you take a single line across the entire façade, you need to add some rational management to this problem. So, there is a continuous line across the entire façade which is a reference plane. What is the reference? What is the only thing in this façade that is continuous? That is the exterior face of the structural steel. It will be a single plane across the entire façade. So that, form the inside of the façade you see all the steel coming in and out. Everything outboard of that is going to be gaskets, aluminum, glass, water proofing, everything inboard of that, steel (Figure C.7). We started kind of analyzing the larger panel and sub visions... The panel will be prefabricated...

Figure C.8. Crane lifting the prefab panel (Courtesy of © Marc Simmons, 2015)

[Spreader beam]

To hang this thing onto the building you need a spreader beam (Figure C.8)… we are hanging this like precast. To hang precast you hang it in two points. You don't want three points because you can't guarantee that the three points will carry the loads properties, it (most be) two points. We (are going to) basically find two points along a (horizontal) tube steel that runs inline… that basically structures the whole panel and hang it. Two points of dead load connection (one to the right, another to the left). The beam (4" by 10") will span that, and has a back span that is perfectly balanced with a maximum1/8" vertical displacement, which is virtually flat. In fabrication, each mega panel would be pre-assembled and the beam connected to the panel on site.

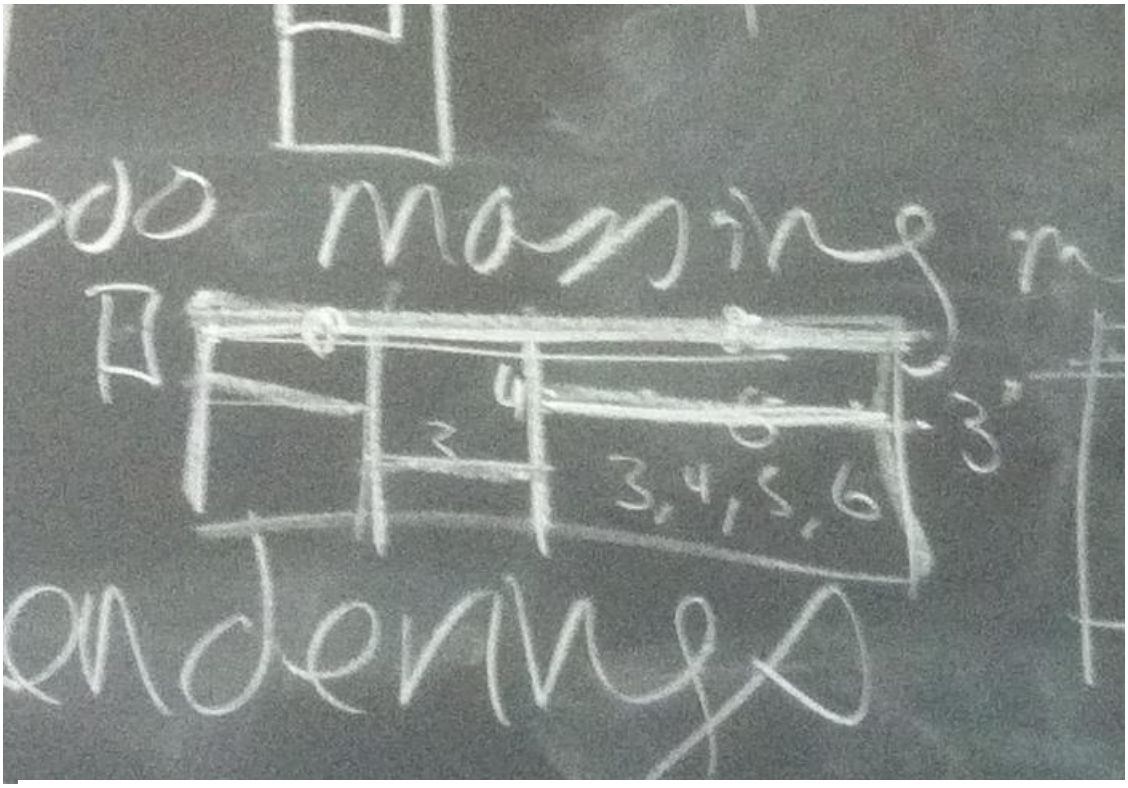


Figure C.9. Panel subdivision based on mullions (Courtesy of © Marc Simmons, 2015)

[Mullions]

So, then we have all the (vertical) millions which are hanged, rationally efficient (Figure C.9). The face dimension of every piece of steel on the inside is going to be limited and harmonized to 3", meaning that now we can play with the depth. And the depth was 3", 4", 5" and 6". Every single piece of steel except for the tube on top is ether 3 by 3, 3 by 4, 3 by 5, or 3 by 6. The bottom member is 6. The glass is not curved (in the curved panel of the corner). The steel frame is curved, but all the glass is faceted. A really long span (is going to be) 6". We will have a vertical piece of steel, and horizontal pieces of steel (sometimes) deeper than the vertical. All this rational is made for the rational of this mega panel.

Figure C.10. Panel subdivision (Courtesy of © Marc Simmons, 2015)

[Mega panel pattern]

How this pattern is made is actually very simple. You have a panel; let's say 11' by 37' and your kitchens close to the right and your living room close to the left. They said that we want our largest piece of glass close to the living room (Figure C.10). We also engineered sized, we set this 7' by 16'. That is our largest piece of glass. Seven feet in one axis is a reasonable piece of glass. We could say 10 by 16, but 7 by 16 is huge but it is also a replaceable size. Then we need 10% of operable window in residential areas in NYC. Then we say, we have 400 sqft. space 40 sqft of operable windows. It could be around 6 by 6. If will have an operable window, you don't want it at the floor level, especially in a high rise. Then you put your window (close to the kitchen) and satisfy your fresh air requirement for the room and you satisfy the requirement of allocate the window from inside where you most appreciate the view. The whole building is composed in that way. Then you connect the dots by extending the edges of these two main modules. A rectangular spreader tube beam behind the frame holds the bracket to put the panel in place on the floor. There are water proofing and insulation in the vertical stack joint, a horizontal metal closure for fire stopping, and room below the beam for the roller blinds. And then you have the vertical steel mullions, the glass and exterior metal trims.

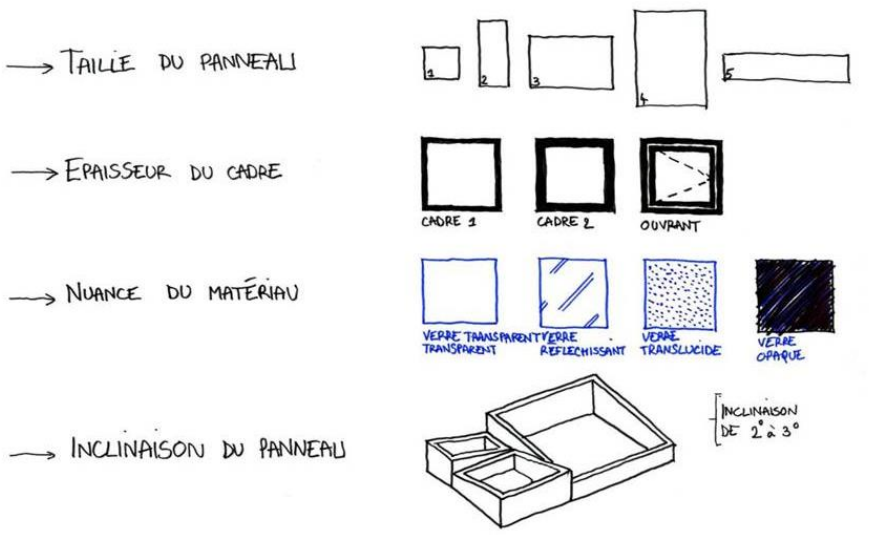


Figure C.11. Drivers for panel composition

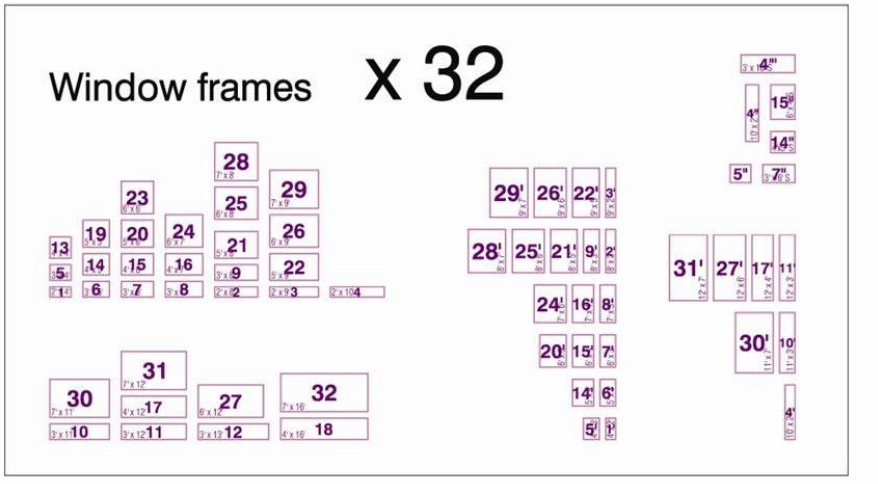


Figure C.12. Window frame sizes (Courtesy of © Marc Simmons, 2015)

[**Panel composition**]

We were using excel design tables to drive the instantiation of the solid components (Figure C.11). We were working very close to (the architect) to create a design map that distill down into the design table driver for the glass… What you can see (in the map) is this mega panelization, structural mullions, etc. You can see also in the curved zone, panels are a bit smaller. The granularity of the panel changes as they get around the corner (Figure C.12). The reading of the building has on the wings the panels get larger. All the area (in the curved zone) hasn't 7' by 16'

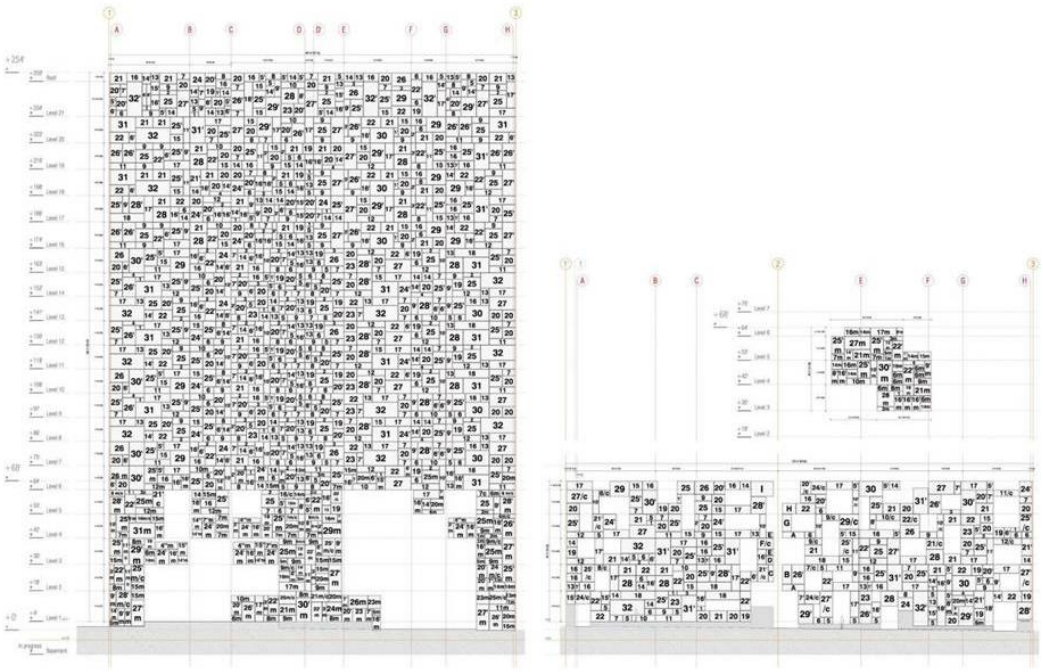


Figure C.13. Window frame distribution (Courtesy of © Marc Simmons, 2015)

The penthouse has 18’ floor to floor. The vertical mullions have bolting patterns, because this part of the top floor was unitized vertical larger assemblies. This is because of trucking limitations for this high.

Atelier Jean Nouvel provided a breakdown of the façade system as a composition of glass panels (Figure C.13) with four direction of rotation: tilting up, down, left and right (Figure C.14); four glass variations (Figure C.15); and angles of rotation varying through 0,2,3,4, and 5 degrees of vertical. Front’s first step was to create a spreadsheet for organizing these parameters along with the glass and panel dimension. The excel file would be referenced as a design table in Digital Project (BIM tool) to associate all parametric variations. In Digital Project, the mega-panels were part-body assemblies with a basic parametric wireframe. The mega-panels were associated with a design table and created as power copy that could be initiated using values from a spread sheet. The mega-panel dimension varies from 11’ x 18’, x 20’, x 37’ and the affect the dimensions and number of component sub-panel”

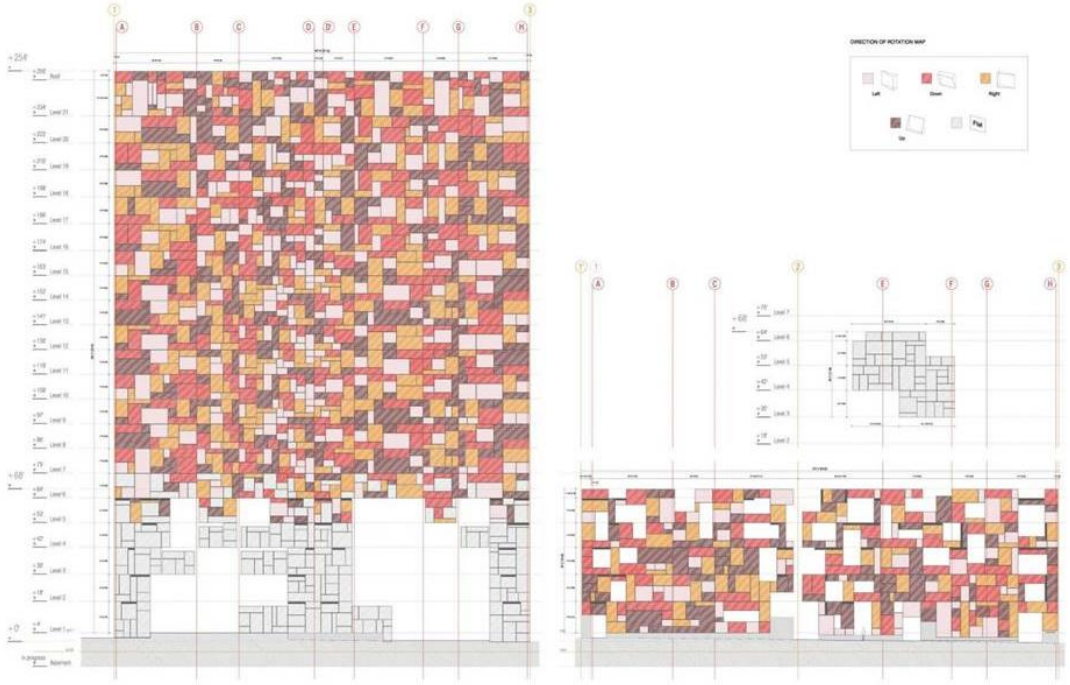


Figure C.14. Tilted angles map (Courtesy of © Marc Simmons, 2015)

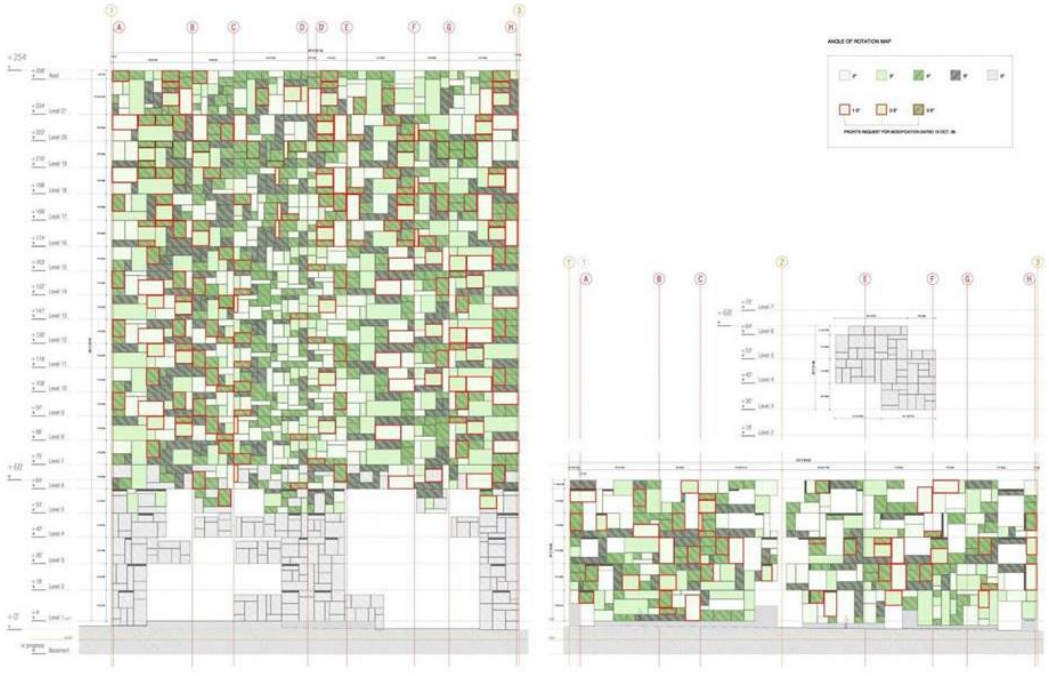


Figure C.15. Glass type map (Courtesy of © Marc Simmons, 2015)

Figure C.16. Adjustable panel bracket (Courtesy of © Marc Simmons, 2015)

[Brackets]

This is something that we worked a lot. That is the edge beam that has a series of horizontal breakers that attach to the top of the edge beam of the slab (Figure C.16)… The curtain wall anchors, basically, they have two compress channels (to adjust it horizontally), four anchors bolts inside, and a dead load seat. The bolt on the bracket pulls on as full lateral and dead load restrains. The panel lays out gently and them it is adjusted.

The crew installed 13 panels per day. There were problems with the paint finish of the steel. It just means that the steel has not been sand blasting to the required degree. It looked great when it was coming over, but in the site it started to bubble… The glass is tilted and rotated. Actually some of the glass are parallelograms

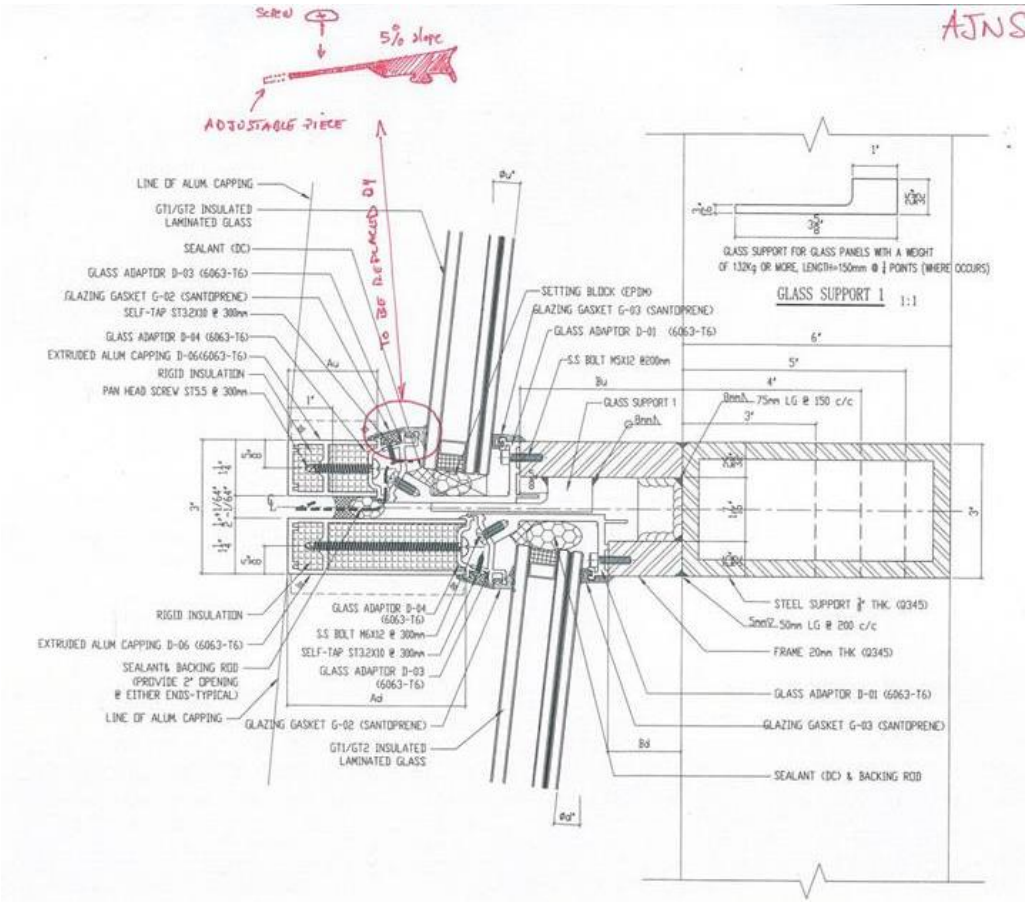


Figure C.17. Window frame details (Courtesy of © Marc Simmons, 2015)

[Extensions]

Steel profile protrusions at the intersections of the mullions vary in length to provide the specified angular tilt of each sub-panel. The triangular gaps between the glass panes and resulting mullions are closed with steel plates at the head and sill of each pane (Figure C.17). The extrusions are welded and sanded smooth to maintain visual continuity between the mullions and the cassettes and to provide place for thermal and acoustical seals.

APPENDIX D: Coding Design Actions

Table D.1 Coding Design Actions

Generalization	Specialization	Case	transcription
Design Situation	Reformulating SBF	C1	The idea that the exterior diagonal grid then come into the picture as a diagonal shear grid that would serve as lateral system to the building, while there were still column grid system in the building, was actually a very late development.
		C2	It (temporary braces) doesn't make any sense from the construction sequence stand point. What you really need is your first panel (hanging from the slab) and the next one seats into that with interlocking pins and you lock it back in (the upper slab), and you are good to go
		C3	So, the whole building becomes rationalized into this mega grid, floor by floor and seven panels per floor plate… Originally, they didn't want a regular grid on the façade…. On top of that, the nonlinear load paths floor to floor added an extra complexity since it then need to go around the frames… We engineered that to show them.
	Forming Analogies	C1	These are just these typical rhetorical stress diagrams from structural analysis software, which resulted in those amoebas as we call them…. And you will see why it matters, because in this area, under dead and wind loads we needed double steel depth in these areas to 24"
		C2	The wind moves across (the facade) like what you see in water
		C3	The collage is based on the specular reflection of the sun on the water… The question was, can we imagine a façade which has such reflection.
	Looking for Emergence	C1	The fact that they (mechanical blinds) are not there is attributable to the density of the steel.
		C2	One thing is also very good is there is nothing between studs. Electricians… have impunity; there is no worries about cutting. They don't have to wait for the carpenters to come in to finish the water proofing
		C3	You have the face of your apartment, and you got this glass protected wind screen that is sealing your apartment and terrace. Then it gets more complex since the still has to support this street façade. The landscape designers started to specify those trees boxes in this tridimensional lattice.
Design Problem	Framing	C1	Everything is a single mega panel…
		C2	The regular approach to layering the panel, was infilling the structure with the insulation… Placing the insulation in the outer face of the frame structure opens things up. (Since) The city is trying to mitigate plumbs impacts (on the insulation).
		C3	So, the whole building becomes rationalized into this mega grid, floor by floor and seven panels per floor plate. After that, everything has some variability in a kind of a crazy grid inside the panels.

Table D.1 (continued)

	Building Ill-define Problems	C1	They (OMA and LMN) wanted the skin elements between the open boxes to be the structure. They just wanted to be that, no columns, all clear span interiors.
		C2	If you add all these tolerances together is unreasonable, basically there is a subjective artful judgment about which one to cut it off against each other. If you are too conservative, people will say that you are not serious, if my consultant is talking that I have to add 2" to all the joints in everywhere, doesn't help.
		C3	…every single piece of glass was intended different than the adjacent panels. These actually don't overlap, they are contiguous, but they are all twisted. Every panel is tilted in four directions (up, down, left and right) in one to five degrees.
	Co-Evolving Problem-Solution	C1	The diagonal grid steel is doing double labor. Imagine we don't have the grid, but I still need to put cladding over 134' span. It needs a massive backup structure, huge, which is actually equal to the size of the bracing structure that would have to be.
		C2	The fact that the door runs across the stack joint which is half way of the room was actually a tradeoff. We had to say that as a technical detail we had to figure out in the macro picture of the project
		C3	Can the mullions actually all be tapered, tilted and offset from each other and structurally one? No. That brought the suggestion: Can we fully prefabricate frames as individual frames and just weld them together to just build this complex assembly? ... But it proved not to be very adequate. It is difficult for water proofing and insulating
Pattern of Organization	Recalling Chunk of Constraints	C1	Stainless steel has one third of the thermal conductivity than aluminum. It is better thermally, but it is not as good as plastic spacers, but the plastic spacers are subject of long term deterioration. So, stainless steel spacers are the most reliable long term solution.
		C2	So, you start to see the joint perimeter. These joints are slightly larger. They are larger because they need to handle thermal expansion…. What panels do, because they are so large, they do expand.
		C3	The penthouse has 18' floor to floor. The vertical mullions have bolting patterns, because this part of the top floor was a unitized vertical larger assembly. This is because of trucking limitations for this high.
	Recalling Conceptual Structures	C1	The idea that the exterior diagonal grid then come into the picture as a diagonal shear grid that would serve as lateral system to the building, while there were still column grid system in the building,
		C2	Everything is a single mega panel, and brise-soleils are bolted in, and the balconies are integrated at the site.
		C3	You can see also in the curved zone, panels are a bit smaller. The granularity of the panel changes as they get around the corner. The reading of the building has on the wings the panels get larger.

Table D.1 (continued)

	Recalling Design Schemas	C1	Eventually the diamond gird came in and it looks more beautiful
		C2	The entire structure of the panel is a vierendeel beam with two hanging portions to the right and left leaving space in the center for the door.
		C3	Their first design intent was a collage like organization of panels, but more importantly every single piece of glass was intended different than the adjacent panels.
Design Solution	Following Parallel Lines of Thought	C1	…during schematic design the entire skin of this building was a tension system. For about six weeks was obsessively done tension system... It took a little while to work up that scheme and get the pre-stress on the cable to support fabric over the 134' spans, and you still have snow loads in Seattle.
		C2	Code is R 13 for a wall. This building has R24 for walls. U-value for the glassing is 0.5. Several options were evaluated with different combinations of material thicknesses to satisfy the requirements
		C3	Concept 1, after detail concept review this (precast beam and single glass) will be more expensive than concept 2 (steel and glass collage).
	Evaluating Preliminary Solutions	C1	Most of the libraries like which have this percent of the glass on it will all have one hundred percent of motorized controlled blinds in there…. In Seattle they (the blinds) are not there, to do diagonal intention trapezoidal blinds could cost $35 sf. just for the blind for the interior.
		C2	The interior of the building can get really hot. We have to say to our clients that if you want a joint that is 2" wide you have to accept that your water proofing may be compromise in an extreme thermal event. So we specify the joints so that the panels never fail structurally, and don't induce in-plane load so strong, but the silicon could be pushed back to the point where we and the contractor wouldn't guarantee or defend it. This is one of the intrinsic problems associated with mega panels.
		C3	If you are going to do nonlinear load paths with aluminum box mullion you need to reinforce them with sheet metal aluminum, (it) requires connections detailed as moment connections with large fasteners and exposed bolts…. It is, actually, very difficult.
	Integrating Knowledge	C1	Industrial stretched metal, we visited the factory… we asked them could they adjust the rate of holes. So once you punch it, the degree of pull governs the degree of aperture in the mesh.
		C2	When we put those C-studs in there we can see a little local degradation, but it is not enough to cause a problem. Now the 38.5°F (dew point) is far from the sheeting. It is in the middle of the insulation, where we want it to be. 38.5°F is relative to the interior temperature, not the exterior temperature. Code is R 13 for a wall. This building has R24 for walls. U-value for the glassing is 0.5. Several options are evaluated (simulated)with different combinations of material thicknesses to satisfy the requirements
		C3	Every other column is moved backwards from the plane of the facade and integrated with the subdivision walls. Those columns are also specifically allocated in some small areas where the apartments are divided permanently. So it allows to the entire apartment have full circulation along the façade.

Table D.1 (continued)

Design Domain	Recognizing Problems	C1	Tension structures always come with a cost since perimeter conditions must be very robust. So, that was killed from the cost stand point of view.
		C2	In thermal expansion it is more critical than curtain walls, because you have up to 15' horizontal mega panels.
		C3	Can we fully prefabricate frames as individual frames and just weld them together to just build this complex assembly?... But it proved not to be very adequate. It is difficult for water proofing and
	Reusing Physical Parts	C1	This (the mullion) could have been a box, it could have been a "T". Obviously, it was chosen an "I" shape because it is conceptually similar to the steel.
		C2	This panel system could have interlocking legs, like a regular curtain wall, but it can also do this old school double coat joint detail that is more like precast, but is more labor on site and quality control.
		C3	To hang this thing onto the building you need a spreader beam… we are hanging this like precast. To hang precast you hang it in two points.
	Applying Design Rules	C1	Only the 50% of the building is covered with glass with mesh integrated, which is only in faces that face South and West all the North, East and down wards faces are actually low-E coatings with pure glass.
		C2	If the fabrication only can do 1/8" plus/minus, you draw all the panel minus 1/8", everything is bigger you draw it plus 1/8", and you assume that it is not accumulative, if one is under 1/8" the other is over. In terms of fab tolerances, it is more logical to say that it will be a systemic tolerance and everything will be uniformly less 1/8" rather than randomly distributed.
		C3	The face dimension of every piece of steel on the inside is going to be limited and harmonized to 3", meaning that now we can play with the depth. And the depth was 3", 4", 5" and 6". Every single piece of steel except for the tube on top is ether 3 by 3, 3 by 4, 3 by 5, or 3 by 6. "The bottom member is 6"
	Applying Evaluation Methods	C1	The testing is not that sophisticated to do, the completely assembly, it has to be component based, so what they do is testing individual layers, that information goes to a data base and a software integrates customized layers built up with all those properties and basically evaluate the aggregate's performance.
		C2	Several options are evaluated (simulated) with different combinations of material thicknesses to satisfy the requirements
		C3	They said that we want our largest piece of glass (close to the living room). We also engineered sized, we set this 7' by 16'.

APPENDIX E: Coding Design Aspects

Table E.1 Coding Design Aspects

Generalization	Specialization	Case	Sample
Structural	Aesthetic	C1	The "I" shape box mullion that has been engineered to span 17' on the clad, this could have been a box, it could have been a "T". Obviously, it was chosen an "I" shape because it is conceptually similar to the steel.
		C2	The rain screen is coming as a sort of architectural bench mark
		C3	(The collage is based on) the specular reflection of the sun on the water… The question was, can we imagine a façade which has such reflection.
	Geometry	C1	The reason it comes about though is that because of the boxes are offset geometrically in two axes, the one in the middle is also a parallelogram. So (the bottom) surface is actually sloping out of plane. And if we try to connect the points below, by definition is a twisted surface.
		C2	Then you end with the crappy geometry there, then your thermal stress is coming, the whole thing expands, and you get a little problem. We map though all those instances…. It comes down to geometrical analyses; you really have to draw it.
		C3	Then you put your window (close to the kitchen) and satisfy your fresh air requirement, for the room you satisfy the requirement of allocate the window from inside where you most appreciate the view. The whole building is composed in that way. Then you connect the dots (by extending the edges of these two main modules)
	Structure	C1	Putting the bracing structure and using it structurally for the lateral stability of the building puts tons of steel inside of the rest of the structure. Then, the façade is important, but actually it is for free.
		C2	The entire structure of the panel is a Vierendeel beam with two hanging portions to the right and the left leaving space in the center for the door
		C3	Every other column is moved backwards from the plane of the facade and integrated with the subdivision walls. Those columns are also specifically allocated in some small areas where the apartments are divided permanently. So it allows to the entire apartment have full circulation along the façade.
	Material	C1	Stainless steel has one third of the thermal conductivity than aluminum. It is better thermally, but it is not as good as plastic spacers, but the plastic spacers are subject of long term deterioration. So, stainless steel spacers are the most reliable long term solution.
		C2	But that panel was made on galvanized plate cold form sheet metal.
		C3	There were problems with the paint finish of the steel. It just means that the steel has not been sand blasting to the required degree. It looked great when it was coming over, but in the site it started to bubble.

Table E.1 (continued)

	Tolerances	C1	Here is the bracket. It is basically a block of aluminum. It has linear slider holes plus minus ¾" adjustments...
		C2	There is actually a completely floating movement joint between those two panels (upper and lower panel)
		C3	The curtain wall anchors, basically, they have two compress channels (to adjust it horizontally), four anchors bolts inside, and a dead load seat. The bolt on the bracket pulls on as full lateral and dead load restrains. The panel lays out gently and them it is adjusted.
	Code	C1	The real problem of that is once you take all of that steel and size it for lateral and gravity it has to be fire protected (by code), and if you are going to fire protect all that steel it is 4.5 million dollars, 5 % percent of the job... and, of course, the density of the steel could be much larger than it is... So what came out of this was the recognition that the building had to have columns
		C2	Code is R 13 for a wall. This building has R24 for walls
		C3	Then we need 10% of operable window in residential areas in NYC. Then we say, we have 400 sqft space 40sqft of operable windows.
Performance	Energy	C1	There are large regions of the building that not have meshes, the mesh is only there where is needed.
		C2	…the insulation is absolutely parametric depending entirely on the needs.
		C3	… to provide room for thermal and acoustical seals.
	Lighting	C1	The fact that they (the blinds) are not there is attributable to the density of the steel.
	Acoustic	C3	…to provide room for thermal and acoustical seals.
	Water proofing	C1	It has three lines of defense inside of the 2" curtain wall.
		C2	The interior of the building can get really hot. We have to say to our clients that if you want a joint that is 2" wide you have to accept that your water proofing may be compromise in an extreme thermal event.
		C3	There are water proofing and insulation in the vertical stack joint,
	Fire protection	C1	The main structure must be fire protected in the connections with the slabs...
		C2	…if you put foam polystyrene outside of the building it just burns,
		C3	…a horizontal metal closure for fire stopping
	Snow accumulation	C1	There is snow accumulation to a certain point. The snow changes the U-value of the assembly, because the snow is insulating. Then, the melt point migrates, the heat inside of the building goes to heat the top part of the glass and then all the snow start to melt, all get lubricated and then in one point snow cap basically goes.

Table E.1 (continued)

	View	C3	And there is a door (in the bedroom) that slides back that can be open. So, you can actually see the entire panorama (from the bedroom), and the whole façade which is (in the corner) 40-45' long.
	Cost	C1	Could the design team be authorized to rise funding? If you didn't have all that certainty somebody could say Can we look other solution? But because it was so well documented the team was allowed to do that.
		C2	The steel stays crude to keep it cheap.
		C3	Once we understood the budget of the building, (we realized that) it was a reasonable average. It was really clear that the façade of the building could be an expensive metal-glass façade.
Procedural	Fabrication	C1	65% of the glass panels on the Seattle library are actually non-standard. Even though the infill panel is forced to the scale of 6x7 panels across the entire facade, the number of edge trimming and small panels is actually huge.
		C2	When you actually have your metal stud wall system here, they literally do the simple L-shape bend for the piece of mount steel strapping, and they just (weld it).
		C3	In fabrication, each mega panel would be pre-assembled and the beam connected to the panel on site"
	Transportation	C1	Those lateral frames basically span from top to bottom. Came in a flat bed and came into the building through cranes, and they are all basically welded.
		C2	There are a limited number of panels that can be stuck in a truck.
		C3	The penthouse has 18' floor to floor. The vertical mullions have bolting patterns, because this part of the top floor was unitized vertical larger assemblies. This is because of trucking limitations for this high.
	Installation	C1	The pre-assembling and indexing of all the components is incredible. The gaskets are actually going on to the extrusions. There these things…. Plastic blocks, and little plastic gaskets.
		C2	What you really need is your first panel (hanging from the slab) and the next one seat into that with interlocking pins and you lock it back in (the upper slab), and you are good to go.
		C3	This is something that we worked a lot. That is the edge beam that has a series of horizontal breakers that attach to the top of the edge beam of the slab
	Maintenance	C1	Cylindrical tie up point on the roof (provide access to maintenance)

REFERENCES

Agarwal, S., & Waggenspack, W. (1992). Decomposition method for extracting face topologies from wireframe models. *Computer-Aided Design, 24*(3), 123-140.

Aho, A. V., & Ullman, J. D. (1972). *The theory of parsing, translation, and compiling*: Prentice-Hall, Inc.

Akin, Ö. (1988). Expertise of the architect. In M. D. Rychener (Ed.), *Expert systems for engineering design* (pp. 171-196). New York: Academic Press.

Alexander, C. (1964). *Notes on the Synthesis of Form* (Vol. 5): Harvard University Press.

Amor, R., & Faraj, I. (2001). Misconceptions about integrated project databases. *Journal of information technology in construction, 6*, 57-68.

Augenbroe, G., & Park, C.-S. (2005). Quantification methods of technical building performance. *Building Research & Information, 33*(2), 159-172.

Baharlou, E., & Menges, A. (2013). *Generative Agent-Based Design Computation.* Paper presented at the eCAADe 2013: Computation and Performance–Proceedings of the 31st International Conference on Education and research in Computer Aided Architectural Design in Europe, Delft, The Netherlands, September 18-20, 2013.

Barg, S., Flager, F., & Fischer, M. (2015). *Decomposition strategies for building envelope design optimization problems.* Paper presented at the Proceedings of the Symposium on Simulation for Architecture & Urban Design.

Becker, R. (2008). *Fundamentals of performance-based building design.* Paper presented at the Building Simulation.

Benros, D., & Duarte, J. (2009). An integrated system for providing mass customized housing. *Automation in Construction, 18*(3), 310-320.

Bento, J., & Feijó, B. (1997). An agent-based paradigm for building intelligent CAD systems. *Artificial Intelligence in Engineering, 11*(3), 231-244.

Bernal, M. (2011). *Analysis model for incremental precision along design stages.* Paper presented at the 16th International Conference on Computer-Aided Architectural Design Research in Asia, New Castle, Autralia.

Bernal, M., & Eastman, C. (2011). *Top-down Approach to Embed Design Expertise in Parametric Objects for the Automatic Generation of a Building Service Core.* Paper presented at the CAAD Futures.

Bernal, M., Haymaker, J. R., & Eastman, C. (2015). On the role of computational support for designers in action. *Design Studies, 41*, 163-182. doi:10.1016/j.destud.2015.08.001

Bielak, R. (1993). Object oriented programming: the fundamentals. *ACM SIGPLAN Notices, 28*(9), 13-14.

Bloom, B. (1956). Taxonomy of educational objectives: The classification of educational goals.

Bohnke, D., Reichwein, A., & Rudolph, S. (2009). *Design language for airplane geometries using the unified modeling language.* Paper presented at the ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference.

Brown, D. (2003). Functional, behavioral and structural features. *ASME Design Engineering Technical*.

buildingSMART Data Dictionary. (2016). Retrieved from http://bsdd.buildingsmart.org/

Caldas, L., & Norford, L. (2003). Genetic algorithms for optimization of building envelopes and the design and control of HVAC systems. *Journal of Solar Energy Engineering, 125*, 343.

Chandrasegaran, S. K., Ramani, K., Sriram, R. D., Horváth, I., Bernard, A., Harik, R. F., & Gao, W. (2013). The evolution, challenges, and future of knowledge representation in product design systems. *Computer-Aided Design, 45*(2), 204-228. doi:http://dx.doi.org/10.1016/j.cad.2012.08.006

Chandrasekaran, B., & Milne, R. (1985). Reasoning about structure, behavior and function. *SIGART Bull.*(93), 4-55. doi:10.1145/1056557.1056561

Chi, M. T. (1997). Quantifying qualitative analyses of verbal data: A practical guide. *The journal of the learning sciences, 6*(3), 271-315.

Crismond, D. (2001). Learning and using science ideas when doing investigate-and-redesign tasks: A study of naive, novice, and expert designers doing constrained and scaffolded design work. *Journal of Research in Science Teaching, 38*(7), 791-820.

Cross, N. (1990). The nature and nurture of design ability. *Design Studies, 11*(3), 127-140.

Cross, N. (2004). Expertise in design: an overview. *Design Studies, 25*(5), 427-441.

Cui, H., & Turan, O. (2010). Application of a new multi-agent hybrid co-evolution based particle swarm optimisation methodology in ship design. *Computer-Aided Design, 42*(11), 1013-1027.

Dabbeeru, M. M., & Mukerjee, A. (2008). Discovering implicit constraints in design *Design Computing and Cognition'08* (pp. 201-220): Springer.

Danso-Amoako, M., O'Brien, W., & Issa, R. (2004). *A case study of IFC and CIS/2 support for steel supply chain processes.* Paper presented at the Proceedings of the 10th International Conference Computing in Civil and Building Engineering (ICCCBE-10), Weimar, June.

Davies, J., Goel, A., & Nersessian, N. (2009). A computational model of visual analogies in design. *Cognitive Systems Research, 10*(3), 204-215.

Dogan, F., & Nersessian, N. J. (2010). Generic abstraction in design creativity: the case of Staatsgalerie by James Stirling. *Design Studies, 31*(3), 207-236.

Dorst, K. (2007). The Problem of Design Problems. In N. G. E. Cross, E. (Ed.), *Expertise in Design - Design Thinking Research Symposium 6*. Sydney: Creativity and Cognition Studios Press,.

Dorst, K., & Cross, N. (2001). Creativity in the design process: co-evolution of problem–solution. *Design Studies, 22*(5), 425-437.

Dreyfus, H. L. (1992). *What computers still can't do: a critique of artificial reason*: MIT press.

Dreyfus, H. L. (1997). Intuitive, deliberative, and calculative models of expert performance. *Naturalistic decision making*, 17-28.

Duarte, J. P. (2005). A discursive grammar for customizing mass housing: the case of Siza's houses at Malagueira. *Automation in Construction, 14*(2), 265-275.

Eastman, C. (1999). *Building product models: computer environments, supporting design and construction*. Boca Raton, Florida: CRC press.

Eastman, C. (2001a). New directions in design cognition: studies of representation and recall: Elsevier.

Eastman, C. (2001b). New directions in design cognition: studies of representation and recall. 147-198.

Eastman, C. (2004). *New Methods of Architecture and Building.* Paper presented at the ACADIA, Toronto, Canada.

Eastman, C., Jeong, Y., Sacks, R., & Kaner, I. (2010). Exchange model and exchange object concepts for implementation of national BIM standards. *Journal of Computing in Civil Engineering, 24*, 25.

Eastman, C., Lee, J.-m., Jeong, Y.-s., & Lee, J.-k. (2009). Automatic rule-based checking of building designs. *Automation in Construction, 18*(8), 1011-1033.

Eastman, C., Sacks, R., & Lee, G. (2003). *Development of a Knowledge-Rich CAD System for American Precast Concrete Industry.* Paper presented at the Association for Computer Aided Design in Architecture (ACADIA), Muncie, Indiana.

Eastman, C., Sacks, R., & Lee, G. (2004). Functional modeling in parametric CAD systems.

Eastman, C., Teicholz, P., Sacks, R., & Liston, K. (2011). *BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers, and Contractors.* (second ed.). Hoboken, NJ: John Wiley & Sons Inc.

Eck, O., & Schaefer, D. (2011). A semantic file system for integrated product data management. *Advanced Engineering Informatics, 25*(2), 177-184.

Eclipse. (2016, 03/02/2016). Retrieved from http://www.eclipse.org/

Eclipse Modeling Framework. (2016). Retrieved from http://www.eclipse.org/modeling/emf/

Ericsson, A., & Simon, H. (1984). Protocol analysis: verbal reports as data Boston: The MIT Press.

Flager, F., Welle, B., Bansal, P., Soremekun, G., & Haymaker, J. (2009). Multidisciplinary process integration and design optimization of a classroom building. *Journal of information technology in construction, 14*, 595-612.

Frazer, J., Tang, M., & Sun, J. (1999). *Towards a generative system for intelligent design support.* Paper presented at the Proceedings of the 4th CAADRIA Conference.

Friedenthal, S., Moore, A., & Steiner, R. (2011). *A practical guide to SysML: the systems modeling language*: Elsevier.

GehryTechnologies. (2009). A Brief Introduction to Digital Project Automation. Retrieved from http://www.gtwiki.org/mwiki/r4doc/English/online/books/AutomationAPI/_book.htm

Generative Components (2016). Retrieved from https://www.bentley.com/en/products/product-line/modeling-and-visualization-software/generativecomponents

Gero, J. S. (1998). Conceptual designing as a sequence of situated acts *Artificial intelligence in structural engineering* (pp. 165-177): Springer.

Gero, J. S., & Kannengiesser, U. (2004). The situated function–behaviour–structure framework. *Design Studies, 25*(4), 373-391.

Gibson, J. J. (1977). The theory of affordances. *Hilldale, USA*.

Glymph, J., Shelden, D., Ceccato, C., Mussel, J., & Schober, H. (2004). A parametric strategy for free-form glass structures using quadrilateral planar facets. *Automation in Construction, 13*(2), 187-202.

Gobet, F., Lane, P. C., Croker, S., Cheng, P. C., Jones, G., Oliver, I., & Pine, J. M. (2001). Chunking mechanisms in human learning. *Trends in cognitive sciences, 5*(6), 236-243.

Goel, A. (1997). Design, analogy, and creativity. *IEEE expert, 12*(3), 62-70.

Goel, A., & Bhatta, S. (2004). Use of design patterns in analogy-based design. *Advanced Engineering Informatics, 18*(2), 85-94.

Goel, A., & Chandrasekaran, B. (1992). Case-based design: A task analysis. *Artificial intelligence approaches to engineering design, 2*, 165-184.

Goel, A., & Stroulia, E. (1996). Functional device models and model-based diagnosis in adaptive design. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing, 10*(04), 355-370.

Goel, A., Vattam, S., Wiltgen, B., & Helms, M. (2012). Cognitive, collaborative, conceptual and creative—four characteristics of the next generation of knowledge-based CAD systems: a study in biologically inspired design. *Computer-Aided Design, 44*(10), 879-900.

Goldschmidt, G. (2006). Quo vadis, design space explorer? *AIE EDAM: Artificial Intelligence for Engineering Design, Analysis, and Manufacturing, 20*(02), 105-111.

Goldscmidt, G. (1988). Interpretation: its role in architectural designing. *Design Studies, 9*(4), 235-245.

Gonnet, S., Henning, G., & Leone, H. (2007). A model for capturing and representing the engineering design process. *Expert Systems with Applications, 33*(4), 881-902.

Grasl, T., & Economou, A. (2011). *GRAPE: Using graph grammars to implement shape grammars.* Paper presented at the Proceedings of the 2011 Symposium on Simulation for Architecture and Urban Design.

Grasl, T., & Economou, A. (2013). From topologies to shapes: parametric shape grammars implemented by graphs. *Environment and Planning B: Planning and Design, 40*(5), 905-922.

Gross, M. D. (1996). The electronic cocktail napkin—a computational environment for working with design diagrams. *Design Studies, 17*(1), 53-69.

Haymaker, J. R., Chachere, J. M., & Senescu, R. R. (2011). Measuring and improving rationale clarity in a university office building design process. *Journal of Architectural Engineering, 17*(3), 97-111.

Heylighen, A., & Neuckermans, H. (2001). A case base of case-based design tools for architecture. *Computer-Aided Design, 33*(14), 1111-1122.

Hollan, J., Hutchins, E., & Kirsh, D. (2000). Distributed cognition: toward a new foundation for human-computer interaction research. *ACM Transactions on Computer-Human Interaction (TOCHI), 7*(2), 174-196.

Hopfe, C. J., Augenbroe, G. L., & Hensen, J. L. (2013). Multi-criteria decision making under uncertainty in building performance assessment. *Building and Environment, 69*, 81-90.

Hubka, V., & Eder, W. E. (1987). A scientific approach to engineering design. *Design Studies, 8*(3), 123-137.

Hutchins, E. (2000). Distributed cognition. *International Encyclopedia of the Social and Behavioral Sciences. Elsevier Science*.

INCOSE. (2016, 01/12/2016). Retrieved from http://www.incose.org/

Ittelson, W. H., Rivlin, L. G., & Prositanskt, H. M. (1970). Chapter 65: The use of behavioral maps in environmental psychology. In W. H. Ittelson, L. G. Rivlin, & H. M. Prositanskt (Eds.), *Man and its physical settings*. New York, NY: Holt, Rinehart and Ivinston Inc.

Jobe, J. M., Johnson, T. A., & Paredis, C. J. (2008). *Multi-Aspect Component Models: A Framework for Model Reuse in SysML.* Paper presented at the ASME 2008 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference.

Kalay, Y. E. (1989). *Modeling objects and environments*: Wiley.

Kalay, Y. E. (1999). Performance-based design. *Automation in Construction, 8*(4), 395-409. doi:Doi: 10.1016/s0926-5805(98)00086-7

Kasik, D., Buxton, W., & Ferguson, D. (2005). Ten CAD challenges. *IEEE Computer Graphics and Applications*, 81-92.

Keeney, R. L., & Raiffa, H. (1993). *Decisions with multiple objectives: preferences and value trade-offs*: Cambridge university press.

Kifer, M., Lausen, G., & Wu, J. (1995). Logical foundations of object-oriented and frame-based languages. *Journal of the ACM (JACM), 42*(4), 741-843.

Kleijnen, J. P. (1997). Sensitivity analysis and related analyses: a review of some statistical techniques. *Journal of Statistical Computation and Simulation, 57*(1-4), 111-142.

Koch, P. N., Simpson, T. W., Allen, J. K., & Mistree, F. (1999). Statistical approximations for multidisciplinary design optimization: the problem of size. *Journal of Aircraft, 36*(1), 275-286.

Kolodner, J. L. (1992). An introduction to case-based reasoning. *Artificial Intelligence Review, 6*(1), 3-34.

Kruger, C., & Cross, N. (2006). Solution driven versus problem driven design: strategies and outcomes. *Design Studies, 27*(5), 527-548.

Kühne, T. (2006). Matters of (meta-) modeling. *Software & Systems Modeling, 5*(4), 369-385.

La Rocca, G. (2011). *Knowledge based engineering techniques to support aircraft design and optimization*: TU Delft, Delft University of Technology.

Lawson, B. (2004). Schemata, gambits and precedent: some factors in design expertise. *Design Studies, 25*(5), 443-457.

Lawson, B. (2012). *What designers know*: Routledge.

Lawson, B., & Dorst, K. (2009). Design expertise. *Recherche, 67*, 02.

Lee, G., Sacks, R., & Eastman, C. (2006). Specifying Parametric Building Object Behavior (BOB) for a Building Information Modeling System. *Automation in Construction, 15*(6), 758-776.

Lee, J.-k., Eastman, C., Lee, J., Kannala, M., & Jeong, Y.-s. (2010). Computing walking distances within buildings using the universal circulation network. *Environment and planning. B, Planning & design, 37*(4), 628.

Lequette, R. (1988). Automatic construction of curvilinear solids from wireframe views. *Computer-Aided Design, 20*(4), 171-180.

Linsey, J., Laux, J., Clauss, E., Wood, K., & Markman, A. (2007). *Effects of analogous product representation on design-by-analogy.* Paper presented at the Proc. Int. Conf. Engineering Design, ICED.

Linsey, J., Murphy, J., Markman, A., Wood, K., & Kurtoglu, T. (2006). *Representing analogies: Increasing the probability of innovation.* Paper presented at the ASME International Design Theory and Methodology Conference.

Lipman, R. (2009). Details of the mapping between the CIS/2 and IFC product data models for structural steel. *ITcon, 14*(1), 1-13.

MagicDraw. (2016, 2016). Retrieved from http://www.nomagic.com/products/magicdraw.html

Maher, M. L., Balachandran, M., & Zhang, D. M. (1995). *Case-based reasoning in design*: Psychology Press.

Maher, M. L., & Poon, J. (1996). Modeling Design Exploration as Co-Evolution. *Computer-Aided Civil and Infrastructure Engineering, 11*(3), 195-209.

Maher, M. L., Poon, J., & Boulanger, S. (1996). Formalising design exploration as co-evolution *Advances in formal design methods for CAD* (pp. 3-30): Springer.

Maher, M. L., Poon, J., & Boulanger, S. (1996). Formalising design exploration as co-evolution: a combined gene approach.

Maher, M. L., & Pu, P. (1997). Issues and Applications of Case Based Reasoning to Design.

Mantyla, M. (1988). *An Introduction to Solid Modeling*.

McCormack, J., Dorin, A., & Innocent, T. (2004). Generative design: a paradigm for design research. *Proceedings of Futureground, Design Research Society, Melbourne*.

McCormack, J. P., Cagan, J., & Vogel, C. M. (2004). Speaking the Buick language: capturing, understanding, and exploring brand identity with shape grammars. *Design Studies, 25*(1), 1-29.

Medjdoub, B. (2009). Constraint-based adaptation for complex space configuration in building services. *ITcon, 14*, 724-735.

Mela, K., Tiainen, T., & Heinisuo, M. (2012). Comparative study of multiple criteria decision making methods for building design. *Advanced Engineering Informatics, 26*(4), 716-726.

Modi, S., Tiwari, M., Lin, Y., & Zhang, W. (2011). On the architecture of a human-centered CAD agent system. *Computer-Aided Design, 43*(2), 170-179.

Moreno, D. P., Hernandez, A. A., Yang, M. C., Otto, K. N., Hölttä-Otto, K., Linsey, J. S., . . . Linden, A. (2014). Fundamental studies in design-by-analogy: A focus on domain-knowledge experts and applications to transactional design problems. *Design Studies, 35*(3), 232-272.

Mourshed, M., Shikder, S., & Price, A. D. (2011). Phi-array: A novel method for fitness visualization and decision making in evolutionary design optimization. *Advanced Engineering Informatics, 25*(4), 676-687.

Najle, C. (2013). *Maquinas Textiles*. Keynote at the XVII Conference of the Iberoamerican Society of Digital Graphics, SIGraDi. Univeridad Técnica Federico Santa María. Valparaíso, Chile.

Nassar, K., Thabet, W., & Beliveau, Y. (2003). Building Assembly Detailing Using Constraint-Based Modeling. *Automation in Construction, 12*(4), 365-379.

Nicknam, M., Bernal, M., & Haymaker, J. (2013). *A Case Study in Teaching Construction of Building Design Spaces.* Paper presented at the eCAADe 2013: Computation and Performance–Proceedings of the 31st International Conference on Education and research in Computer Aided Architectural Design in Europe, Delft, The Netherlands, September 18-20, 2013.

OMG Systems Modeling Language. (2015). Retrieved from http://www.omgsysml.org/

Ousterhout, J. K. (1998). Scripting: Higher level programming for the 21st century. *Computer, 31*(3), 23-30.

Oxman, R. (2002). The thinking eye: visual re-cognition in design emergence. *Design Studies, 23*(2), 135-164. doi:http://dx.doi.org/10.1016/S0142-694X(01)00026-6

Pahl, G., Beitz, W., & Wallace, K. (1996). *Engineering Design: A Systematic Approach*. Berlin: Springer Verlag.

Parnas, D. L. (1972). On the criteria to be used in decomposing systems into modules. *Communications of the ACM, 15*(12), 1053-1058.

Parrish, K., & Tommelein, I. (2009). *Making design decisions using Choosing By Advantages.* Paper presented at the Proc. 17th Annual Conference of the International Group for Lean Construction (IGLC 17).

Pearce, M., Goel, A., Kolodner, J., Zimring, C., Sentosa, L., & Billington, R. (1992). Case-based design support: A case study in architectural design. *IEEE expert, 7*(5), 14-20.

Pedersen, K., Emblemsvåg, J., Bailey, R., Allen, J., & Mistree, F. (2000). *The'Validation Square'-Validating Design Methods.* Paper presented at the ASME Design Theory and Methodology Conference.

Penn, A., & Turner, A. (2001). Space syntax based agent simulation.

Popovic, V. (2004). Expertise development in product design—strategic and domain-specific knowledge connections. *Design Studies, 25*(5), 527-545.

Pratt, J. W., Raiffa, H., & Schlaifer, R. (1964). The foundations of decision under uncertainty: An elementary exposition. *Journal of the American Statistical Association, 59*(306), 353-375.

Purcell, T., Gero, J., Edwards, H., & McNeill, T. (1996). The data in design protocols: The issue of data coding, data analysis in the development of models of the design process. In N. Cross, H. Christiaans, & K. Dorst (Eds.), *Analysing design activity* (pp. 225-251): Wiley.

Reichwein, A., & Paredis, C. J. (2011). *Overview of Architecture Frameworks and Modeling Languages for Model-Based Systems Engineering.* Paper presented at the Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference.

Requicha, & Rossignac, J. (1992). Solid modeling and beyond. *IEEE Computer Graphics and Applications, 12*(5), 31-44.

Requicha, A. (1980). Representations for rigid solids: Theory, methods, and systems. *ACM Computing Surveys (CSUR), 12*(4), 437-464.

Salomon, G. (1993). *Distributed cognitions: Psychological and educational considerations* (Vol. 11): Cambridge University Press Cambridge.

Sanguinetti, P., Bernal, M., El-Khaldi, M., & Erwin, M. (2010). *Real-time design feedback: coupling performance-knowledge with design iteration for decision-making.* Paper presented at the Spring Simulation Multiconference, Orlando, Florida.

Schaefer, D. (2006). A generic approach to automated product variant design technology. *Engineering designer: the journal of the institution of engineering designers., 32*(1), 30-33.

Schmitt, G. (1993). Case-based design and creativity. *Automation in Construction, 2*(1), 11-19.

Schön, D. A. (1983). *The reflective practitioner: How professionals think in action* (Vol. 5126): Basic books.

Schön, D. A. (1988). Designing: Rules, types and words. *Design Studies, 9*(3), 181-190.

Seebohm, T., & Wallace, W. (1998). Rule-based representation of design in architectural practice. *Automation in Construction, 8*(1), 73-85.

Seepersad, C. C., Pedersen, K., Emblemsvåg, J., Bailey, R., Allen, J. K., & Mistree, F. (2006). The validation square: how does one verify and validate a design method? *Decision Making in Engineering Design, KE Lewis, W. Chen, and LC Schmidt, eds., ASME Press, New York*.

Shah, A., Kerzhner, A., Schaefer, D., & Paredis, C. (2010). Multi-view modeling to support embedded systems engineering in SysML. *Graph transformations and model-driven engineering*, 580-601.

Shah, A., Paredis, C. J., Burkhart, R., & Schaefer, D. (2012). Combining mathematical programming and SysML for automated component sizing of hydraulic systems. *Journal of Computing and Information Science in Engineering, 12*(4), 041006.

Shan, S., & Wang, G. G. (2010). Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions. *Structural and Multidisciplinary Optimization, 41*(2), 219-241.

Shea, K., Aish, R., & Gourtovaia, M. (2003). Towards integrated performance-based generative design tools. *Proceedings of the ECAADE*, 553-560.

Shen, W., & Norrie, D. H. (1999). Agent-based systems for intelligent manufacturing: a state-of-the-art survey. *Knowledge and information systems, 1*(2), 129-156.

Simmons, M. (2013). *Contemporary Practice Seminar*. College of Architecture, Georgia Institute of Technology. Atlanta, USA.

Steinberg, D., Budinsky, F., Merks, E., & Paternostro, M. (2008). *EMF: eclipse modeling framework*: Addison-Wesley Professional.

Stiny, G. (1980). Introduction to shape and shape grammars. *Environment and Planning B, 7*(3), 343-351.

Stiny, G. (1994). Shape rules: closure, continuity, and emergence. *Environment and Planning B, 21*, 49-49.

Stiny, G. (2001). How to calculate with shapes. *Formal engineering design synthesis*, 20-64.

Suhr, J. (1999). *The choosing by advantages decisionmaking system*: Greenwood Publishing Group.

Szokolay, S. V. (2008). *Introduction to Architectural Science*: Routledge.

Tapscott, D., & Williams, A. D. (2008). *Wikinomics: How mass collaboration changes everything*: Penguin.

Tarandi, V., & Froese, T. (2002). Future Directions for IFC-based Interoperability. *ITcon, 8*, 231-246.

Unified Modeling Language®. (2016, 02/01/2016). Retrieved from http://www.uml.org/

Venugopal, M., Eastman, C., Sacks, R., & Teizer, J. (2012). Semantics of model views for information exchanges using the industry foundation class schema. *Advanced Engineering Informatics, 26*(2), 411-428.

Von Neumann, J., & Morgenstern, O. (1945). Theory of games and economic behavior. *Bull. Amer. Math. Soc, 51*(7), 498-504.

Wang, M. Y., Wang, X., & Guo, D. (2003). A level set method for structural topology optimization. *Computer methods in applied mechanics and engineering, 192*(1), 227-246.

Woo, J. C., MJ.; Johnson, RE.; Flores, BE. & Ellis, C. (2004). Dynamic Knowledge Map: reusing experts' tacit knowledge in the AEC industry. *Automation in Construction, 13*(2), 203-207.

Woodbury, Aish, R., & Kilian, A. (2007). *Some patterns for parametric modeling.* Paper presented at the 27th Annual Conference of the Association for Computer Aided Design in Architecture.

Woodbury, R., & Burrow, A. (2006). Whither design space? *AIE EDAM: Artificial Intelligence for Engineering Design, Analysis, and Manufacturing, 20*(02), 63-82.

Woodwark, J. (1986). Generating wireframes from set-theoretic solid models by spatial division. *Computer-Aided Design, 18*(6), 307-315.

Yvars, P.-A. (2010). A Constraint-Based Approach to the Composition Relation Management of a Product Class in Design. *Journal of Computing and Information Science in Engineering, 10*(3), 031002.