

U**Design**

Toward A User-Centered Architecture

by

Yaniv Corem (Ophir)

Bachelor of Architecture, Faculty of Architecture and Town Planning
Technion - Israel Institute of Technology, 2005

Submitted to the Department of Electrical Engineering and Computer Science
and the Department of Architecture
in Partial Fulfillment of the Requirements for the Degrees of

Master of Science in Electrical Engineering and Computer Science
and
Master of Science in Architecture Studies

at the
Massachusetts Institute of Technology
June 2010

© 2010 Massachusetts Institute of Technology
All Rights Reserved

Signature of Author:.....
Department of Electrical Engineering and Computer Science
May 21, 2010

Certified by:.....
Kent Larson
Principal Research Scientist
Department of Architecture
Thesis Supervisor

Certified by:.....
Una-May O'Reilly
Principal Research Scientist
Computer Science and Artificial Intelligence Lab
Thesis Supervisor

Accepted by:.....
Julian Beinart
Professor of Architecture
Chair, Department Committee on Graduate Students

Accepted by:.....
Terry P. Orlando
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Thesis Committee

Kent Larson
Principal Research Scientist
Department of Architecture
Thesis Supervisor

Una-May O'Reilly
Principal Research Scientist
Computer Science and Artificial Intelligence Lab
Thesis Supervisor

Terry Knight
Professor of Design & Computation
Department of Architecture
Thesis Reader

UDesign

Toward A User-Centered Architecture

by

Yaniv Corem (Ophir)

Submitted to the Department of Electrical Engineering and Computer Science
and the Department of Architecture on May 21, 2010
in Partial Fulfillment of the Requirements for the Degrees of
Master of Science in Electrical Engineering and Computer Science
and
Master of Science in Architecture Studies

ABSTRACT

As more companies encourage users to participate in the design of personalized products through online configuration tools, a new kind of user-centered business model emerges. One of the outcomes of this transformation, is the restructuring of a company's products - from a one-size-fits-all to a kit-of-parts - allowing customers to mix-n-match. A similar process is taking place in architectural design, as more research projects and a few commercial applications employ mass-customization techniques to allow users to design and build their own living solutions.

In this thesis, I propose a framework for user-centered architecture, called UDesign, and describe its implementation as a web application that allows users to design their own custom apartment. UDesign includes a sample one-bedroom apartment which users can customize through a kit-of-parts approach, i.e., a catalog of rooms (called assemblies), that can be combined to create a complete floor plan solution. While available configuration tools in architecture require the user to think like an expert, e.g., integrate form and function, UDesign takes a novel approach by deploying a suite of machine learning algorithms coupled with data from Facebook to model users' design preferences and match them with design solutions. Users can take advantage of these recommendations as their design starting point and continue to explore other alternatives by dragging and dropping rooms from the catalog on to the sample floor plan. As users explore design solutions, UDesign updates its recommendations to guide users through the design space and help them find solutions that best fit their needs. Finally, UDesign's integration with Facebook, allows users to share their designs, making UDesign part of their social network.

Thesis Supervisor: Kent Larson
Title: Principal Research Scientist

Thesis Supervisor: Una-May O'Reilly
Title: Principal Research Scientist

Acknowledgments

Kent Larson for taking me under his wing | **Una-May O'Reilly** for taking a chance and opening the door | **Terry Knight** for reading, caring and always being there | **Daniel Smithwick** for watching my back and being a friend | **Yonatan Friedman** for getting me through Machine Learning with a mechanical pencil and a worn-out eraser | **Liraz Greenfeld** for trying to teach me electronics | **Takehiko Nagakura** for teaching me to respect the old Masters | **Aaron Sprecher & Eran Neuman** for the endless support and inspiration | **The Makluba Club** (Duks, Kaustuv, Carl and Daniel) for putting up with me | **The Israeli Mafia** (Yoni, Orit, Tal and Itai) for reminding me how old I am, but keeping me young at heart | **The Fantastic Four** (Natsuki, Chai, Oliver and Gerhard) for getting me wasted and dancing in Helsinki | **Jarmo and the Finns** for the warm hospitality in Helsinki | **The Admins** (Cynthia, Janet and Daniela) for making business feel like pleasure | **Dad** for teaching me math when everyone else gave up | **Mom** for making me the man I am today | **Sis** for never giving up on us | **Bro** for letting me be a big brother | **Gila & Yotam** for being the loves of my life.

UDesign

Toward A User-Centered Architecture

by Yaniv Corem (Ophir)

for Gila & Yotam

Table of Contents

1. Introduction

- 1.1 Yona Friedman
 - 1.1.1. The Information (Short-) Circuit
 - 1.1.2. The Two Loop System
 - 1.1.3. Going Forward
 - 1.1.4. References & Suggested Reading List

2. Prior Work

- 2.1 Smithwick
- 2.2 Duarte
- 2.3 Plewe
- 2.4 Phillips
- 2.5 References & Suggested Reading List

3. Algorithms

- 3.1 Collaborative Filtering
 - 3.1.1. User-Based Recommendations
 - 3.1.2. Item-Based Recommendations
 - 3.1.3. User-Based vs. Item-Based Recommendations
- 3.2 Clustering
 - 3.2.1. Hierarchical Clustering
 - 3.2.2. K-means Clustering
 - 3.2.3. Multidimensional Scaling
 - 3.2.4. Hierarchical Clustering vs. K-means Clustering vs. Multidimensional Scaling
- 3.3 Feature Extraction
 - 3.3.1. Non-Negative Matrix Factorization
- 3.4 Conclusions
- 3.5 References & Suggested Reading List

4. UDesign

- 4.1 A Framework For User-Centered Architecture
- 4.2 Implementation
 - 4.2.1. Data
 - 4.2.2. Personalization
 - 4.2.3. Design
 - 4.2.4. Contribute
 - 4.2.5. Feedback
 - 4.2.6. References & Suggested Reading List

5. Conclusions

- 5.1 Yona Friedman: Reloaded
- 5.2 Representation
- 5.3 Scalability
- 5.4 Collaboration
- 5.5 Data Collection
- 5.6 Recommendations
- 5.7 Usability
- 5.8 Contributions
- 5.9 References & Suggested Reading List

6. Appendix

- 6.1 List of assemblies and solutions

1 | Introduction

This section introduces the work of Yona Friedman in the context of participatory design. More specifically, Friedman's proposal of The Two Loop System, a user-centered design process, is examined and used as the starting point of this thesis.

Yona Friedman

A Hungarian-born French architect, urban planner and designer. In 1958, Friedman published his first manifesto, *Mobile Architecture*, which described a new kind of mobility in architecture - not of buildings but of inhabitants. Mobile architecture was the "dwelling decided on by the occupant" by way of "infrastructures that are neither determined nor determining." Mobile architecture embodies an architecture that responds to a "mobile society." In his writings, Friedman criticizes contemporary architecture for its lack of responsiveness to users and their needs. Friedman argues that the invention of the average user, as a method devised by architects to deal with the complexities of designing for multiple users, is immoral and destructive. In the mid-70's, Friedman published an influential book called *Toward A Scientific Architecture* (Friedman, 1975), where he first voiced his critique of the architectural design process as well as proposed a solution which he called *The Two Loop System*. In Friedman's system, the first loop constitutes a repertoire of "all possible design solutions" that is made available for the user to choose from. The second loop ties the individual user to the community by informing all users of the individual's selection and how it may affect them.

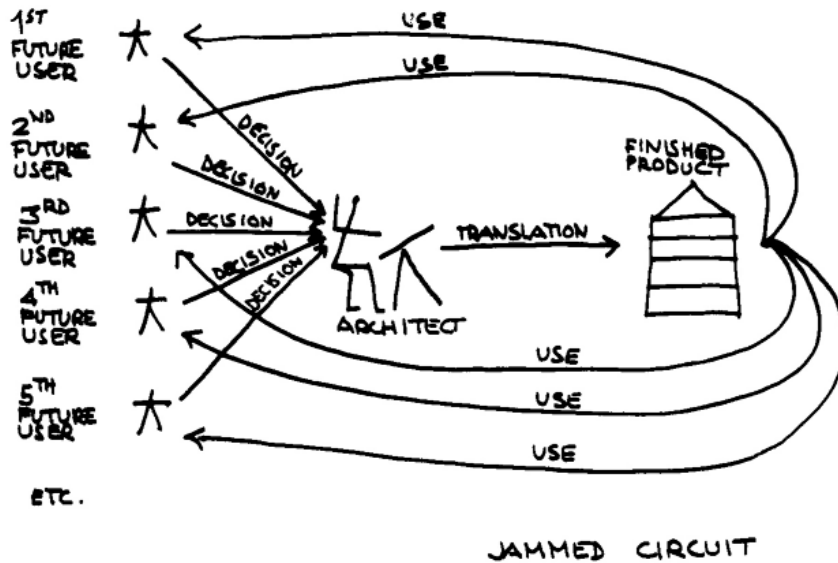


Figure 1.1: Jammed circuit (Friedman, 1975).

The information (short-) circuit

Friedman argues that the architectural design process originated from a very primitive interaction between user and product, in which a user conceived, planned, and executed the production of his home, i.e., the product. The design process was personal, a one-on-one interaction, where a user's decision was informed by the expected use of the finished product. Moreover, responsibility over the outcome was entirely in the hands of the user. With time, some users began building homes for others and a new stakeholder was introduced into the design process - the builder. As the building task became more complex, the builder needed someone who could translate users' needs into building instructions - the architect. By then, the design process was very different from its original, primitive form. It was no longer personal, and included both architect and builder, as mediators between the user and product. Friedman refers to the connecting thread between user and product as the information circuit, i.e., the way in which a user's needs travel as information through the architect, on to the builder, and finally to the product.

The contemporary architectural design process is not much different than Friedman's formulation and the fact that it has remained unchanged for so many years suggests that it works. As a matter of fact, as long as the complexity of the information circuit is low, i.e., an architect having to deal with a handful of users, the design process seems to work quite well. However, as soon as complexity increases (e.g., by adding more users) the design process starts to break down. It simply does not scale well.

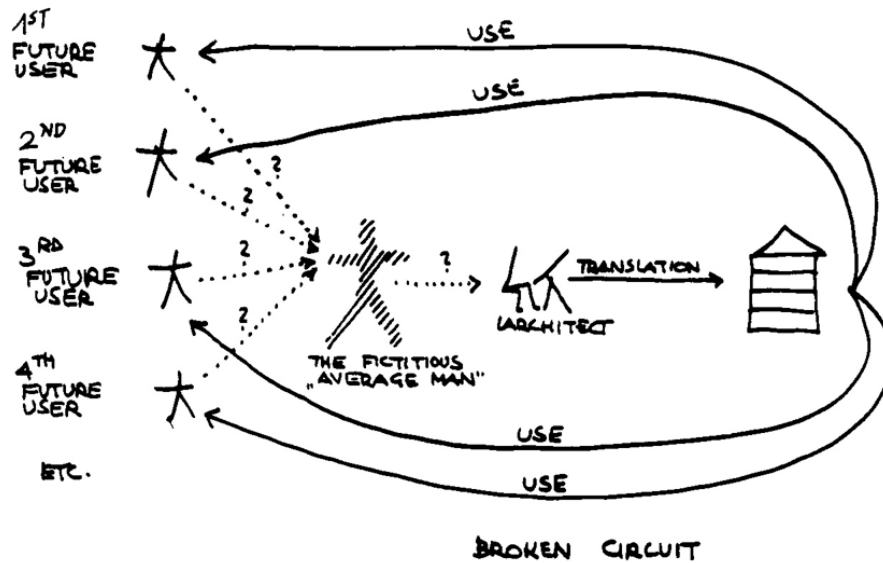


Figure 1.2: Broken circuit (Friedman, 1975).

With multiple users, the architect's job of translating users' demands into building instructions involves a significant amount of resources. For every user, the architect must go back and forth through a lengthy process involving many parts - Q&A, sketching, building models, and more - in order to better understand the user and tailor the product to his needs. Friedman voices the architect's dilemma: increase the number of architects so that there is an architect for every user or come up with an average user that represents the average properties of all users. For lack of a better option, Friedman claims, architects adopted the average user approach.

The Two Loop System

The two loops, proposed by Friedman, is a system which restructures the traditional design process in order to overcome its inherent problem, i.e., the information short-circuit. The first loop in the system involves only the individual user interacting with a repertoire of all possible designs from which he can pick and choose. The repertoire includes a warning system that's embedded into every design to prevent the user from making mistakes (e.g., picking two designs that don't fit). The second loop involves the community of users and is responsible for informing the community of decisions made by the individual that could potentially influence others. Note that Friedman's solution to the information short-circuit is to completely remove the architect from the design process and replace him with an objective and automated component (the repertoire).

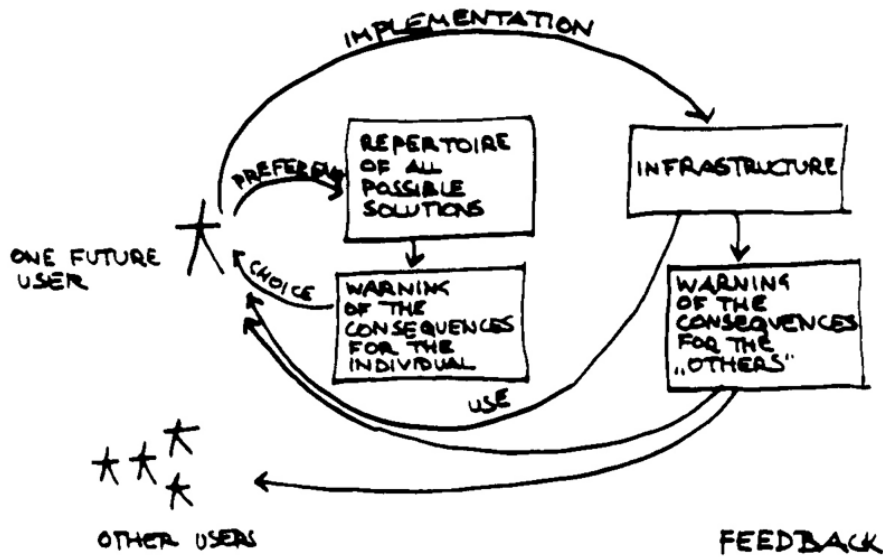


Figure 1.3: The Two Loop System - feedback (Friedman, 1975).

The Two Loop System operates on two levels. The first is local, when a user is presented with a number of different design solutions from which to choose and a warning system that helps to guide his decisions. The second is global, when the design process becomes part of a community with users who are fully aware of the implications of its members' design decisions, the trade-offs, as well as its identity as a whole. After removing him from the design process, Friedman proposes to introduce the architect back into the system by placing him within the first loop as creator and manager of the repertoire. In his new place in the process, the architect is no longer an obstacle between user and product, but instead becomes a facilitator of user-centered design. With the application of the two loops, the design process becomes scalable. Before, the design process would break as the number of users increased, primarily because all information had to go through a single line of communication, resulting in a communication short-circuit. The Two Loop System provides multiple lines of communication, one per user, which are all decoupled, preventing interdependencies that could result in an information short-circuit.

Going Forward

Friedman's work planted the seed that is this thesis. His critique of the architectural design process, raises many questions: What is user-centered architecture? How does it work? How is it better than the current practice? This thesis aims to tackle these difficult questions. The next chapter will give a brief overview of past research conducted at MIT's Department of Architecture concerning user-centered architecture from different perspectives. Chapter three will present a suite of algorithm from machine learning and

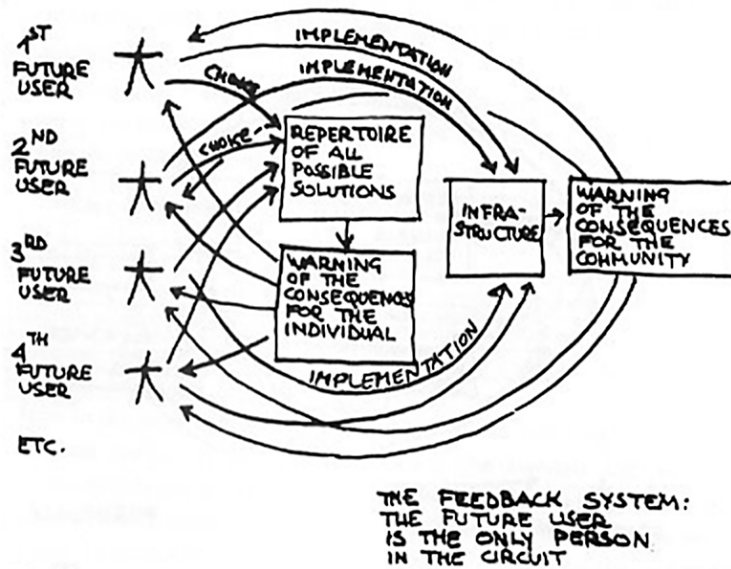


Figure 1.4: The Two Loop System - scalability (Friedman, 1975).

their application to a problem initially posed by Friedman, i.e., matching users to designs. The solution to this problem comes from an unexpected source, i.e., users themselves, where the proposed algorithms learn the design preferences of a user from other, like-minded, users. Then, chapter four describes the implementation of a web application that allows user to create their own custom apartment, inspired in part by Friedman's *Two Loop System*. The web application, UDesign, is a proof-of-concept of user-centered architecture that deploys the algorithms described in chapter three to provide users with recommendations as a way of guiding them through the design process. Finally, the last chapter draws conclusions from various topics discussed in this thesis as well as outlines its contributions.

References & Suggested Reading List

1. http://en.wikipedia.org/wiki/Yona_Friedman (May 20, 2010).
2. Friedman, Y. (1975). *Toward A Scientific Architecture*. MIT Press.

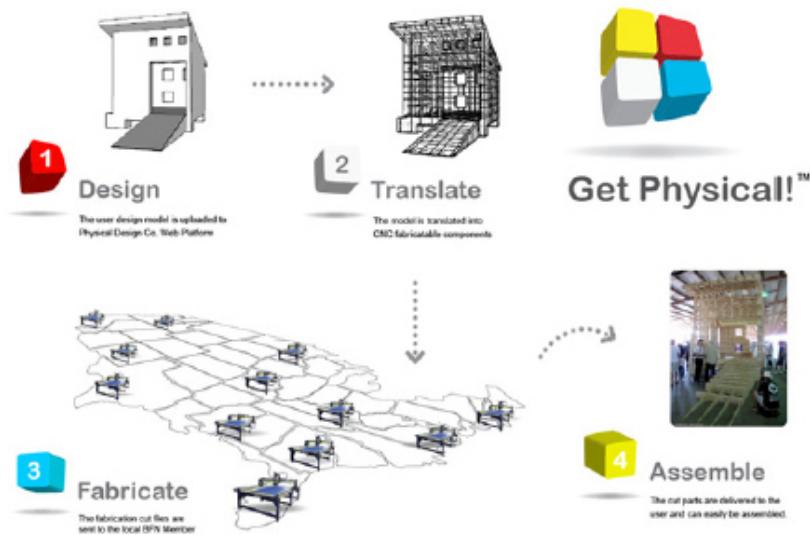


Figure 2.1: Get Physical (Smithwick, 2009).

2 | Prior Work

This chapter presents prior work by four researchers at MIT who have addressed various aspects of user-centered architecture.

Smithwick

Smithwick's work pieces together existing ideas into a cohesive whole outlined by the following scenario: A user joins an open-source online design community where he can: browse designs by other members, design and share his own, send any design to fabrication, and get back a kit-of-parts to assemble, discuss his design or the design of others, participate in joint designs, and so on. The scenario has three underlying components: (1) design - facilitated by tools and software available to all users via the Web for free, (2) fabrication - facilitated by local and small-scale CNC fabrication facilities, and (3) community - facilitated by existing (and increasingly popular) Web 2.0 infrastructure (social networking sites, blogs, etc).

Smithwick's vision shares much in common with the ideas of Yona Friedman as discussed in the first chapter. There is a reference to software and hardware in both Smithwick's and Friedman's work that's joined together by a third, more virtual entity - the Web (Smithwick), or feedback loop (Friedman).

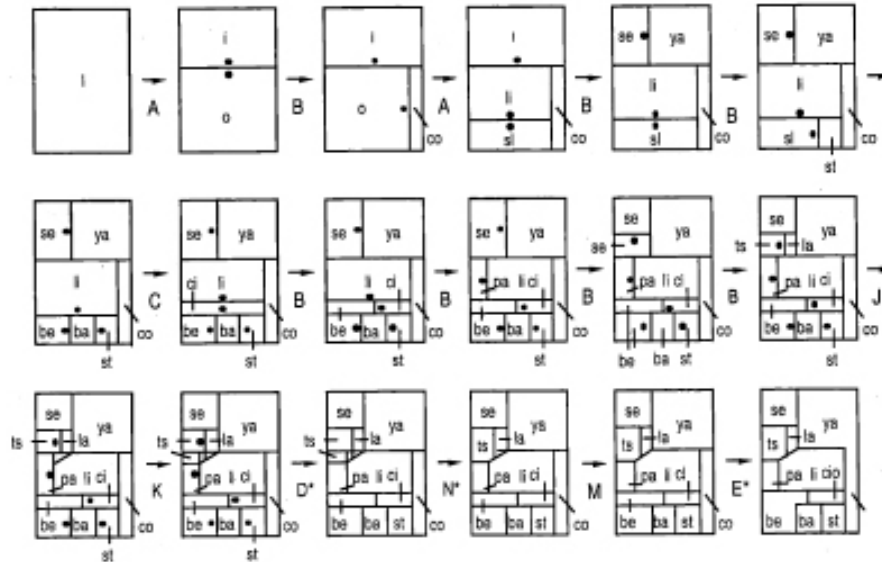


Figure 2.2: Discursive Grammar (Duarte, 2001).

Duarte

Duarte's work describes the process of trying to learn the underlying rules of Alvaro Siza's Malagueira houses through discursive shape grammar. Duarte uses the learned grammar to develop his own user-centered design process to provide mass-customized housing. The work has several interesting aspects:

The first aspect is the ability to learn an architect's design methodology or style. Whether through discursive shape grammars or other methods, Duarte's work shows that (a) there is an underlying pattern in how architects design, and (b) the ability to design systems that capture these patterns is a powerful tool.

The second aspect relates to how this tool is used. For example, an architect's design pattern can be used to create a larger variety of design solutions for users, perhaps way beyond the architect's original repertoire. Furthermore, turning an architect's design pattern into a tool, empowers users to experiment with the design patterns themselves in a safe and predictable environment.

The third aspect is the platform Duarte chooses for the distribution of his system, i.e., the Web. Duarte recognizes the power of the Web as a place where millions of people can explore, share and create designs. The end result of Duarte's work is a website where users can create their own home based on Siza's design pattern.

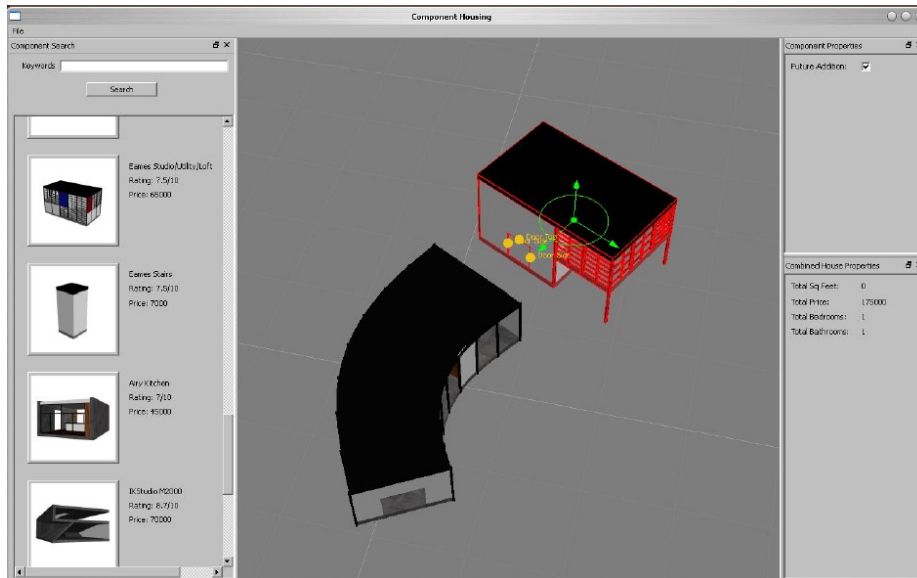


Figure 2.3: Screenshot of H++ (Plewe, 2008).

Plewe

Plewe’s work presents a bricolage approach to designing mass-customized housing for low to middle income families as an alternative to the existing tract home paradigm. As a proof-of-concept, Plewe developed a design tool, which included a 3D environment, a library of components and a way for users to search the library and find what they need. Users who wanted to create their own custom home would use Plewe’s tool to search the library by using keywords like “concrete”, find the components they liked, add them to the 3D authoring environment, adjust the components until they’re satisfied, and finally create construction documents for the contractor.

Plewe’s approach - the library of components, search and selection as a process of design, and an emphasis on what the user wants - provides the technical base for the work described in this thesis.

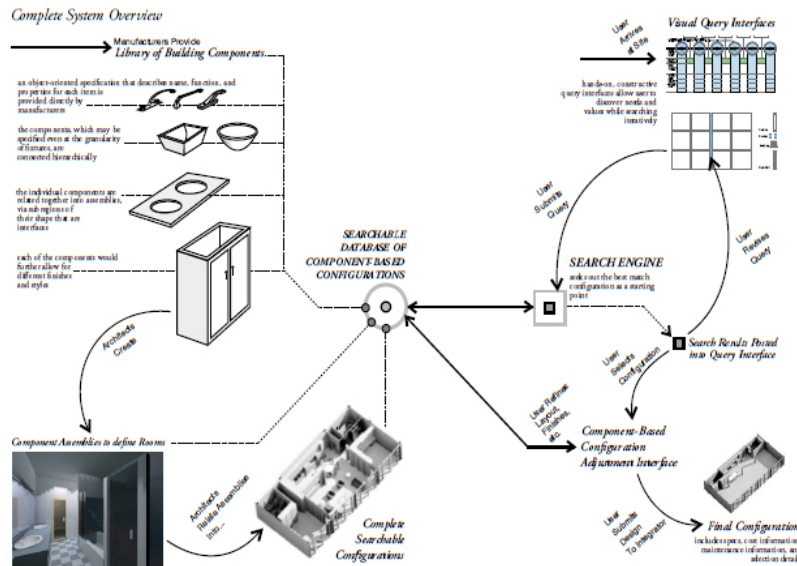


Figure 2.4: Design By Searching (Phillips, 2007).

Phillips

Phillips's work defines design as a process of searching, and provides the conceptual and technical framework to support search in Web-based design tools for everyday users. The work is motivated by the notion of user-centered design – users creating their own living solutions - and the fabrication abilities to support it. Phillips proposes a way of combining Building Information Modeling (BIM), modern search methods and an Object Oriented Programming (OOP) approach to create a space of design solutions that “can be searched directly, without indexing.”

References & Suggested Reading List

1. Smithwick, D. (2009). Architectural Design 2.0: An Online Platform for the Mass Customization of Architectural Structures. Masters Thesis. Massachusetts Institute of Technology, Cambridge, Massachusetts.
2. Duarte, J. (2001). Customizing Mass Housing: A Discursive Grammar for Siza's Malagueira Houses. PhD Thesis. Massachusetts Institute of Technology, Cambridge, Massachusetts.
3. Plewe, T. (2008). Besting the Tract Home: A software-Based Bricolage Approach to Affordable Custom Housing. Masters Thesis. Massachusetts Institute of Technology, Cambridge, Massachusetts.
4. Phillips, G. (2007). Design By Searching: A System for Creating and Evaluating Complex Architectural Assemblies. Masters Thesis. Massachusetts Institute of Technology, Cambridge, Massachusetts.

3 | Algorithms

Within user-centered architecture, a set of critical tasks is the ability to model users, designs, and the relationship between them. To tackle these difficult tasks, this section explores the following machine learning algorithms by way of experimentation and considers their potential contribution to the tasks at hand:

- Collaborative Filtering (User-based / Item-based) - is used to match users to design solutions by way of recommendations.
- Clustering (Hierarchical / K-means / Multidimensional Scaling) - is used to discover groups of users and design solutions.
- Feature Extraction (Non-Negative Matrix Factorization) - is used to uncover the latent features of a design solution.

For the purpose of these experiments, a unique data set was created with the help of 30 architects (referred to as *critics*) who were asked to rate, on a scale of 1 (low) to 5 (high), 20 architectural floor plans of a one bedroom apartment. Figure 3.1 shows a sample page from a booklet given to each critic containing the floor plan and rating box. Critics' ratings were collected into a 30 by 20 matrix (referred to as the rating matrix) with rows representing critics and columns representing floor plans (Figure 3.2).

The highest average floor plan rating was 3.83 and the lowest was 2.37. The highest average critic rating was 3.9 and the lowest 1.25. The results indicate that most critics rated according to the highest average floor plan rating, and chose to rate higher than the lowest average floor plan rating. However, critic_9 gave all but three floor plans a rating of 1, bringing the average of the lowest critic rating down. Also, the standard deviation for critic_9's ratings (0.64) suggests an outlier and was excluded from the experiments.

Figure 3.1: How would you rate this floor plan?



Plan #: 02.03

Attributes

- 1. adjacency
- 2. ventilation
- 3. natural light
- 4. storage
- 5. accessibility
- 6. connectivity
- 7. functionality
- 8. average room size
- 9. views / visual connections
- 10. flexibility
- 11. intimate
- 12. partitioned
- 13. open
- 14. symmetry
- 15. order
- 16. privacy
- 17. social
- 18. spacious
- 19. flow
- 20. efficiency

Add your own:

RATING:

Figure 3.2: Rating matrix.

Critic	SOL19	SOL25	SOL33	SOL38	SOL54	SOL59	SOL74	SOL81	SOL125	SOL139	SOL144	SOL149	SOL150	SOL158	SOL169	SOL202	SOL212	SOL224	SOL228	SOL268	Average Critic Rating	Standard Deviation	
Critic_1	4	3	4	2	1	1	4	4	3	3	4	5	4	3	5	3	3	2	2	2	5	3.25	1.21
Critic_2	2	3	4	3	2	2	4	4	3	3	4	4	4	2	2	5	4	2	4	3	5	3.35	0.99
Critic_3	4	5	3	2	2	2	4	4	3	3	5	1	1	3	5	5	5	5	5	3	2	3.5	1.4
Critic_4	2	3	4	3	1	2	3	2	4	4	4	5	3	3	5	2	2	3	3	3	5	3.15	1.14
Critic_5	4	5	3	3	2	2	3	2	1	1	4	3	1	3	1	2	2	3	4	1	1	2.5	1.19
Critic_6	5	5	4	5	4	4	4	4	4	3	1	1	2	3	3	2	2	3	3	3	3	3.25	1.21
Critic_7	5	5	2	1	1	2	5	1	2	1	4	5	5	2	5	1	2	4	1	4	1	2.9	1.71
Critic_8	5	2	1	1	3	2	4	2	3	2	1	1	1	4	4	1	2	4	5	4	4	2.6	1.43
Critic_9	3	3	1	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1.25	0.64
Critic_10	5	5	4	3	2	5	5	3	4	3	3	4	1	2	2	5	4	4	3	2	2	3.45	1.23
Critic_11	3	4	4	3	4	3	4	4	3	4	2	2	2	4	1	4	3	2	2	1	1	2.95	1.05
Critic_12	4	4	3	2	3	3	3	3	2	3	3	4	5	3	4	3	4	4	2	4	4	3.25	0.79
Critic_13	4	4	2	2	1	1	5	4	1	1	3	3	1	1	1	4	3	4	3	2	2	2.5	1.36
Critic_14	3	4	5	3	3	4	2	3	4	4	2	3	4	3	4	2	3	1	3	3	3	3.15	0.93
Critic_15	5	5	3	4	4	4	4	4	4	3	3	4	2	3	3	4	5	3	4	3	3	3.7	0.8
Critic_16	3	3	4	1	5	3	4	4	4	5	3	3	2	3	4	2	2	4	4	5	3	3.4	1.1
Critic_17	4	4	3	2	1	1	4	3	2	1	2	3	1	3	2	4	5	4	4	2	2	2.75	1.25
Critic_18	5	5	2	3	3	3	5	4	3	1	2	3	1	1	2	3	4	3	4	1	1	2.9	1.33
Critic_19	5	5	2	3	3	3	5	5	3	3	4	5	2	3	3	2	2	3	2	2	3.25	1.16	
Critic_20	3	4	2	2	3	3	4	3	3	2	2	2	3	3	1	3	3	5	4	2	2	2.85	0.93
Critic_21	1	2	3	3	2	2	4	3	2	1	3	2	4	2	3	3	2	3	2	3	2	2.5	0.83
Critic_22	4	3	1	1	1	1	3	3	1	1	4	5	4	2	4	3	3	4	2	4	4	2.7	1.34
Critic_23	4	3	3	2	2	2	2	3	3	4	3	3	4	2	4	2	2	2	2	3	3	2.8	0.77
Critic_24	3	3	4	4	3	3	3	2	4	4	3	3	4	5	5	3	4	4	3	4	3	3.5	0.76
Critic_25	3	4	3	2	3	3	3	3	2	2	3	4	1	2	2	3	3	3	2	1	1	2.6	0.82
Critic_26	5	3	5	4	3	2	3	4	4	3	2	4	4	2	4	4	5	3	5	2	2	3.55	1.05
Critic_27	3	3	4	4	2	3	3	4	3	4	4	4	2	2	4	4	3	2	3	4	4	3.25	0.79
Critic_28	4	4	2	3	2	2	3	4	4	1	2	4	5	5	5	3	4	2	2	2	2	3.25	1.21
Critic_29	3	4	4	3	3	3	4	4	3	3	4	5	3	3	4	4	5	4	5	4	5	3.9	0.79
Critic_30	5	5	4	1	1	2	3	2	3	3	2	1	1	2	5	1	2	2	5	5	2	2.75	1.55
Average Floor Plan Rating	3.77	3.83	3.1	2.57	2.37	2.47	3.63	3.2	2.87	2.57	2.9	3.23	2.67	2.67	3.4	2.97	3	3.2	3.13	2.93			
Standard Deviation	1.07	0.95	1.12	1.07	1.1	1.01	0.89	1	1.01	1.22	1.06	1.36	1.52	0.92	1.48	1.22	1.08	1.1	1.17	1.41			

Collaborative Filtering

Collaborative Filtering (CF) is a method for making automatic predictions (filtering) about the interest or taste of a user by collecting and analyzing the personal preferences of many other users (collaborative). The first system to use CF was the Information Tapestry project at Xerox PARC (Goldberg et al., 1992). In general, CF systems can be categorized as user-based or item-based. User-based CF systems do the following:

- Look for users who share the same rating pattern with the active user (i.e., the user for whom we are trying to make a prediction).
- Use the ratings from those like-minded users to formulate a prediction for the active user.

Alternatively, item-based CF systems focus on items, and the relationships amongst them, to make predictions, e.g., “Users who bought X also bought Y” (popularized by Amazon.com). Item-based CF systems do the following:

- Build a Similarity Matrix to determine the relationship between pairs of items.
- Use the Similarity Matrix, and user’s personal data, to formulate a prediction.

```

C: Critic
M: Rating Matrix
U: Active User
For each C in M,
    If C is not U,
        Then compute Similarity of C and U, (see Eq. 3.1)
For each Item in M,
    If Item was not rated by U,
        Then compute user similarity weighted average rating, (see Eq. 3.2)
    And store it in List,
Normalize List,
Sort List based on normalized weighted similarity,
Return List

```

Figure 3.3: Pseudocode for user-based recommendation.

User-Based Recommendations

When we're interested in finding good recommendations, we tend to turn to people with a similar taste to ours. User-based collaborative filtering takes the same approach by comparing us, the active user, with everyone else, finding those who are most similar and using their preferences to make a prediction about what we like. In user-centered architecture, this approach takes advantage of design preferences and taste expressed by various users to help other users find what they need. To illustrate how user-based collaborative filtering works, a brief experiment was conducted using the data set of floor plan ratings to predict a rating for a fictitious user (referred to as Critic_N) who rated some but not all the floor plans.

The algorithm (Segaran, 2007) presented in pseudocode (Figure 3.3) starts by iterating through all users and computing their similarity to Critic_N using a Pearson correlation coefficient (Eq. 3.1). Then, the algorithm iterates through every item, i.e., floor plan, and computes the weighted average rating (Eq. 3.2) by multiplying each of the ratings by the critics's similarity score and summing over all the scores. The intuition behind weighting the ratings is that users who are more like Critic_N will have more weight put on their ratings, whereas ratings by users who are different will receive very little weight. Finally, the algorithm normalizes, sorts, and returns the result as predicted ratings for Critic_N (Figure 3.4).

Critic	SOL9	SOL25	SOL33	SOL38	SOL54	SOL59	SOL74	SOL81	SOL125
Critic_N	4.1	5	3.08	2.61	2.36	2.74	3.81	2	2.76
	SOL139	SOL144	SOL149	SOL150	SOL158	SOL169	SOL202	SOL212	
	2.36	2.97	3.27	2.37	1	3.15	3.03	3.25	
	SOL224	SOL228	SOL268						
	3.47	3.28	2.77						

Figure 3.4: Predicted ratings (in green) and initial ratings (in yellow) for Critic_N.

(Eq. 3.1)

$$w_{a,u} = \frac{\sum_{i=1}^m (r_{a,i} - \bar{r}_a)(r_{u,i} - \bar{r}_u)}{\sigma_a \sigma_u}, \text{ where } r_a \text{ is the active user and } r_u \text{ is the neighbor}$$

(Eq. 3.2)

$$\bar{r} = \frac{\sum_{i=1}^n w_i r_i}{\sum_{i=1}^n w_i}, \text{ where } w_i \text{ is the weight and } r_i \text{ is the rating}$$

(Eq. 3.3)

$$\text{Given } \theta_1 = \begin{bmatrix} x_{1,1} \\ x_{1,2} \\ \vdots \end{bmatrix} \text{ and } \theta_2 = \begin{bmatrix} x_{2,1} \\ x_{2,2} \\ \vdots \end{bmatrix}, \text{ RMSE}(\theta_1, \theta_2) = \sqrt{\text{MSE}(\theta_1, \theta_2)} = \sqrt{\frac{\sum_{i=1}^n (x_{1,i} - x_{2,i})^2}{n}}$$

In order to measure the predictive power of the proposed algorithm, the ratings matrix was transformed into a test set by randomly deleting ratings in critics 1 through 8. The missing ratings were then estimated and the Root Mean Square Error (Eq. 3.3) was computed, yielding a value of 1.14 (Figure 3.5). To evaluate this result (1.14), RMSE was calculated ten more times substituting missing ratings with random numbers between 1 and 5, yielding an average value of 1.95. The results show that the algorithm makes predictions which are better than chance with an error that is 58% of the random error. In addition, RMSE was calculated for Critic_30 with 5, 10, and 15 ratings yielding 1.31, 1.24, and 0.87, respectively. These results indicate that by rating more floor plans, critics improve the accuracy of the recommendations they get.

Figure 3.5: Test matrix:
 Predicted ratings (in green) and initial ratings (in yellow) for critics 1 through 8, used to calculate algorithm's predictive power by measuring the Root Mean Square Error (RMSE).

Critic	SOL19	SOL25	SOL33	SOL38	SOL54	SOL59	SOL74	SOL81	SOL125	SOL139	SOL144	SOL149	SOL150	SOL158	SOL169	SOL202	SOL212	SOL224	SOL228	SOL268
Original Ratings																				
Critic_1	4	3	4	2	1	1	4	4	3	3	4	5	4	3	5	3	3	2	2	5
Critic_2	2	3	4	3	2	2	4	4	3	3	4	4	4	2	5	4	2	4	3	5
Critic_3	4	5	3	3	2	2	4	4	3	3	5	1	1	1	3	5	5	5	5	2
Critic_4	2	3	4	3	1	2	3	2	4	4	4	5	3	3	5	2	2	3	3	5
Critic_5	4	5	3	3	2	2	3	2	1	1	4	3	1	3	1	2	2	3	4	1
Critic_6	5	5	4	5	4	4	4	4	4	3	1	1	2	3	3	2	2	3	3	3
Critic_7	5	5	2	1	1	2	5	1	2	1	4	5	5	2	5	1	2	4	1	4
Critic_8	5	2	1	1	3	2	4	2	3	2	1	1	1	4	4	1	2	4	5	4
Test Set																				
Critic_1	4	3	4	2	2.23	2.43	3.17	2.74	2.82	3.25	3.11	3.44	3.16	2.72	4.25	2.14	3	2	2	5
Critic_2	2	3	4	3	2	2	4	4	2.74	2.4	3.35	3.46	4	2	5	4	3.2	3.24	3.19	3.49
Critic_3	3.41	3.82	3.23	2.53	2	2	4	4	3	3	5	1	1	3	5	5	2.98	3.02	3.42	3.49
Critic_4	2	3	4	3	1	2	3	2	2.83	2.97	3.57	3.08	3.06	2.31	4.54	3.28	2	3	3	5
Critic_5	3.8	3.99	2.55	2.34	2	2	4.2	3.2	1	1	4	3	2.75	2.14	3.04	3.14	2	3	4	1
Critic_6	5	5	4	5	2.56	2.82	3.82	3.37	2.98	2.68	2.98	3.61	2.49	2.8	3.23	2.84	2	3	3	3
Critic_7	5	5	2	1	1	2	5	1	2	1	4	5	5	2	5	1	2.88	3.34	2.96	3.08
Critic_8	5	2	1	1	3	2	4	2	2.79	2.56	3.16	3.39	1	4	4	1	2.84	3.07	3.09	3.34

Pre-compute item similarity matrix with user-based approach but substituting items for users by transposing the rating matrix.

For each Item1 in User's items,

For each Item2 in Item1's similarity matrix,

If User has already rated Item2, Then continue to the next iteration of the loop,

Compute item similarity weighted rating, (see Eq. 3.2)

Compute average of weighted item ratings,

And save them in List,

Sort List based on normalized weighted item similarity,

Return List

Figure 3.6: Pseudocode for item-based recommendation.

Item-Based Recommendations

When a data set like the rating matrix becomes very large, a user-based approach might start to break down due to the computational cost of users' pairwise comparison. An item-based approach alleviates this problem by pre-computing a similarity matrix for all items. The assumption is that items do not change as frequently as users, therefore, the similarity matrix will only need to be updated every once in a while. Amazon.com, for example, uses an item-based approach, i.e., item-to-item collaborative filtering (Linden et al., 2003), to recommend products to its users (Figure 3.6).

Making recommendations using item-based collaborative filtering consists of the following steps: pre-computing an item-to-item similarity matrix, looking up the active user's top-rated items, and creating a weighted list of the most similar items. To illustrate how this works, an algorithm (Segaran, 2007) was tested with the ratings matrix to get recommendations for Critic_N (Figure 3.7).

Critic_N's rated items:

(5.0, SOL25), (2.0, SOL81), (1.0, SOL158)

Critic_N's recommended items:

(5.0, SOL9), (3.65, SOL74), (3.24, SOL144), (2.85, SOL202), (2.81, SOL228), (2.67, SOL224), (2.49, SOL212), (2.43, SOL33), (2.09, SOL125), (1.5, SOL38), (1.42, SOL59), (1.41, SOL54), (1.0, SOL139)

Figure 3.7: Item-based recommendations for Critic_N



Figure 3.8: SOL9 (Left) and SOL25 (Right) with a similarity score of 0.04.

Figure 3.7 shows that given Critic_N's high rating of floor plan SOL25, the algorithm recommended SOL9 as the highest ranking (best matching) item. Figure 3.8 validates the algorithm's precision and predictive power as the two floor plans are nearly identical. There are some minor differences between the two floor plans, for example, SOL9 has a bar-top dining table, but SOL25 has a 6-seat standalone dining table. However, the overall similarity of the two floor plans is evident and emerges from the users' ratings.

User-Based vs. Item-Based Recommendations

In order to compare the performance of the algorithm, a random critic was selected (Critic_30), all but the first five of his ratings deleted, and then predicted by each of the algorithms. The RMSE for the user-based algorithm was 1.18, and the item-based algorithm was 1.61 indicating that user-based recommendations are more accurate than item-based recommendations. However, there are other issues to consider besides accuracy when choosing one algorithm over the other, for example, size of data set, and frequency of change have a tremendous influence on the performance of these algorithms.

```

Initialize all rows to be clusters,
Create BigCluster,
While number of Clusters is bigger than 1,
    For each pair of Clusters,
        Find the pair with the smallest distance,
        Calculate the average of the two Clusters,
        Combine the two clusters to form a new Cluster,
    Add new Cluster to BigCluster
Return BigCluster

```

Figure 3.9: Pseudocode for hierarchical clustering.

Clustering

Clustering, or Cluster Analysis, is a way of discovering and visualizing groups of things - people, news stories, blogs, and more. This section will focus on two clustering techniques - Hierarchical and K-means - and apply them to the ratings matrix in order to discover groups of critics and floor plans. In user-centered architecture, these clusters can be utilized, for example, to infer a new user's design preferences by finding his cluster membership. A user who has been assigned to a certain cluster, shares the design preferences of that cluster, making it possible to tailor design recommendations to his taste.

Hierarchical Clustering

Figure 3.9 is the pseudocode for a simple hierarchical clustering algorithm (Segaran, 2007). The algorithm starts by assigning each item, i.e., floor plan, to a cluster of its own. Then, the distance between every pair of clusters is calculated and the closest (most similar) ones are merged together to form a new cluster. The process is repeated until there is only one cluster left. The algorithm uses the Pearson correlation coefficient to compute the distance between clusters.

Figure 3.10 shows the groups of critics discovered when running the hierarchical clustering algorithm on the rating matrix. The dendrogram clearly separates the critics into two groups, each characterized by its own design preferences as extracted from the ratings. Each of the two main clusters contains smaller clusters, for example, critics 16, 8, 30, 6, 11, and 3, form a cluster within the bigger cluster at the top of Figure 3.10. Rated floor plans were also clustered (Figure 3.11) by running the hierarchical clustering algorithm on a rotated rating matrix. Clustering shows two distinct clusters made up of smaller clusters such as floor plans SOL9, 25, 74, 224, and 228 (Figure 3.12).

Figure 3.11: Hierarchical clustering of floor plans.

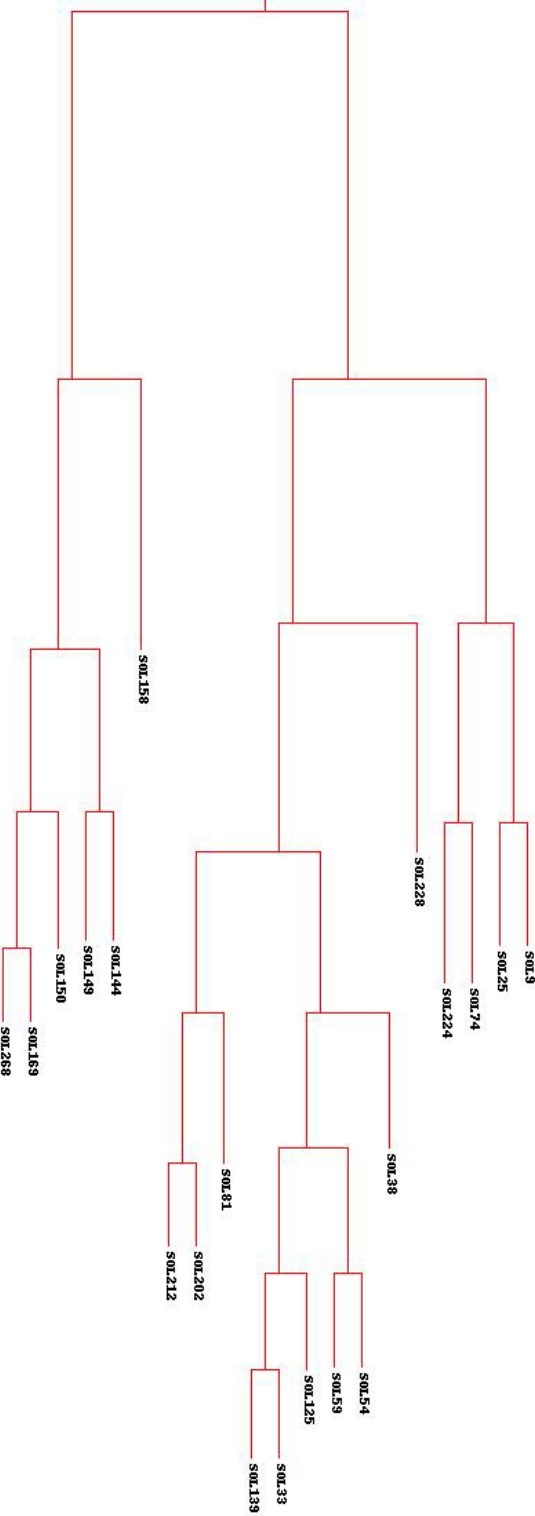


Figure 3.12: Clustered floor plans.



Attributes of clustered floor plans (partial list):

- A small bathroom
- A large kitchen
- A dining table
- Clearly defined private & public spaces

M: Rating Matrix

For each Item in M,

Compute the Min and Max value of Item,

Create K randomly placed centroids,

While assignment of Item is changing,

Find the closest centroid to Item,

Assign Item to centroid,

Move centroid to average location of Items

Return K Clusters

Figure 3.13: Pseudocode for K-means clustering.

K-means Clustering

Hierarchical clustering is computationally expensive because the pairwise distance calculations have to be carried out every time clusters are merged, therefore, this method might break down on larger data sets. K-means clustering takes a different approach because the number of clusters is specified in advance removing some of the ambiguity in dividing up the data.

The K-means clustering algorithm (Segaran, 2007) presented as pseudocode (Figure 3.13) starts by randomly placing centroids, i.e., center of the cluster, and assigning every item to the nearest one. Then, each centroid is shifted to the average location of the items it contains, items are reassigned and the process repeats itself until the assignments stop changing.

Figure 3.14 shows the K-means clustering of floor plans with different values of K.

K=8 SOL(158, 169, 268), (144, 149), (9,25), (74), (33), (150), (202, 212,224, 228), (38, 54, 59, 81, 125, 139)

K=5 SOL(54, 59, 125, 139, 158), (25, 74, 144, 202, 224), (33, 228, 268), (149, 150, 169), (9, 38, 81, 212)

K=3 SOL(9, 25, 74, 228), (38, 81, 144, 149, 150, 202, 212, 224), (33, 54, 59, 125, 139, 158, 169, 268)

Figure 3.14: K-means clustering of floor plans.

M: Rating Matrix

For each Item1 and Item2 in M,

Compute the real distance between Item1 and Item 2,

Randomly place Item1 and Item2 in 2D space

While TotalError is changing,

Find the projected distance between Item1 and Item2,

Calculate Error between real and projected distance

Move Item1 and Item2 in proportion to Error

Update TotalError

Return location of Items

Figure 3.14: Pseudocode for Multidimensional scaling.

Multidimensional Scaling

Items in the rating matrix are multidimensional, i.e., each floor plan is represented by a vector of 30 ratings. Multidimensional scaling is a technique for finding a two-dimensional representation of the rating matrix which is sometimes easier to understand than a dendrogram.

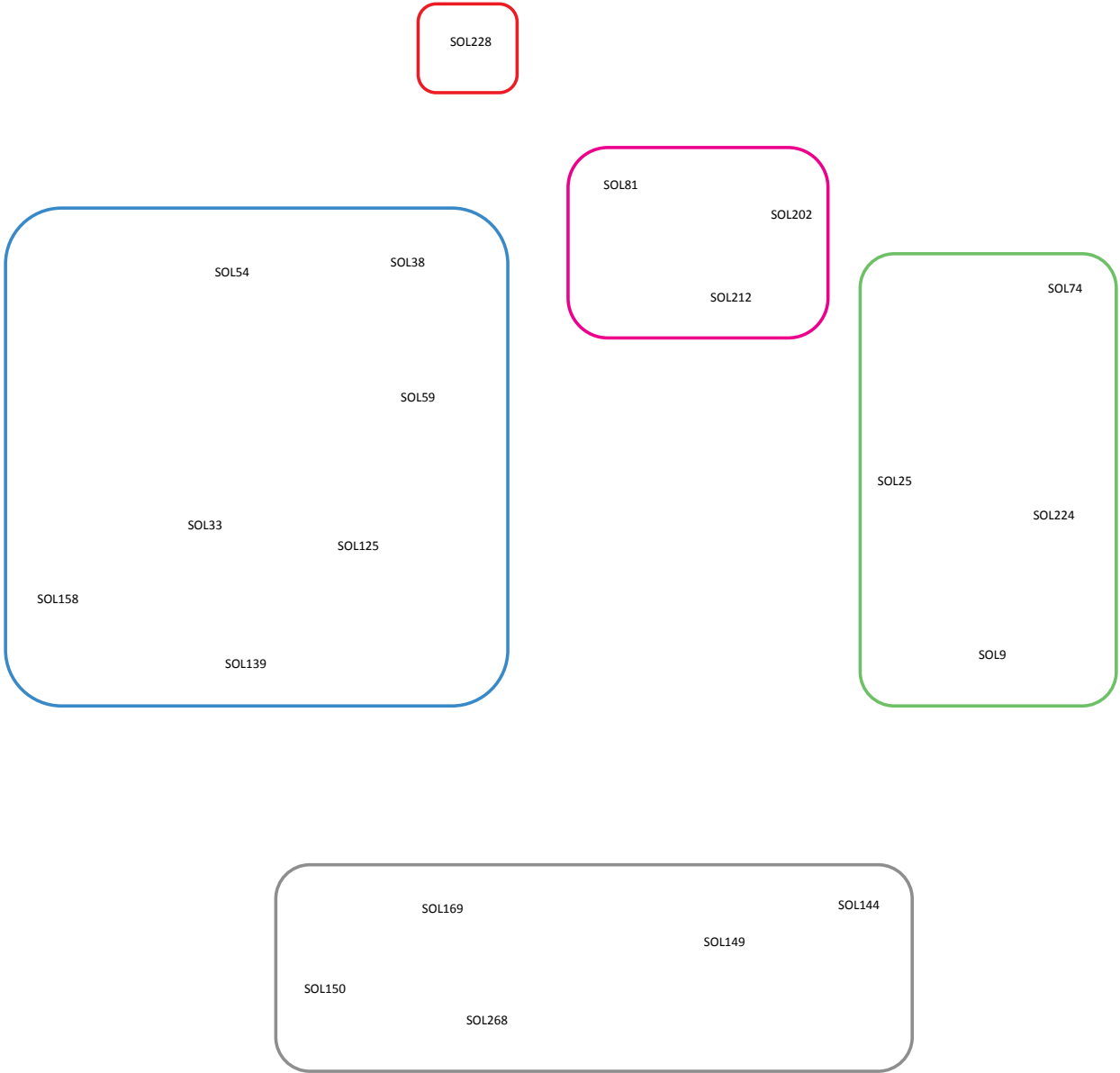
The algorithm (Segaran, 2007) presented as pseudocode (Figure 3.14) starts by calculating the target distances between all the items. Next, all items are placed randomly on a two-dimensional plane. The current distances between all the items are calculated using the actual distance, i.e., the sum of the squared differences. For every pair of items, the target distance is compared to the current distance to calculate the error. Every item is moved a small amount in proportion to the direction of the error between the two items. The process is repeated until the total error cannot be reduced by moving any of the items.

Figure 3.15 shows the results of running the multidimensional scaling algorithm on the rating matrix.

Hierarchical Clustering vs. K-means Clustering vs. Multidimensional Scaling

All three methods, when applied to the floor plans, produced comparable results, making the choice of any one method unclear. This is due to the small size of the data set and some ad-hoc choices of parameters for the algorithms. With a more extensive data set one expects to evaluate and choose based upon computational cost of the algorithm, motivation for and use of the derived information, and auxiliary information from the problem domain.

Figure 3.15: Multidimensional scaling of rating matrix.
Colored boxes correspond to the hierarchical clustering in Figure 3.11



Feature 4	(SOL81, SOL224)
Feature 5	(SOL149, SOL150, SOL169, SOL268)
Feature 7	(SOL202, SOL212)
Feature 8	(SOL158, SOL228)
Feature 9	(SOL33, SOL54, SOL59, SOL125, SOL139)

Figure 3.17: 5 features extracted from the rating matrix.

Conclusions

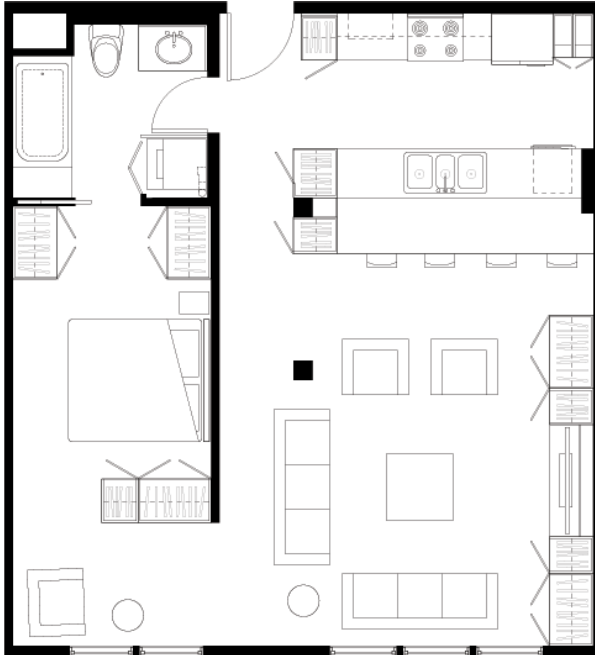
The dataset used in these experiments is extremely small compared to the Netflix or Amazon data sets. Nevertheless, each of the applied methods revealed some interesting information about the data - critics, floor plans, and the relationship between them.

Collaborative filtering, whether user or item based, showed that it is possible to leverage a community's opinion, expressed through ratings, in order to help an individual find what he likes. Analysis of the algorithm showed that it performs much better than a random guess.

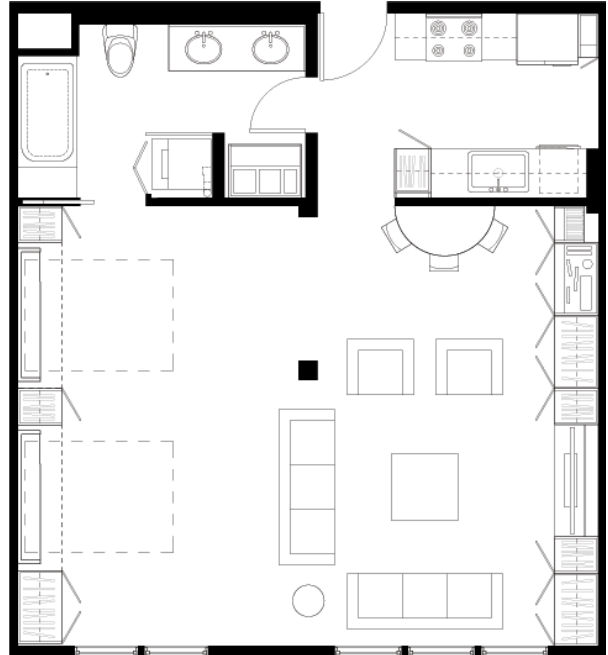
Clustering - Hierarchical, K-means, or Multidimensional Scaling - showed that it is possible to find groups of critics and floor plans, and uncover their common characteristics. With a large enough data set, clustering can be used to extract user and design typologies which can help stakeholders in the building industry - contractors, developers, architects, etc. - tailor their services to clients' needs.

Feature Extraction showed that it is possible to extract the features of a floor plan in order to create a design vector. The design vector can be used to score floor plans based on a user's interest in one or all of its features allowing the user to explore the design space in multiple directions.

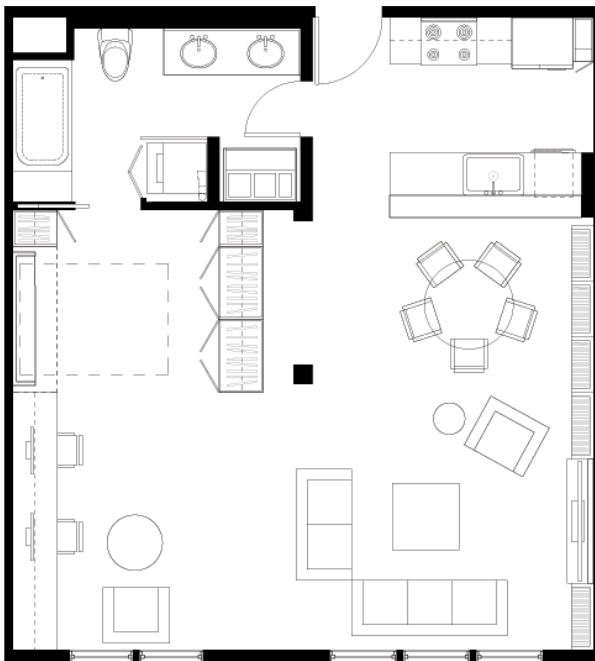
Figure 3.18: Articulating Feature 5 through its floor plans.
Feature 5 could be the free-standing column that appears in all the floor plans below.



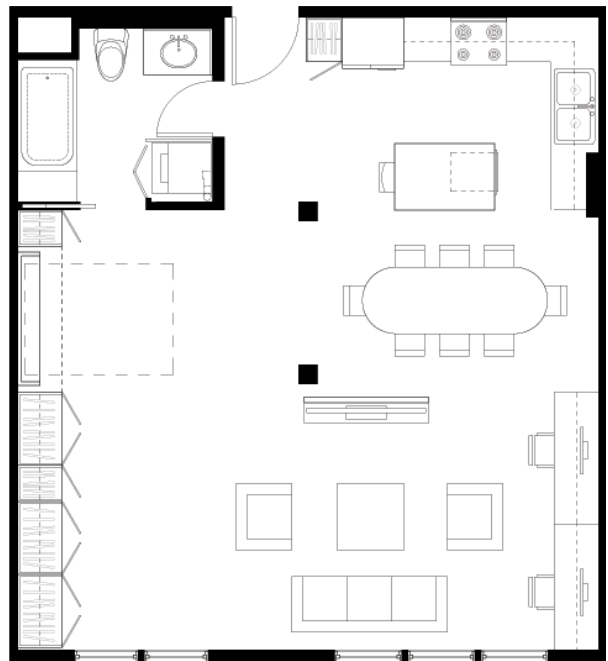
SOL149



SOL150



SOL169



SOL268

References & Suggested Reading List

1. Billsus, D., & Pazzani, M. J. (1998). Learning collaborative information filters. Proceedings of the 15th International Conference on Machine Learning, pp. 46–54.
2. Blei, D. M., Ng, A. Y., & Jordan, M. I. (2002). Latent Dirichlet allocation. *Advances in Neural Information Processing Systems* 14.
3. Breese, J. S., Heckerman, D., and Kardie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the 14th Conf. on Uncertainty in Artificial Intelligence, pp. 43–52.
4. Goldberg, D., Nichols, D., Oki, B.M., & Terry, D. (1991). Using Collaborative Filtering to Weave an Information Tapestry. *Comm. ACM*, vol. 35, no. 12, pp. 61-70.
5. Herlocker, J.L., Konstan, J.A., Borchers, A., and Riedl, J. (1999). An Algorithmic Framework for Performing Collaborative Filtering. Proc. 22nd Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '99).
6. Herlocker, J.L., Konstan, J.A., Terveen, L.G., and Riedl, J.T. (2004). Evaluating Collaborative Filtering Recommender Systems. *ACM Trans. Information Systems*, vol. 22, no. 1, pp. 5-53.
7. Hofmann, T. (2004). Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems*, 22, pp. 89–115.
8. Koren, Y., Bell, R., Volinsky, C. (2009). Matrix Factorization Techniques for Recommender Systems. *IEEE Computer*, 08/2009, Volume 42, Issue 8, pp.30-37.
9. Linden, G., Smith, B., and York, J. (2003). Amazon.com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Computing*.
10. Marlin, B. (2004). Collaborative filtering: A machine learning perspective. Master's thesis, University of Toronto, Computer Science Department.
11. Segaran, T. (2007). *Programming Collective Intelligence*. O'Reilly Media.
12. Srebro, N., Rennie, J. D. M., & Jaakkola, T. S. (2005). Maximum margin matrix factorization. *Advances in Neural Information Processing Systems* 17.

4 | UDesign

When a user wishes to purchase an apartment from a developer, he is usually presented with a few sample apartments from which to choose. These sample apartments usually offer very few parameters that can be changed because they are designed to fit the taste and lifestyle of an average user. The solution to this problem can be found in two different places - hardware and software - which complement each other to create user-centered architecture. On the one hand, there is hardware, the building technology that is responsible for making sure users' designs are feasible. The mass-customization community in architecture has made considerable progress in recent years and managed to push fabrication technology forward. On the other hand, there is software, the interface that is responsible for making sure users can express their wants and needs. The ability of an interface to meet users' needs depends on its front-end - how easy it is to use the interface - and back-end - how the objects of design are represented internally in the system. This section will focus on the software, and illustrate through the implementation of UDesign, what are some of the issues to consider in terms of software for user-centered architecture.

A Framework For User-Centered Architecture

As a framework, UDesign mediates between the user and stakeholders in the building industry - architects, developers, contractors, suppliers, realtors, and community - in order to create a product, in this case, an apartment. Figure 4.1 outlines the components and relationships of user-centered architecture as a series of four concentric circles - *User, UDesign, Data, and Stakeholders* - floating in *Design Space*, the space of design solutions offered to the user. Metaphorically, as the user is designing his own product, he is navigating through the design space, cocooned within these layers of UDesign and Stakeholders, through which he communicates his ideas about the product. To visualize the metaphor, imagine Figure

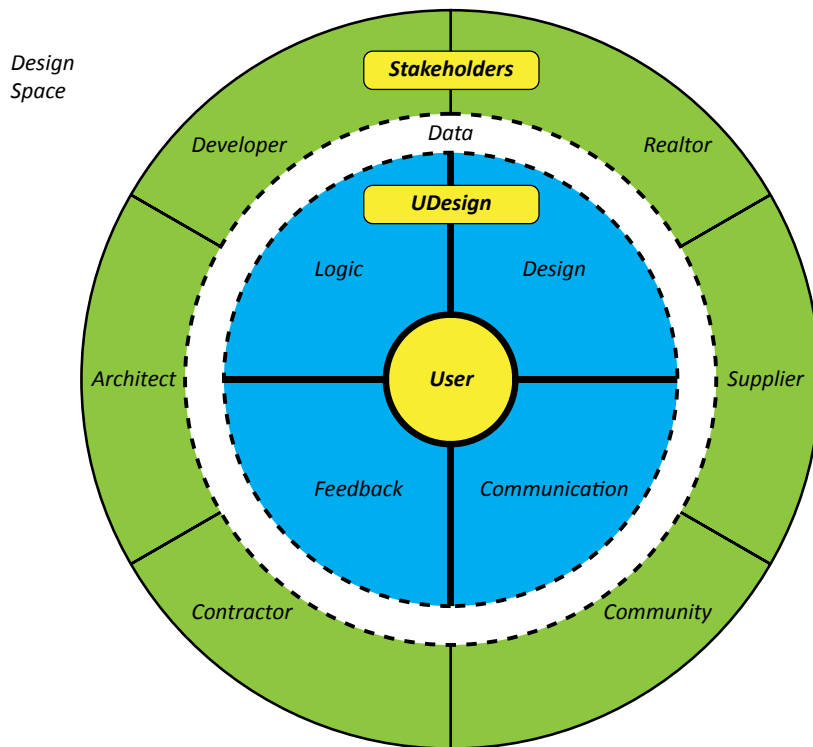


Figure 4.1: User-Centered Architecture.

4.1 as a ball, floating in space, and rays of information shooting from its center (User), guided by an inner mechanism (UDesign, Stakeholders) trying to find an anchor in space (Design).

UDesign is comprised of four parts: *Design*, *Communication*, *Feedback*, and *Logic*. Design is responsible for visualizing the design space and interacting with the user. In a way, it is the front-end of the interface. Design includes the library of parts from which a user can choose to create his design (Friedman's repertoire), tools to transform and edit the design, and ways to explore the design space. Communication is responsible for the exchange of information between user and stakeholders. The radial arrangement facilitates multiple channels of communication at any given point in time. For example, a user can communicate with the architect in order to get professional feedback about his design while also getting feedback from the community. Communication must pass through UDesign to avoid overloading the user, therefore, messages are stored in a queue and ordered by their relevancy to the current context of the design process. Feedback is responsible for continuously collecting information about the state of the design process, analyzing it, and providing notifications at the right time and place. For example, when a user selects a design component, he is immediately given feedback about the availability of the component,

its price, compatibility with other components, and more. Stakeholders can use UDesign's feedback to get information about their users - purchase history, preferences, lifestyle, status, state of the design, and more. Logic is responsible for holding everything (*Design, Communication and Feedback*) together so that users' experience of UDesign is both productive and positive. Logic is supported by a suite of algorithms from different categories - user interaction, network, database, machine learning - some of which are described in more detail in chapter 3 (Algorithms), as well as in the next section (Implementation).

Implementation

UDesign was implemented in Adobe Flex Builder 3, an IDE (Integrated Development Environment) for creating cross-platform RIA (Rich Internet Application) based on Adobe Flash. A typical application development process was followed including:

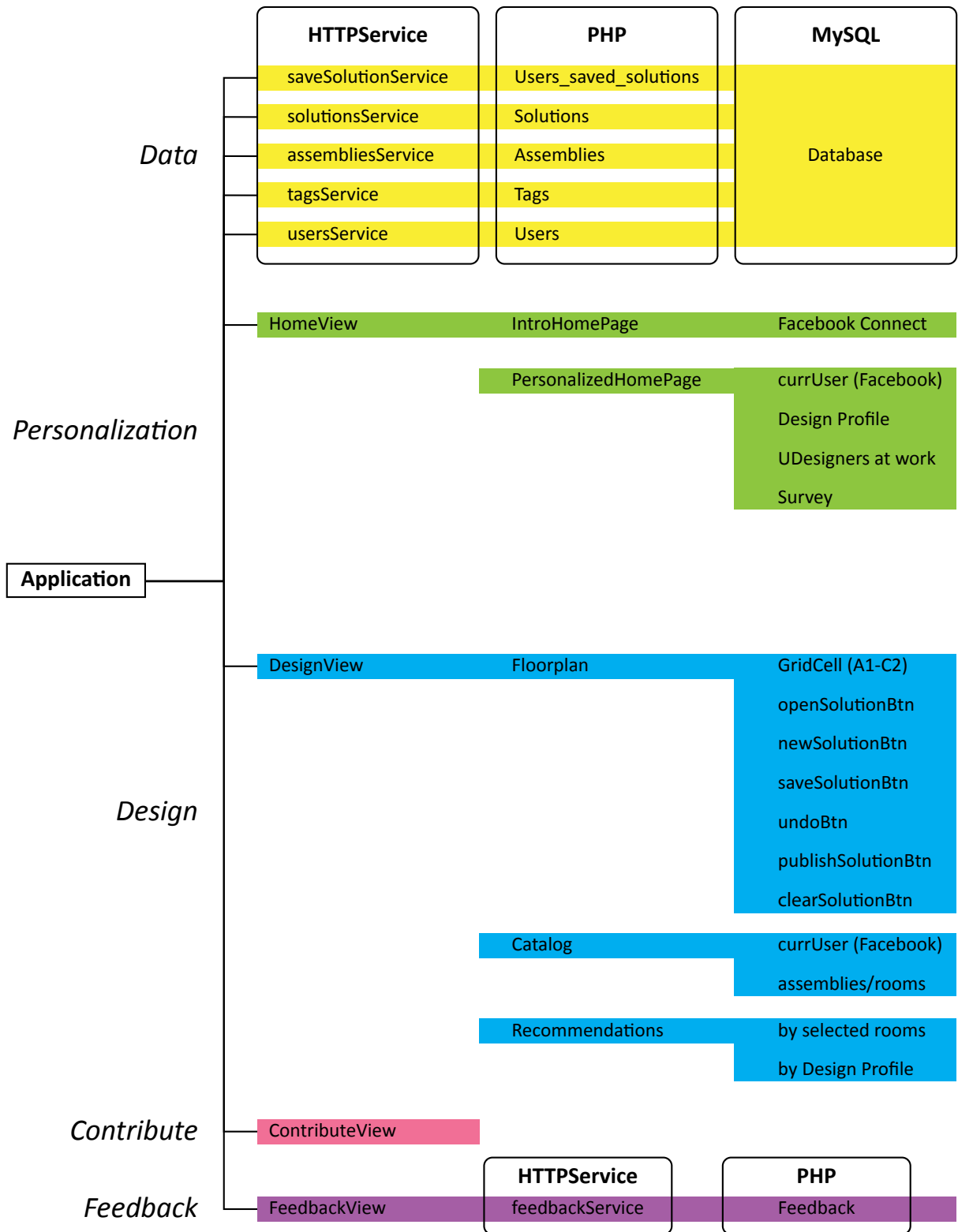
- Defining an application interface using a set of pre-defined components (forms, buttons, and so on) as MXML Components.
- Arranging components to create a User Interface (UI) design.
- Using styles and themes to define the visual design: CSS, skins, etc.
- Adding dynamic behavior using proper event-driven programming techniques.
- Defining and connecting to data services as needed: MySQL, ColdFusion, etc.
- Compiling the source code into an SWF file that can be read by Flash Player and embedded within an HTML file.

UDesign is comprised of five parts: Data, Personalization, Design, Contribute, and Feedback (Figure 4.2), where Data constitutes most of the back-end and the remaining parts the front-end.

Data

The data in UDesign is stored in tables within a MySQL database that resides on a server. The mxml application makes requests to the database through a Flex HTTPService component. In turn, the HTTPService component calls one of five methods (*FindAll, Insert, Update, Delete, Count*) encapsulated within a PHP file, which then retrieves the data from the appropriate table of the database and returns the data to the application in an E4X format (Figure 4.3).

Figure 4.2: UDesign - system overview



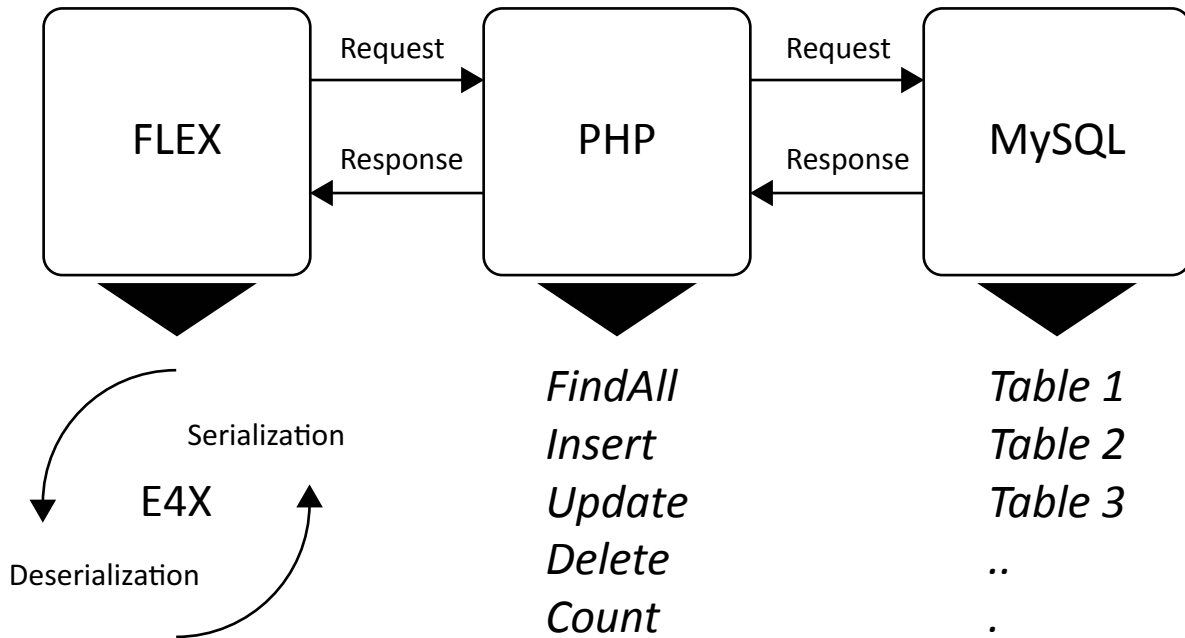


Figure 4.3: Data pipeline.

The database includes the following tables:

- *assemblies* (Figure 4.4).
- *solutions* (Figure 4.5).
- *users_saved_solutions* (Figure 4.6).
- *users* (Figure 4.7).
- *design_profile* (Figure 4.8).
- *solution_vector* (Figure 4.9).
- *feedback* (Figure 4.10).

Table: assemblies

This table holds the 48 assemblies (custom-designed rooms) available to the user on the *Design* page of UDesign.

Fields: id: room id | label: room label | type: room type | description: room description | price: room price | cell: an index indicating to what functional zone the room belongs (more details below) | area: room area (sqft) | dimx: room dimension in x-axis (pixels) | dimy: room dimension in y-axis (pixels) | locx: room position in x-axis (pixels) | locy: room position in y-axis (pixels) | color: room color code | fit: a flag indicating whether the room fits/doesn't fit with the user's design (more details below).

Table: solutions

This table holds the 285 solutions (custom-designed floor plans) available to the user on the *Design* page of UDesign.

Fields: id: floor plan id | label: floor plan label | A1 - C2: a functional zone in the floor plan (more details below).

Table: users_saved_solutions

This table holds users' saved solutions (custom-designed floor plans) created by users on the *Design* page of UDesign.

Fields: id: floor plan id | label: floor plan label | A1 - C2: a functional zone in the floor plan (more details below).

Table: users

This table temporarily holds users' personal information obtained from their Facebook account (more details about UDesign's connection to Facebook in the section about Personalization).

Fields (Facebook generated): uid: user's id | about_me: user's personal message | activities: user's favorite activities | birthday: user's birthday | books: user's favorite books | first_name: user's first name | interests: user's interests | last_name: user's last name | movies: user's favorite movies | music: user's favorite music | pic: user's profile picture | relationship_status: user's current relationship status | sex: user's gender | wall_count: number of posts on user's wall.

Table: design_profile

This table holds users' design profile. Users are presented with ten parameters of lifestyle and asked to rate their importance on a scale of one (low) to five (high).

Fields: id: user's Facebook id | watch: importance of watching TV/Movies | entertain: importance of entertaining | cook: importance of cooking | work: importance of being able to work from home | privacy: importance of privacy | pamper: importance of pampering | storage: importance of storage | kitchen_dining: importance of physical/visual connection between kitchen and dining room | bed_bath: importance of physical/visual connection between bedroom and bathroom | flexible: importance of flexible living space.

Table: solution_vector

This table holds the design vectors for 71 out of 285 solutions (custom-designed floor plans) where each solution is represented as a different combination of the same parameters used to define user's design profile (more details in the section about the Design page). Values indicate how much the corresponding parameter is present in the solution.

Fields: label: solution label | watch: supports watching TV/Movies | entertain: supports entertaining | cook: supports cooking | work: supports being able to work from home | privacy: supports privacy | pamper: supports pampering | storage: amount of storage | kitchen_dining: degree of physical/visual connection between bedroom and bathroom | bed_bath: degree of physical/visual connection between bedroom and bathroom | flexible: amount of flexible living space.

Table: feedback

This table holds users' feedback submitted through the feedback form on UDesign (more details below).

Fields: subject: subject of the feedback | content: content of the feedback.

id	label	type	description	price	cell	area	dimx	dimy	locx	locy	color	fit
A11BT001	Bathroom	Bathroom		99	A1	68	174	166	0	0	0x0071BC	0
A11BT002	Bathroom	Bathroom		99	A1	68	174	166	0	0	0x0071BC	0
A12BT001	Bathroom	Bathroom		99	A1	100	250	166	0	0	0x0071BC	0

id	label	A1	A2	B1	B2	C1	C2
SOL1	322,000	A12BT001	A21KT001	B15ST001	B21DN001	C15BD001	C23LV001
SOL10	123,000	A11BT001	A22KT005	B15ST001	B21DN001	C15BD001	C23LV001
SOL100	453,243	A11BT001	A22KT004	B12BD001	B25DN001	C12BD001	C25LV004

id	label	A1	A2	B1	B2	C1	C2
652795248		A11BT001	A22KT003	B13BD001	B25DN001	C11ST001	C25LV002
717158422	brain drain	A11BT001	A22KT001	B15ST004	B25DN002	C13BD001	C23LV001
533833493	carlyu	A12BT001	A21KT001	B15ST001	B21DN001	C15BD001	C23LV001

uid	about_me	activities	birthday	books	first_name	interests
last_name	movies	music	pic	relationship_status	sex	wall_count

id	watch	entertain	cook	work	privacy	pamper	storage	kitchen_dining	bed_bath	flexible
1065658831	3	5	5	1	1	1	3	5	1	1
542150884	5	2	4	1	4	1	3	1	1	1
566433466	1	5	5	2	2	1	2	4	2	1

label	watch	entertain	cook	work	privacy	pamper	storage	kitchen_dining	bed_bath	flexible
SOL1	3	2	2	5	4	5	3	5	1	1
SOL2	3	2	2	4	4	5	4	5	5	1
SOL3	3	2	2	5	4	5	4	1	1	1

subject	
this is my feedback	feedback feedback
scrolling	hi junno! Nice work! Good start. Maybe I am to quick to try out everything- the scroll bar or the arrow... It is just a little bit annoying. :-)-And will I be ab proportions to the length of the walls... I tried the little pen and ruler tool I

Figure 4.4 - 4.10: Database tables (from top to bottom).



Figure 4.11: Introduction home page

Personalization

UDesign's home page has two states: an introduction page (Figure 4.11) and a personalized page (Figure 4.12). The introduction page is the first page to be displayed when a user first loads the application. It contains information about the application and how to use it. To start using the application, a user must log in using their Facebook account by pressing the Facebook Connect button. Facebook Connect is integrated into the Adobe Flex application using the Facebook API and corresponding Flash-Flex libraries. By signing in to UDesign through Facebook Connect, a user gives the application permission to make requests to Facebook and retrieve their personal information (according to the user's privacy settings).

The information from Facebook, now encapsulated as a FacebookUser object, is leveraged to provide the user with tailored recommendations of design solutions, i.e., custom-designed floor plans. As shown in chapter 3 (Algorithms), users' information, whether it's ratings or a Facebook profile, can be represented as a vector (Figure 4.13) and compared to other users in order to cluster the UDesign community. Once a user is clustered, other members of the cluster become his design critics and their design solutions are offered as recommendations (Figure 4.14). That is the case with a new user or a user who has not created, rated, or saved any design solutions. When a user has given feedback, implicitly (by saving certain design solutions) or explicitly (by rating design solutions), his recommendations are updated and improved through a user-based collaborative filtering algorithm. As shown in chapter 3 (Algorithms), the algorithm

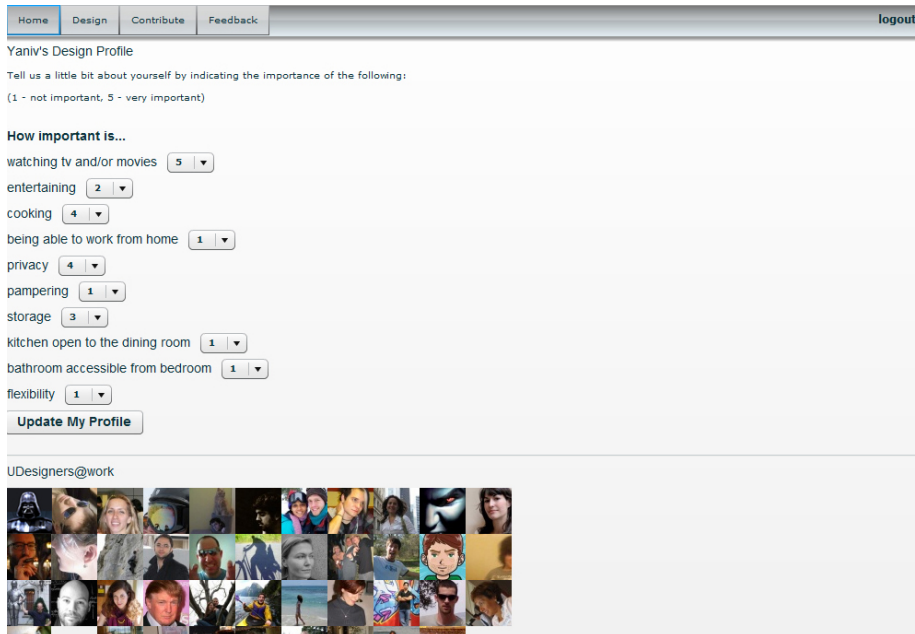


Figure 4.12: Personalized home page.

compares the current user to the entire UDesign community, finds the most similar users, and collects the highest rated design solutions from these users as recommendations. Consequently, the more design solutions users rate, the more accurate UDesign's recommendations get (see chapter 3).

Once the user has logged in and UDesign has made a request to Facebook to retrieve the user's personal information, the application waits for the response and then temporarily stores it as a *FacebookUser* object for the duration of the session. If log in is successful, the user is automatically navigated to a personalized home page which includes the following: user's Facebook picture and name, user's Rated Design Profile, UDesigners at work, and Survey.

The user's Rated Design Profile (Figure 4.15) is a set of ten lifestyle parameters that capture the user's preferences in respect to the database of 285 floor plans available in UDesign. For each option, the user has to rate the importance of that aspect on a scale of one (low) to five (high). For example, if the user is an avid baker, he might rate high the following parameters: cooking, connection between kitchen and dining, and entertaining, to reflect their importance in his lifestyle. The user has the option to go back and change his ratings at any time by making the changes and pressing the *Update* button. The Rated Design Profile complements the Facebook information as they are both used to model the user and provide him

with recommendations of design solutions.

UDesigners at work features the Facebook profile pictures of the UDesign community with which the current user may wish to interact - comment on designs, share designs, ask questions, share information, and more. Finally, a short survey is included with various design styles and lifestyle parameters. The user is asked to indicate whether or not they are relevant to him. The results of the survey are collected and aggregated across the entire UDesign community to understand trends and user typologies.


facebook	Age	Sex	School	Job	In a Relationship			
	32	Male	MIT	House_n	Yes			
facebook	29	Male	MIT	House_n	Yes			

Figure 4.13: User vectors.

Design

The Design page (Figure 4.16) is where users create their custom apartment and it includes three main components: floor Plan, catalog and recommendations.

The floor plan (Figure 4.17) has two components - toolbar and sample floor plan - with which a user can interact to create their design. The toolbar offers the following functionality:

- Open a saved project - will load a saved design from the database.
- Create a new project - will create a new project in the database.
- Save a project - will save the current state of the design in the database.
- Undo - will undo the user's last operation.
- Publish to Facebook - will post the current state of the design to the user's Facebook feed (Figures 4.17A and 4.17B).
- Clear the project - will remove everything from the sample floor plan.

The sample floor plan is the empty canvas on to which a user drags and drops rooms from the catalog (Figure 4.18). UDesign was implemented with a single sample floor plan as a proof-of-concept. However, in reality, UDesign should be able to offer users a variety of floor plans to customize. as well as an option to create their own floor plan from scratch. Figure 4.19 explains how the sample floor plan is structured such that it supports customizing. Essentially, the floor plan is divided into six functional zones - A1, A2, B1, B2, C1, C2 - where each zone can hold one or more functions. Zones A1 and A2 have hard-coded constraints, i.e., A1 can only hold a bathroom and A2 a kitchen. These constraints are imposed by the

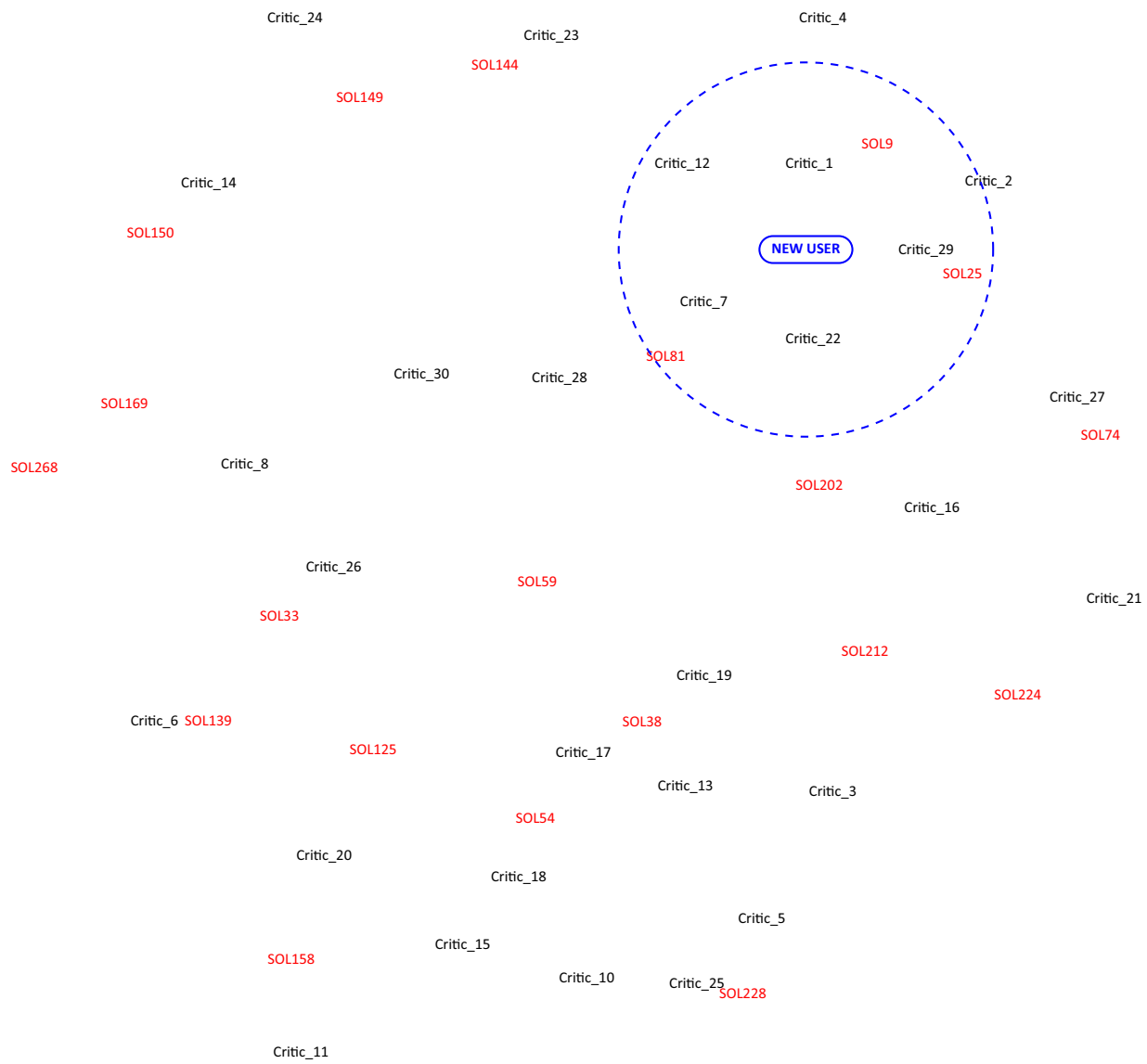


Figure 4.14: Clustering users to find recommendations.
50

Yaniv's Design Profile

Tell us a little bit about yourself by indicating the importance of the following:
(1 - not important, 5 - very important)

How important is...

watching tv and/or movies

entertaining

cooking

being able to work from home

privacy

pampering

storage

kitchen open to the dining room

bathroom accessible from bedroom

flexibility

Update My Profile

Figure 4.15: Rated Design Profile.

building's infrastructure, for example, the location of plumbing risers. The remaining zones can each take a number of prescribed functions - dining room, bedroom, closet, media room, and more - according to the user's choice.

The Catalog (4.20) holds the assemblies (custom-designed rooms) that a user can add to the sample floor plan. The catalog includes an information box with the user's Facebook name and the title of the project. It also has a drop-down list of room types that filters the available rooms according to the selected type, e.g., bathroom, kitchen, dining, etc. Rooms are represented as tiles. When a room is clicked a pop-up window appears with additional details. Room tiles can be dragged and dropped onto the floor plan (Figure 4.18). Each room has a prescribed location within the floor plan to reflect real-world constraints. Once a room has been dragged and dropped on to the floor plan, an algorithm (Figure 4.21) updates the rooms in the catalog to reflect which ones fit and which ones don't with the user's current selection of rooms. The ill-fitting rooms are grayed-out (disabled). The recommendation lists also update (see below).

The recommendations (Figure 4.22) are represented as two lists of solutions (custom-designed floor

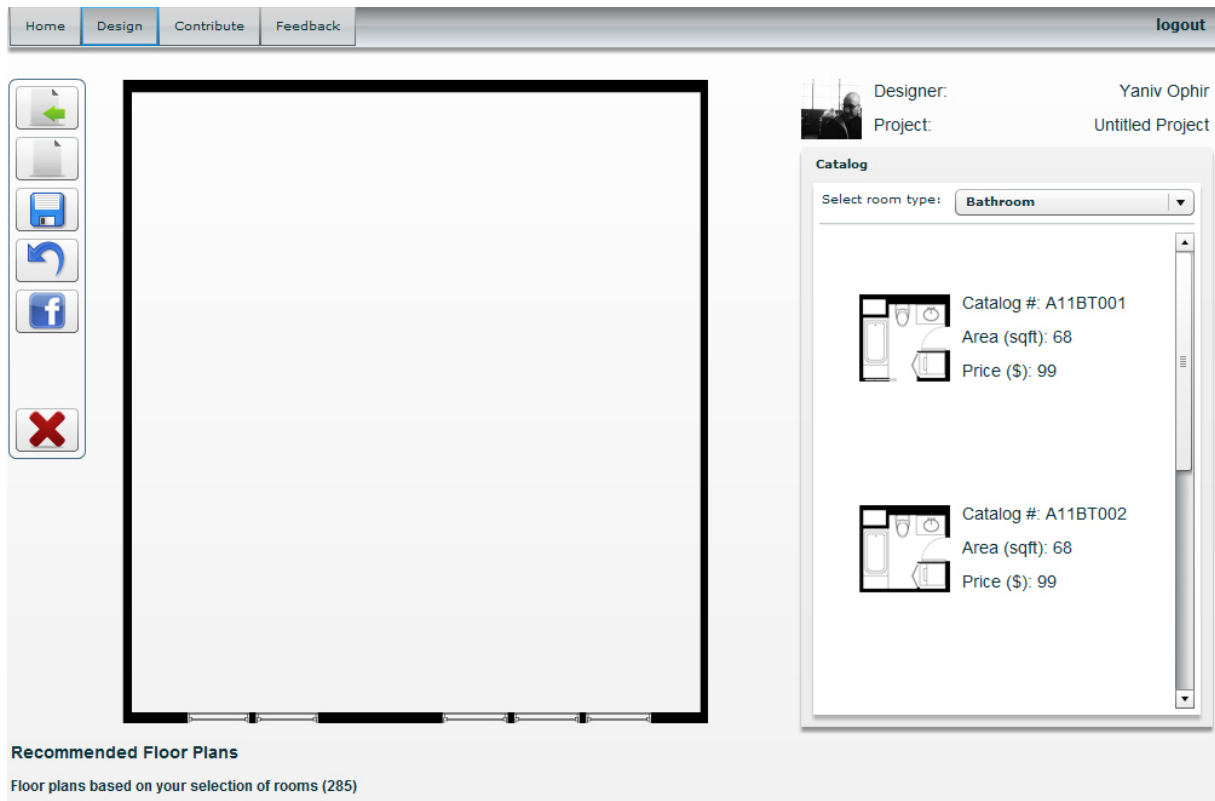


Figure 4.16: Design page.

plans) that are updated to reflect: (a) solutions which are compatible with the user's currently selected rooms and, (b) the ten highest ranking solutions which best fit the user's Design Profile. Recommendations *Based on your selection of rooms* (Figure 4.22) are generated and updated using the algorithm described in Figure 4.21. Recommendations *Based on your design profile* (Figure 4.22) are generated and updated using the user-based recommendation algorithm described in chapter 3 (Figure 3.3). When a user clicks on any of the recommendations, a pop-up window appears with additional details about the selected floor plan, an explanation of why the floor plan was recommended, and an option to add the selected floor plan to the sample floor plan (Figure 4.23). If the user clicks the *Add To My Project* button, the user's sample floor plan is cleared and the rooms of the currently selected floor plan are added instead.

From a usability perspective, the Design page uses the direct manipulation design pattern to facilitate a seamless, intuitive, and efficient design process for users. Following a gradual learning curve, a user can create his own custom floor plan within a matter of minutes by using the catalog's drop-down menu, locating a room, and adding it to the sample floor plan.

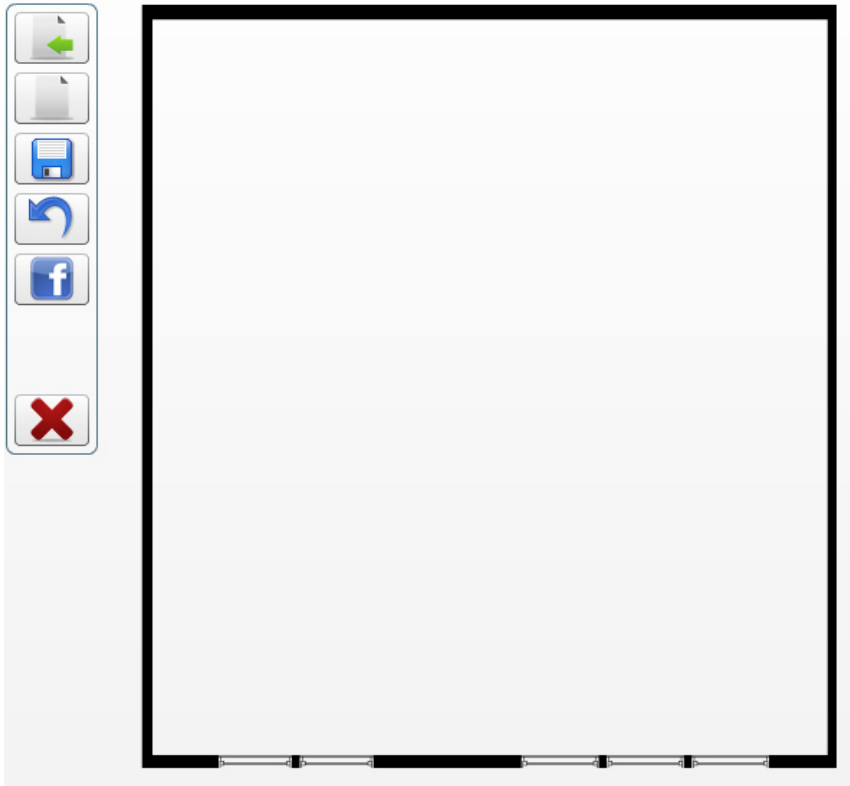


Figure 4.17: Floor Plan Component - sample floor plan and toolbar.

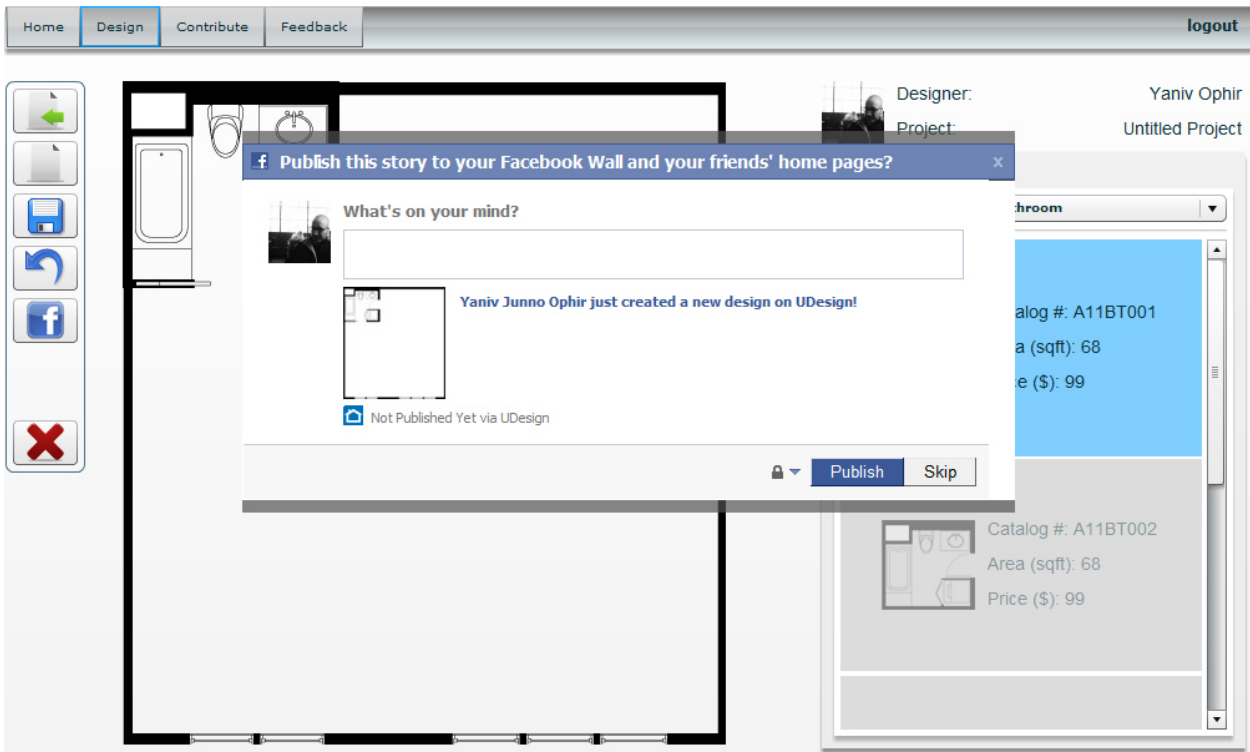


Figure 4.17A: Posting a floor plan to Facebook from within UDesign.

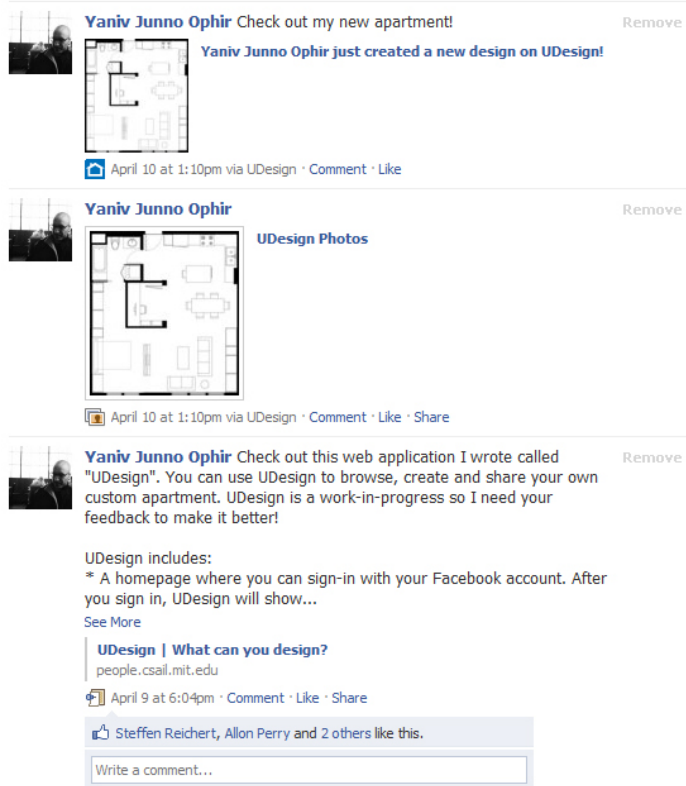


Figure 4.17B: The posted floor plan in Facebook.

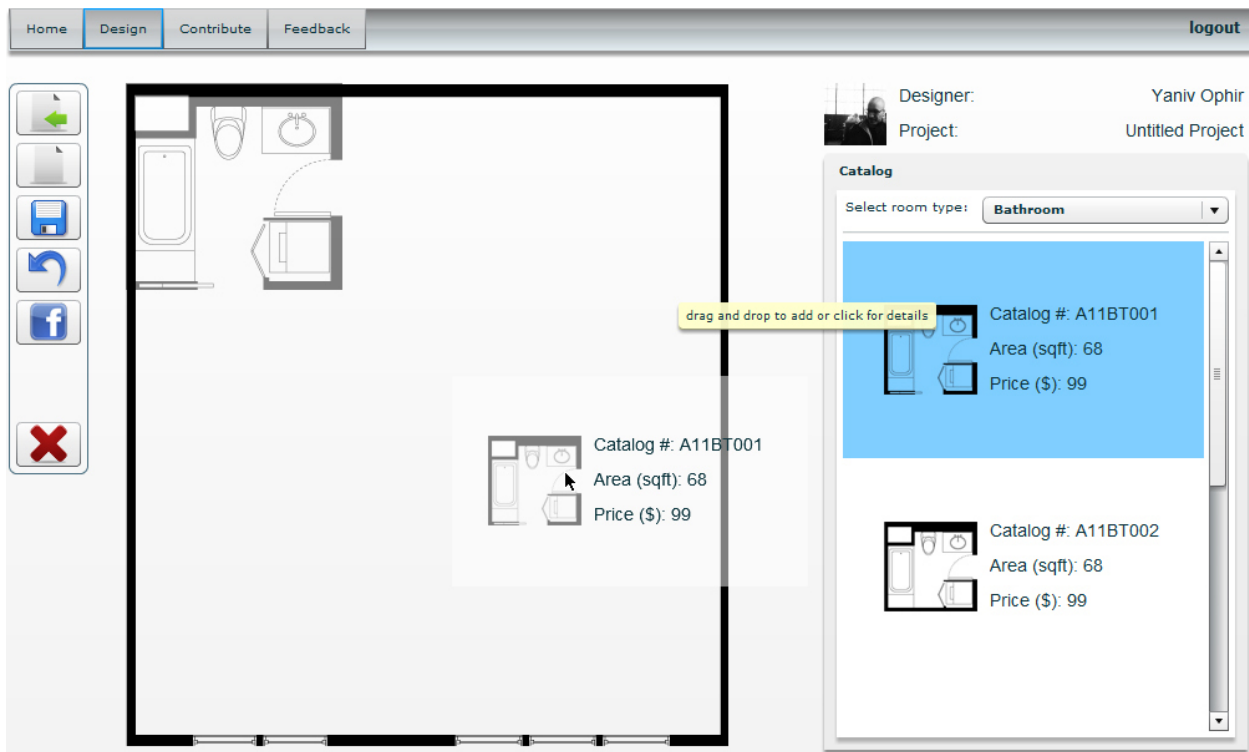


Figure 4.18: Drag and drop rooms.



UDesign

How does it work?

The sample apartment is divided into six functional zones (A1, A2, B1, B2, C1, C2). The bathroom is always in A1 and the kitchen in A2. The rest of the zones can take on any number of functions.

Each zone can have one of a set of prescribed dimensions depending on the function it performs. For example, A1 can be 174 pixels wide by 166 pixels long OR 250 pixels wide by 166 pixels long.

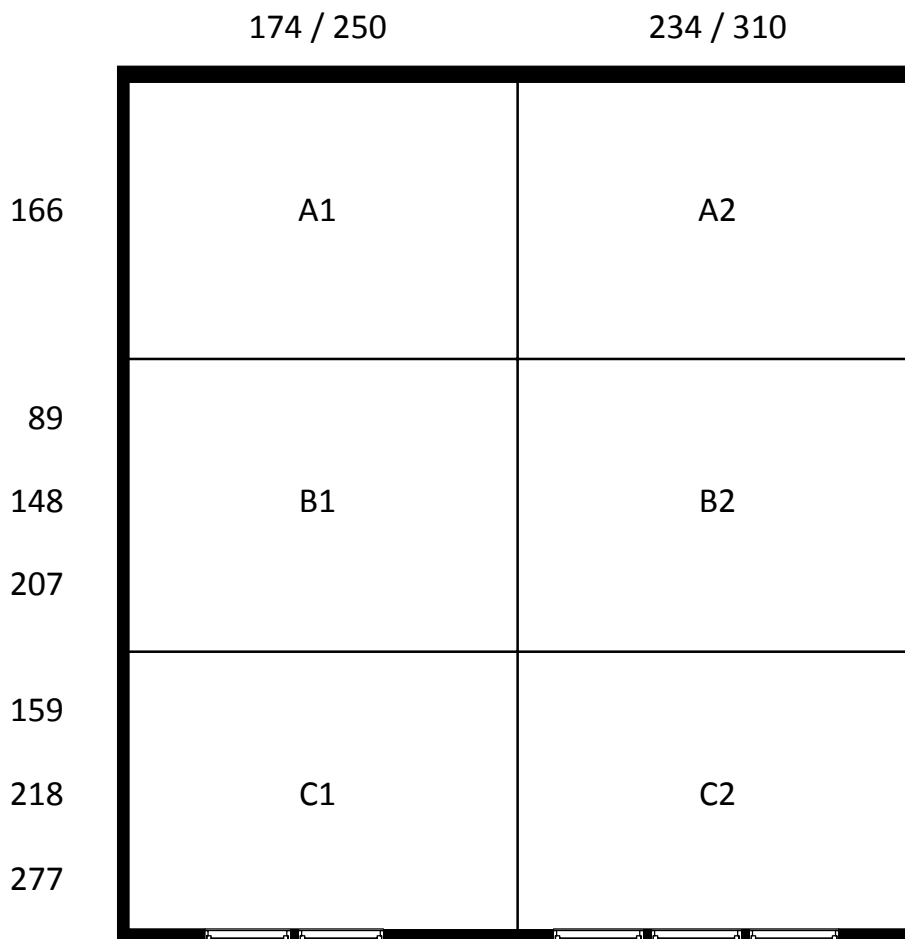


Figure 4.19: How does it work.



UDesign

Sample Room/Assembly

Type: Bathroom

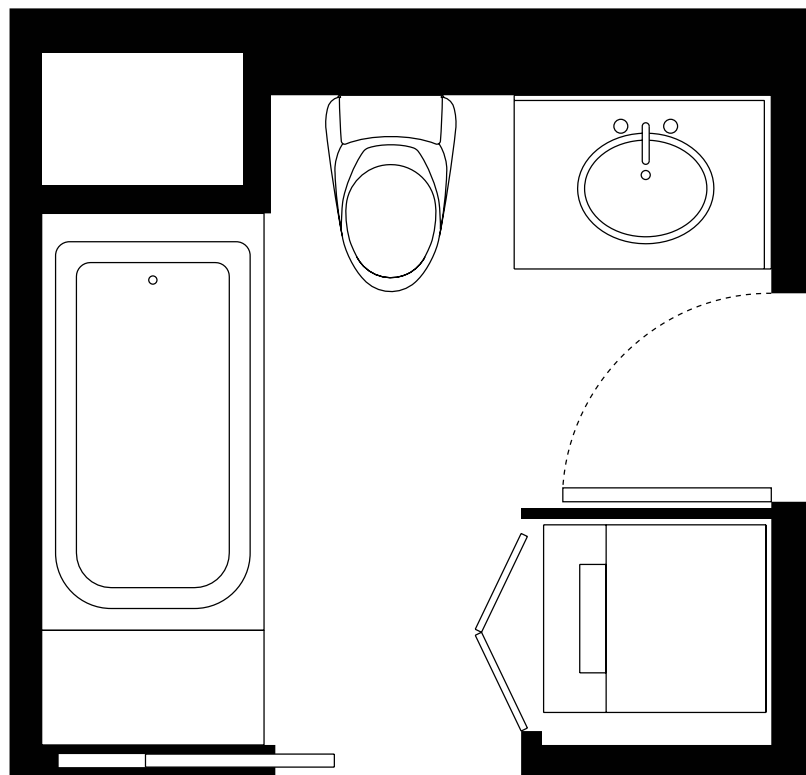
Catalog#: A11BT001

Functional zone: A1

How does it work?

174 pixels

166 pixels



Format: PNG
Resolution: 72 ppi
Dimensions: 174 x 166 pixels
Other: Transparent background

Figure 4.19: How does it work (continued).

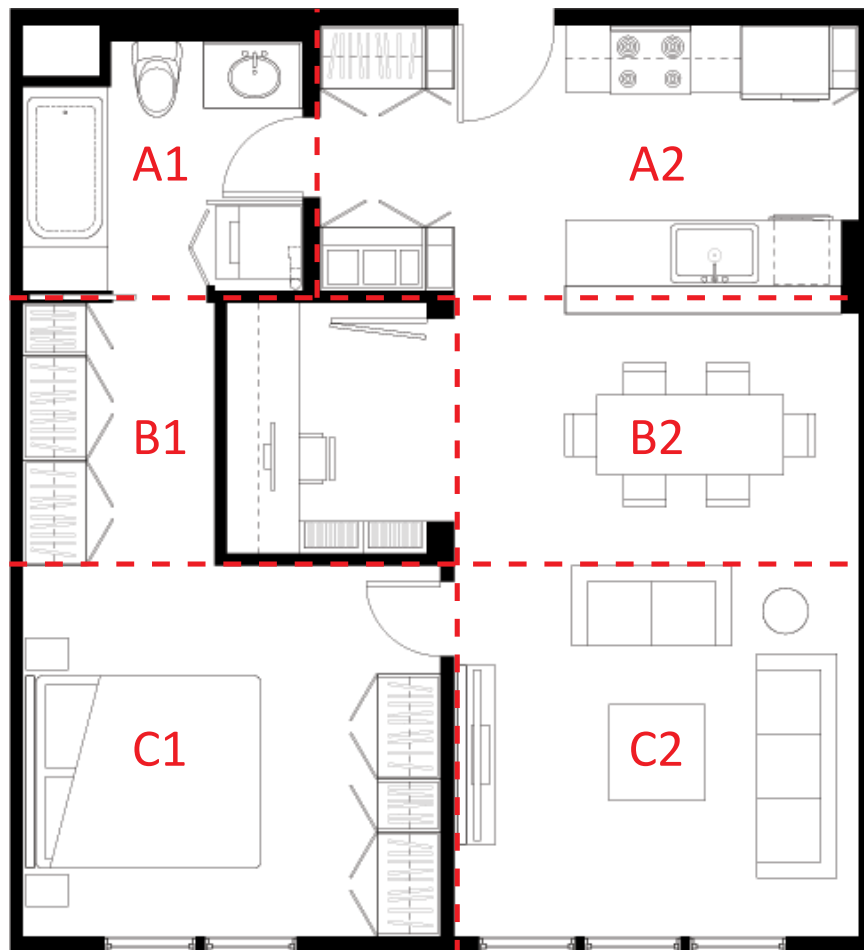


UDesign

Sample Floor plan/Solution

Catalog#: SOL25

How does it work?



Format: PNG
Resolution: 72 ppi
Dimensions: 484 x 531 pixels
Other: Transparent background

Figure 4.19: How does it work (continued).

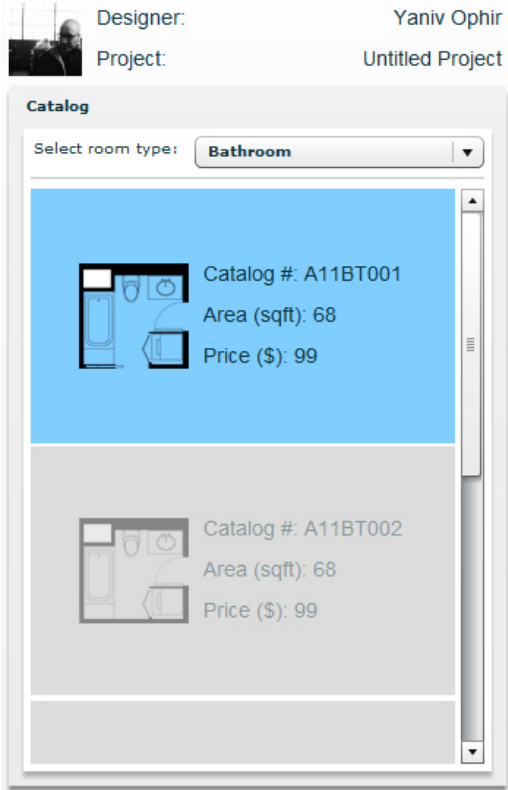


Figure 4.20: Catalog.

```
For each Solution,  
    For each Assembly in Solution,  
        If Assembly is in the user's sample floor plan,  
            Save Solution,  
            Save Assembly,  
For each Solution in Recommendations,  
    Update Solution  
For each Assembly in Catalog,  
    Update Assembly
```

Figure 4.21: Pseudocode of algorithm for updating solutions and assemblies.

Recommended Floor Plans

Floor plans based on your selection of rooms (175)

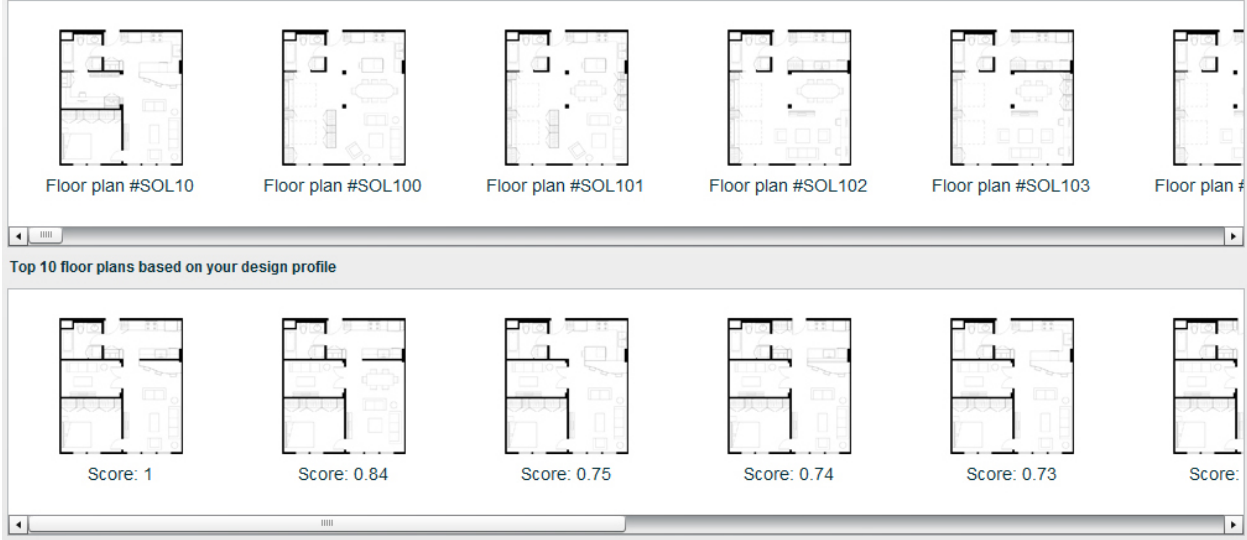


Figure 4.22: Recommendations.

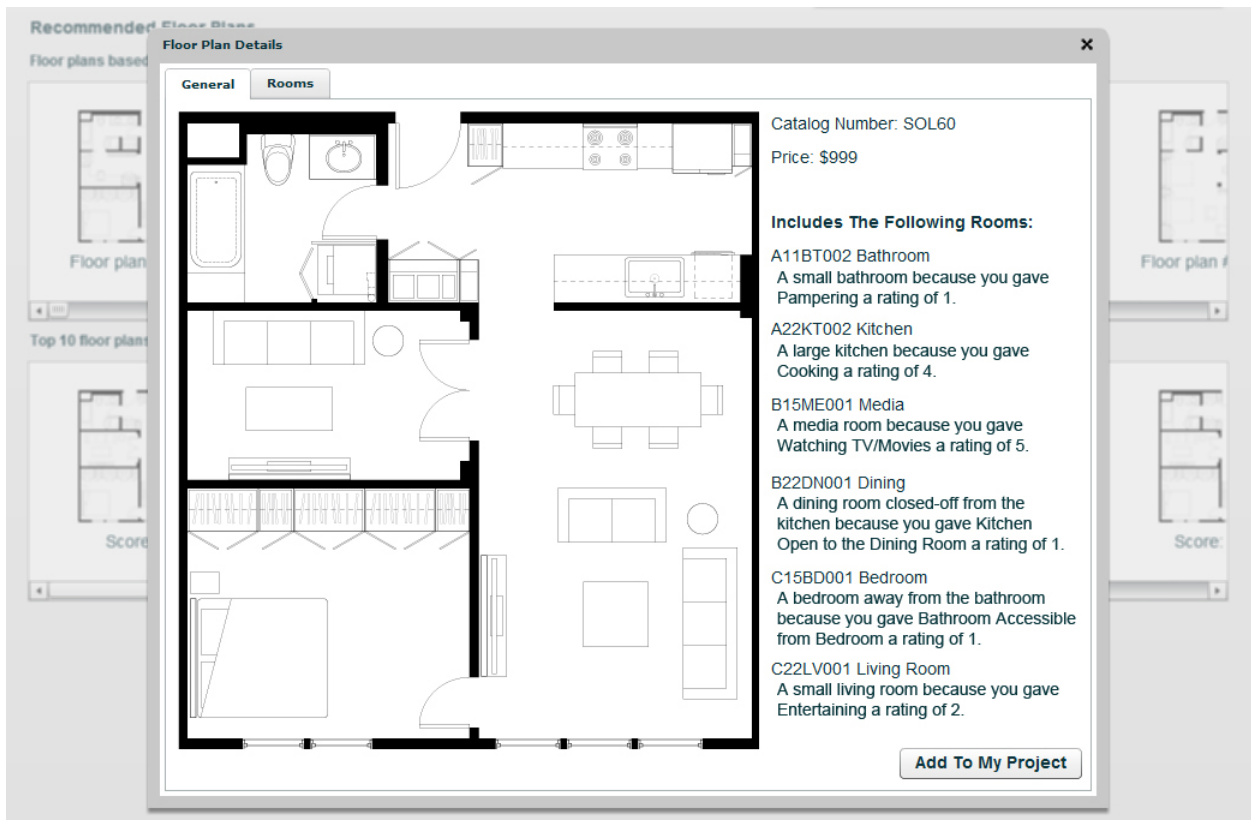


Figure 4.23: Solution details.

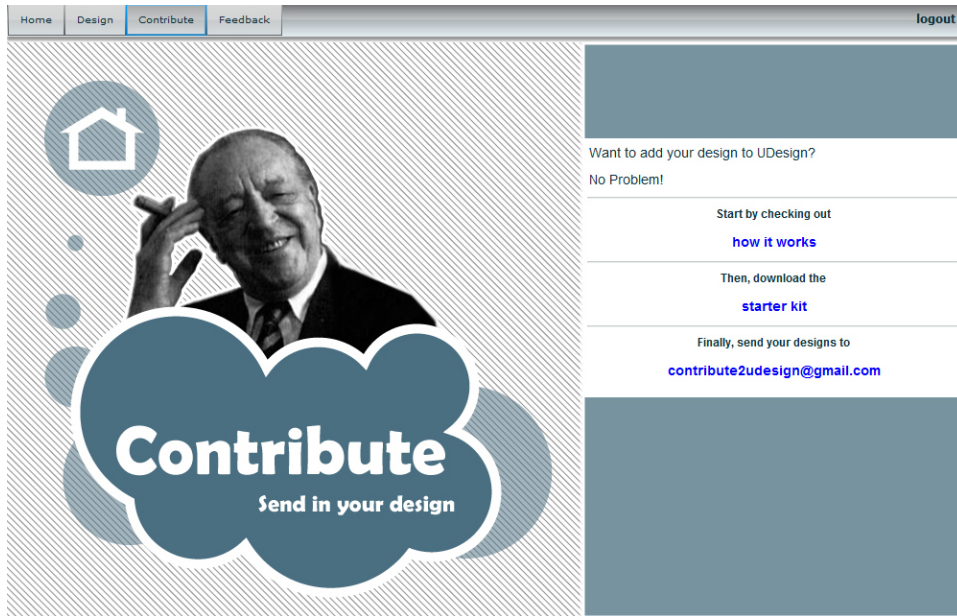


Figure 4.24: Contribute page.

Contribute

UDesign is intended to be extended by its user community. A user who is interested in contributing a new design can download the *Starter Kit* (a template file of available rooms in Adobe Illustrator and/or Autodesk Autocad format), design their own room, or floor plan, and send it via e-mail to be added to the UDesign database (Figure 4.24). The “starter kit” is a file containing drawings. However, the current approach is far from ideal and leaves much to be desired. As an improvement, the web application can include an integrated configuration tool which will let users edit any available assembly and save it as their own. A more comprehensive discussion of how to extend and scale UDesign is reserved for the conclusions.

Feedback

To improve UDesign, a simple feedback form is implemented allowing users to write comments and suggestions, and submit them to the system (Figure 4.25). Once a form is submitted, it is saved in the feedback table (Figure 4.10), and awaits a response by the application administrator.



Figure 4.25: Feedback page.

References & Suggested Reading List

1. UDesign: <http://people.csail.mit.edu/junno/hgp/> (May 20, 2010)

5 | Conclusions

This thesis proposes a framework for user-centered architecture and provides a proof-of-concept in the form of a web application that allows users to create their own custom apartment. The ideas presented in this thesis are merely a starting point and fall short of solving the problem. To mitigate the situation, this section suggests ways of improving some of the ideas presented, as well as additional directions for future work.

Yona Friedman: Reloaded

This thesis started out by framing its conceptual framework through the work of Yona Friedman on user-centered design. Friedman's work is compelling, however, it is still very much focused on the architect as the sole cause of problems in the design process. As I have shown in chapters three and four, a community-based approach supported by collaborative filtering algorithms and social networks (e.g., Facebook) can provide users with the safety net they need in order to feel comfortable designing their own spaces and living solutions. There is a specific reference by Friedman, when he talks about the repertoire, to all possible design solutions, which I find problematic. It seems to me that any system that forces its users to navigate through a huge corpus of data, compromises its usability. For example, if the repertoire had thousands of possible design solutions, would a user be able to find what he needs? And if he could, how long would it take him? These questions raise doubts in my mind as to the ability of Friedman's system to respond to users' needs in terms of efficiency (time), and visibility (finding what they need). Furthermore, any system that proposes to find ALL possible solutions must also address the issue of *Completeness*, or in

other words, provide proof that there are no other possible design solutions beyond those found in the repertoire. Finally, I would argue that users aren't interested in all possible design solutions, but just those that fit their needs. Finding these needs is a difficult problem with the beginning of a solution in chapters three and four.

Representation

Marshall McLuhan (McLuhan, 1964) coined the phrase "The medium is the message." The medium, in the context of architectural design, is how we choose to represent the finished product. Whether through 2D floor plans, or 4D fly-through animations, the message has to be received and understood on the user's side. The fact that users are not trained at deciphering architecturally-encoded messages (architectural notation, symbols, etc.) makes the task of representation that much more difficult. The representation chosen for the implementation of UDesign, i.e., a 2D floor plan layout comprised of functional zones or assemblies in PNG format, leaves much to be desired. First, the use of the PNG format seriously hinders users' freedom to make changes to the design at room level. While only having to deal with small chunks of space simplifies the design task on the user's end, it also presents a problem when a user only wants to move the bed in the master bedroom one foot to the left. This limitation will cause users to become frustrated and the entire user-centered design process to break. Therefore, representation should provide users more freedom by increasing the granularity of the medium. For example, enabling users to edit assemblies and then save them as their own custom components. Having said that, perhaps the problem is not granularity but dimensionality. In many informal discussions I conducted with users (non-experts designers, if you will), I've come to realize that floor plans are a poor conduit for the architectural message. Most users just don't get them. They have a hard time "sensing" the space through a 2D representation and yearn for a more engaging spatial experience. At this point, I am tempted to suggest Virtual Reality as a more appropriate medium for architectural data and design, however, it doesn't seem like a viable option in the foreseeable future. If we were to consider (hypothetically speaking, of course) Virtual Reality, I would imagine users being able to connect through some device to a virtual world where they could create their own living spaces, walk through them, share them with others, and so on. A hint of the potential of this scenario surfaced when architects started taking advantage of SecondLife by designing homes for users' avatars. Until Virtual Reality catches up, video/online games and virtual worlds, provide a better alternative, in my opinion, to 2D floor plan layouts, or even 4D walk-through animations. In an interactive virtual environment, users can create their design in 3D space, walk around, interact with objects, explore

views and perspective, really “see” and “feel” the consequences of their actions, and be able to share that experience with others. Another thing to consider about representation when venturing into 3D space is the level of abstraction appropriate for users’ comprehension. Video/Online games are fun because they tend to be an abstract representation of the real environment for the purpose of reducing the cognitive load on the user. On the other hand, architectural design has many varying levels of abstraction which are linked to the context and position within the design process. So the question becomes would users want to see everything - from the texture of a wooden floor to the finish of the kitchen cabinets? Or would the level of detail dynamically change to match the user’s focus and location within the design process? I suspect a dynamic and gradient control over spatial details is probably more appropriate, and less likely to overwhelm the non-expert user.

Scalability

At the heart of user-centered architecture is scalability - how can we create a platform that supports the design processes of multiple users? Scalability is tightly coupled with representation in the sense that representation ultimately determines the degree to which a user-centered design process can scale. For example, the representation chosen for implementation of UDesign has a low degree of scalability because users can’t edit assemblies or create their own assemblies, which is necessary if the catalog (or repertoire) is to grow and expand. A more flexible representation, one where users can manipulate low-level components, such as pieces of furniture, would result in better scalability allowing users to extend the initial catalog of assemblies. Another issue to consider in regards to scalability, is that of constraints, or rather the need to enforce and validate constraints. In architectural design, components, across various scales, relate to each other in various ways through interdependencies or constraints. As the number and types of components increase, so do the constraints and the complexity of validating them. For example, UDesign implicitly enforces constraints on room adjacencies through the soft suggestions it provides to the user. That means that if the user chooses to ignore the recommendations, he can easily place mismatching rooms next to each other. What is needed in UDesign specifically, and user-centered design systems in general, is an adaptive constraint management approach with some notion of learning. Ideally, such a management approach would start with a small set of predefined constraints and learn new constraints from users’ actions and design patterns. The goal is to avoid having to hard-code the constraints, but rather infer them from what users do as they are designing. Winston (Winston, 1972) developed the notion of learning (constraints) from examples and demonstrated the application of his approach in the construction

of an arch. Using examples of what constitutes an arch (good examples), what doesn't (bad examples), and what's close enough (near miss), Winston's approach incrementally constructs a semantic net that is a representation of the arch and its embedded constraints. A similar approach may be applicable in UDesign where a user's actions are being tracked, parsed, and interpreted to construct a semantic representation of his design, and its embedded constraints.

Collaboration

Architectural design is a collaborative effort that includes the input of many different experts, such as architects, contractors, users, and many others. Therefore, there's no reason why non-experts couldn't take advantage of the same approach and collaborate amongst themselves to help each other create better designs. Collaboration is addressed in UDesign by giving users the option to publish their work on Facebook. This form of social collaboration can potentially help a user who is trying to get some feedback. However, what is needed is an additional form of collaboration where a user can remotely interact with another user's design. Hands-on, remote collaboration is a more expressive approach to design that facilitates users designing together.

Data Collection

The majority of approaches discussed in this thesis (e.g., the algorithms in chapter three) are data-driven, meaning that without sufficient data, they become inaccurate, or just don't work. In fact, the more data, the less sensitive these algorithms are to outliers, and the more accurate the recommendations. Unfortunately, in the course of writing this thesis and developing UDesign, I was unable to collect as much data as I had hoped. As a result, I can only speculate as to the performance and potential contribution of my approach to user-centered architecture.

Recommendations

As a corollary to data collection, this thesis lacks a comprehensive evaluation of the different methods discussed in chapter three. In order to effectively measure the performance of various collaborative filtering approaches, one needs access to a sufficiently large data set. I believe that with additional time and effort such a data set can be obtained for architectural design, specifically in the residential sector, and then, the collaborative filtering methods discussed in this thesis will be able to accurately match users and design solutions.

Usability

For any user-centered architecture framework to be successful, it must first be usable. UDesign was implemented as a web application following standard usability design guidelines and an iterative UI design process. However, there is much more that can be done in terms of prototyping, UI design, and user testing.

Contributions

In this thesis, I propose a framework for user-centered architecture, called UDesign, and describe its implementation as a web application that allows users to design their own custom apartment. UDesign includes a sample one-bedroom apartment which users can customize through a kit-of-parts approach, i.e., a catalog of rooms (called assemblies), that can be combined to create a complete floor plan solution. While available configuration tools in architecture require the user to think like an expert, e.g., integrate form and function, UDesign takes a novel approach by deploying a suite of machine learning algorithms coupled with data from Facebook to model users' design preferences and match them with design solutions. Users can take advantage of these recommendations as their design starting point and continue to explore other alternatives by dragging and dropping rooms from the catalog on to the sample floor plan. As users explore design solutions, UDesign updates its recommendations to guide users through the design space and helps them find solutions that best fit their needs. Finally, UDesign's integration with Facebook, allows users to share their designs, making UDesign part of their social network.

References & Suggested Reading List

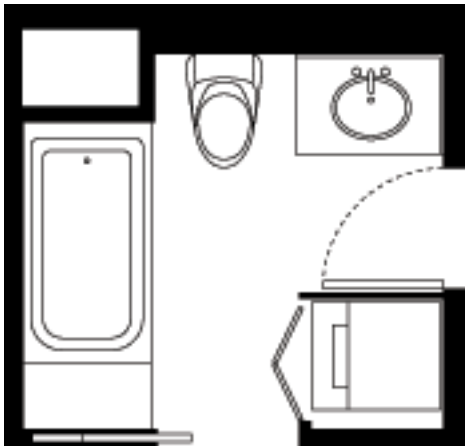
1. Winston, P. H. (1970). Learning Structural Descriptions From Examples. AI Technical Reports, Artificial Intelligence Lab, Massachusetts Institute of Technology, Cambridge, Massachusetts.
2. McLuhan, M. (1964). Understanding Media: The Extensions of Man. 1st Ed. McGraw Hill, NY, 1964; reissued MIT Press, 1994, with introduction by Lewis H. Lapham; reissued by Gingko Press, 2003 ISBN 1-58423-073-8

Appendix

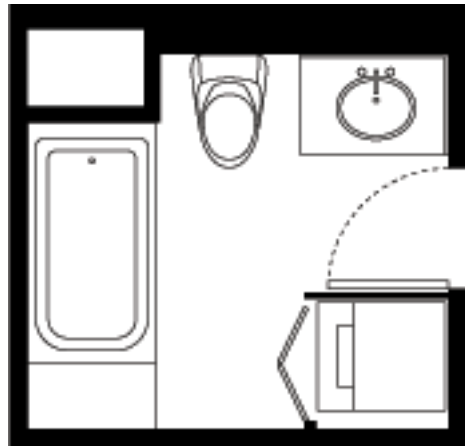
A compilation of *assemblies* (rooms) and *solutions* (floor plans) included in UDesign and throughout the thesis. Total number of: assemblies = 48 / solutions = 285. Only 20 of each are shown here.

Assemblies

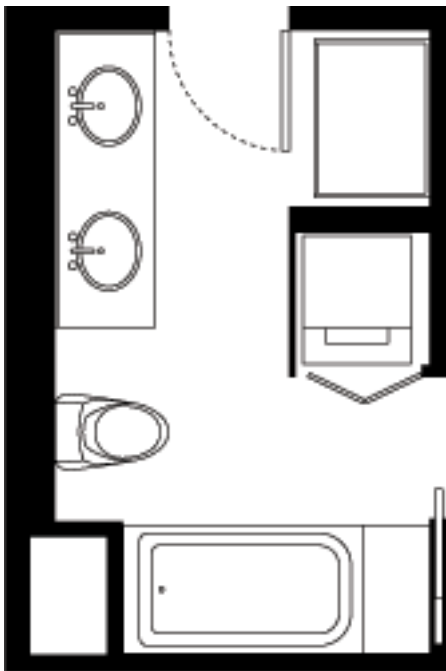
A11BT001



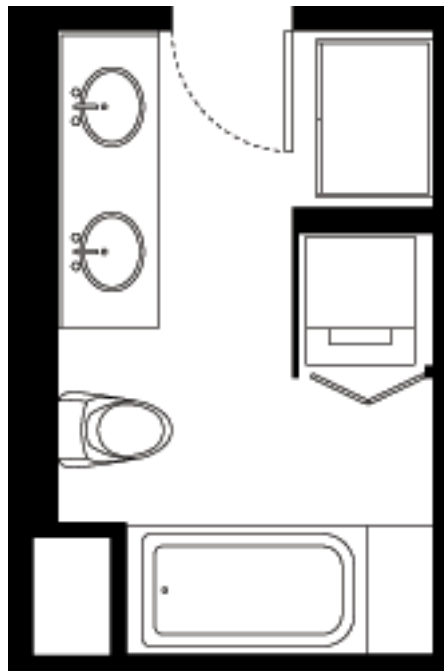
A11BT002



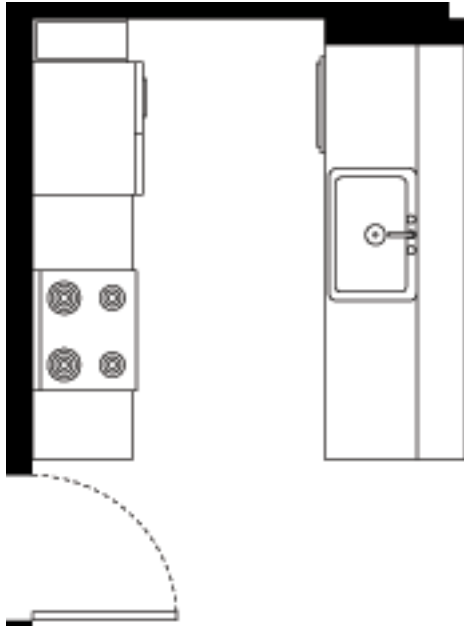
A12BT001



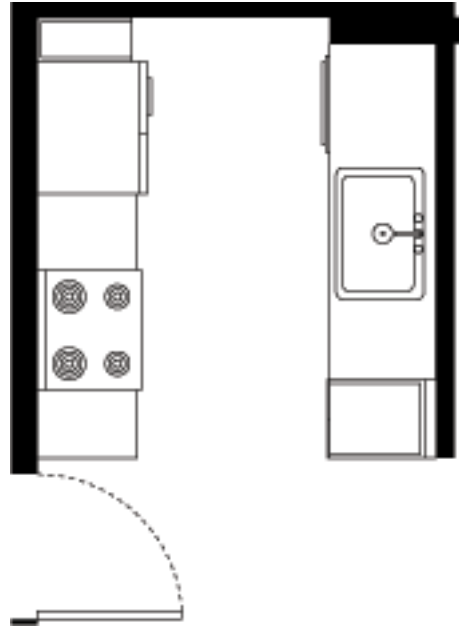
A12BT002



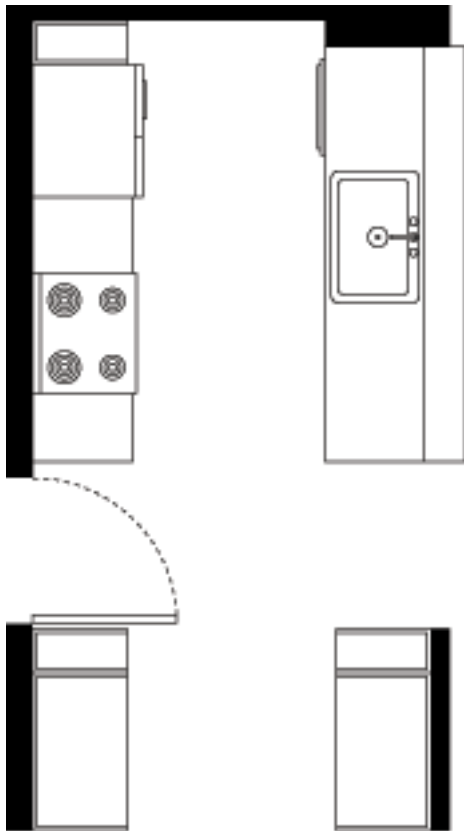
A21KT001



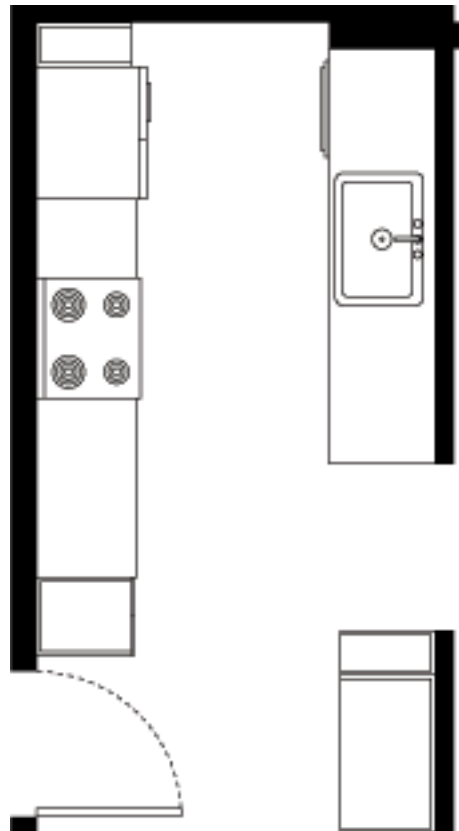
A21KT002



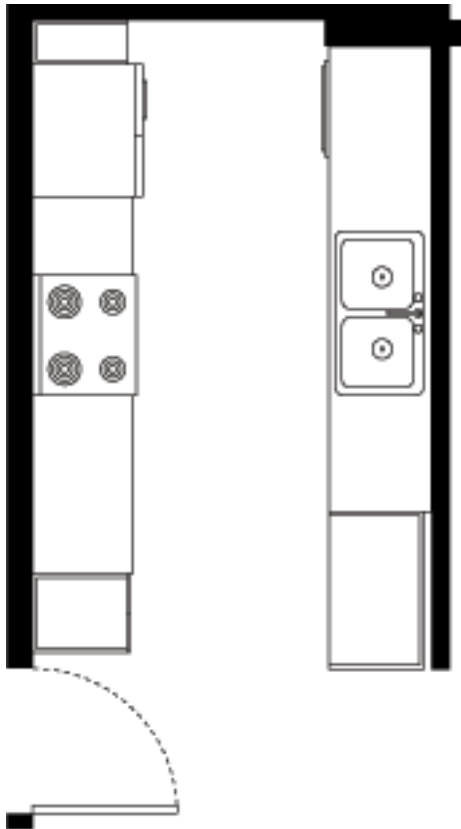
A22KT001



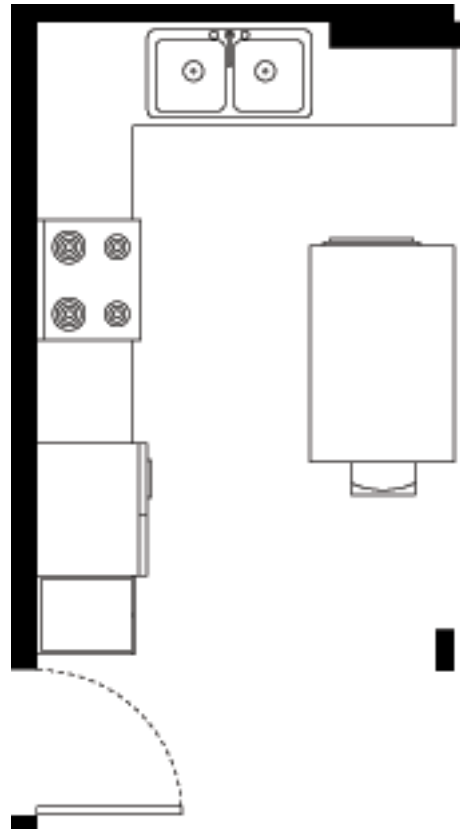
A22KT002



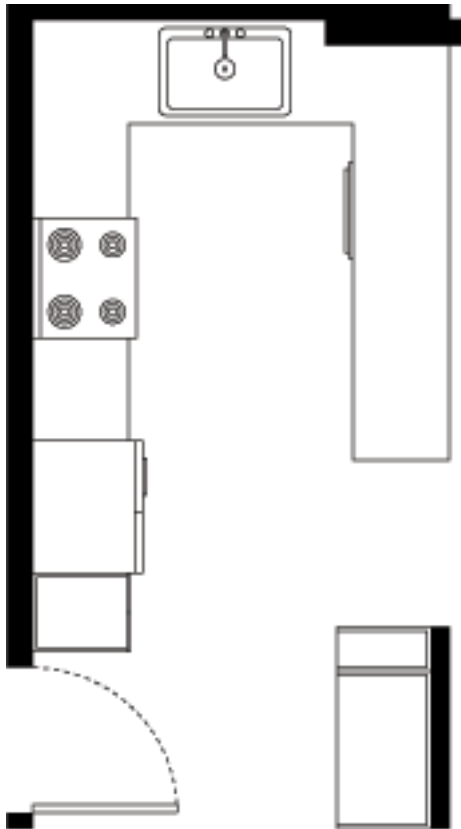
A22KT003



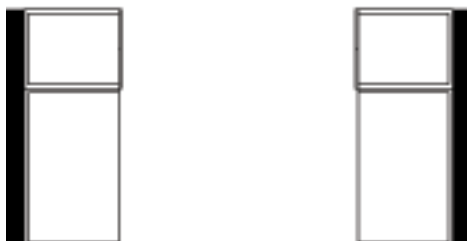
A22KT004



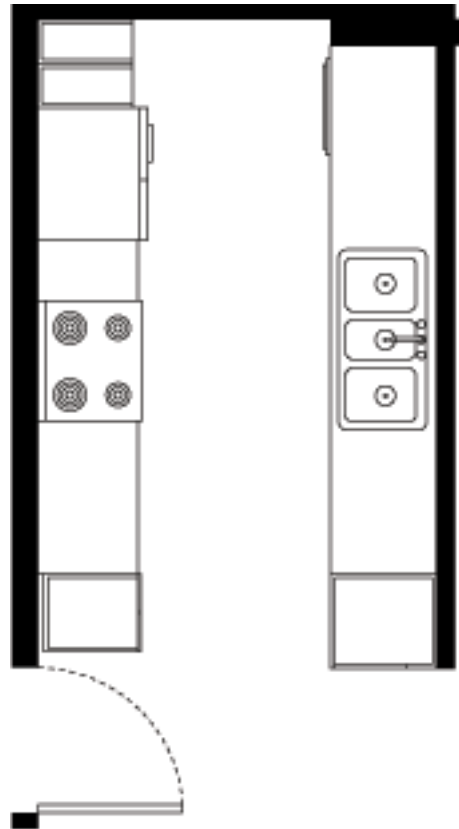
A22KT005



B11CL001



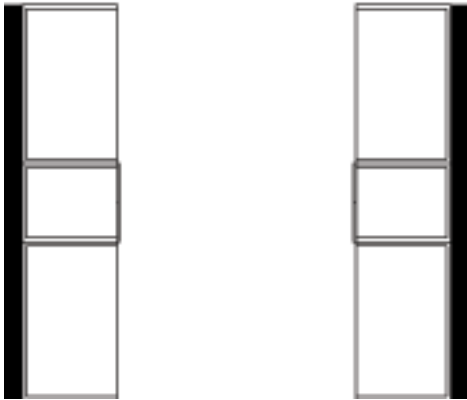
A22KT006



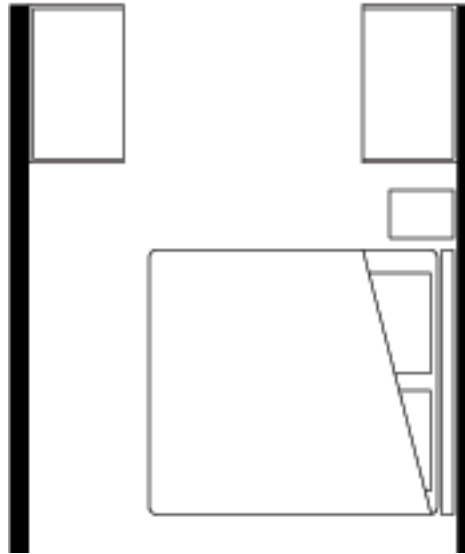
B12BD001



B12CL001



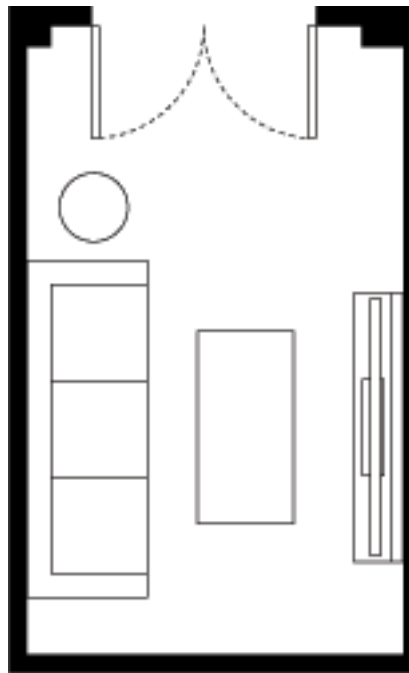
B13BD001



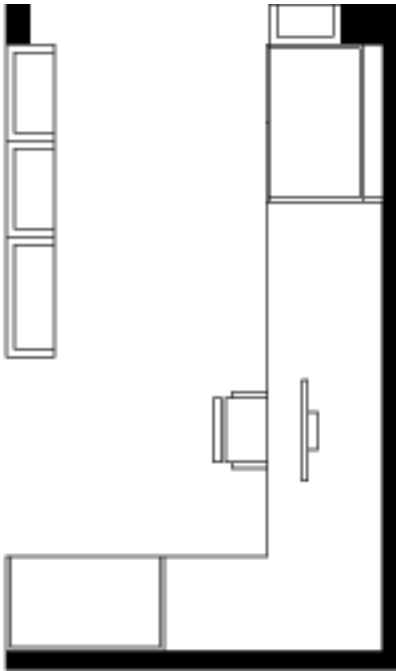
B14CL001



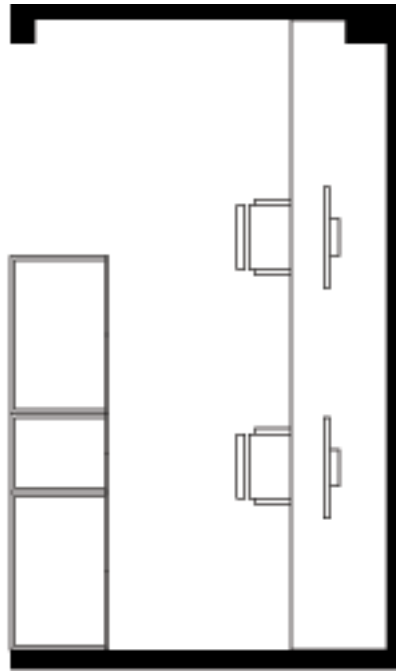
B15ME001



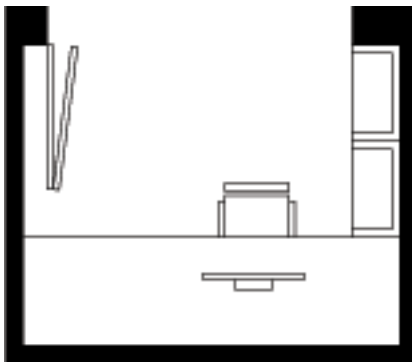
B15ST001



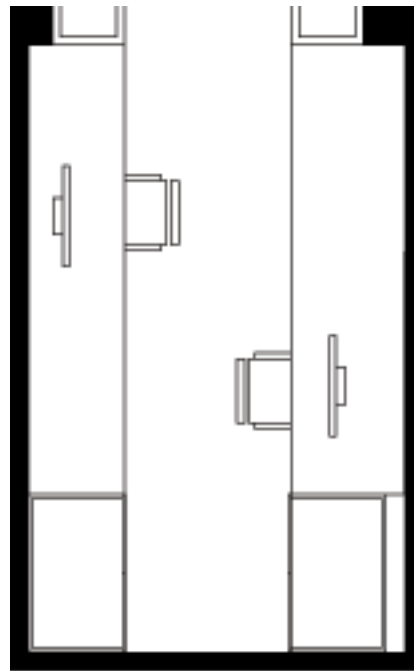
B15ST002



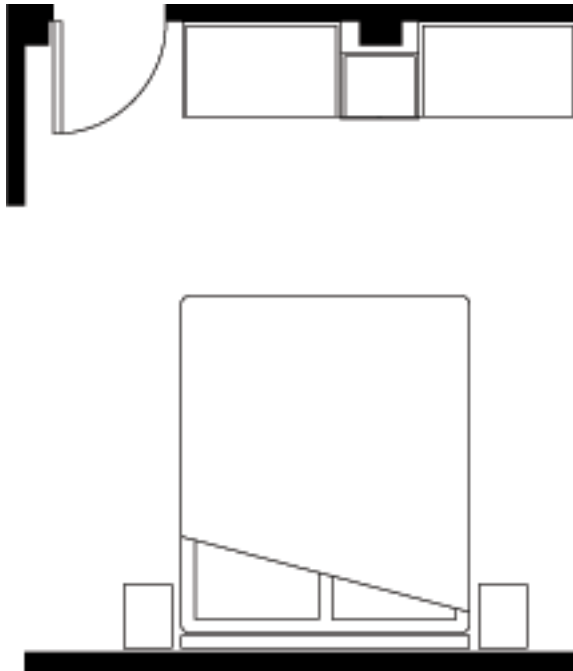
B15ST003



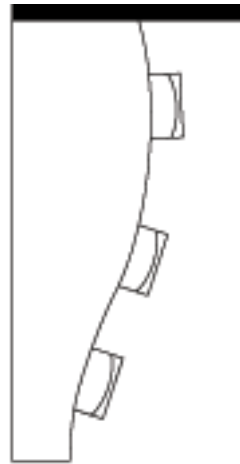
B15ST004



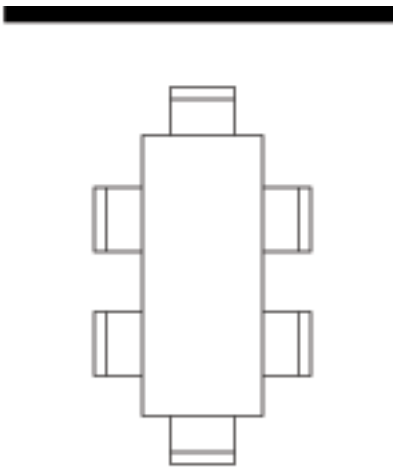
B16BD001



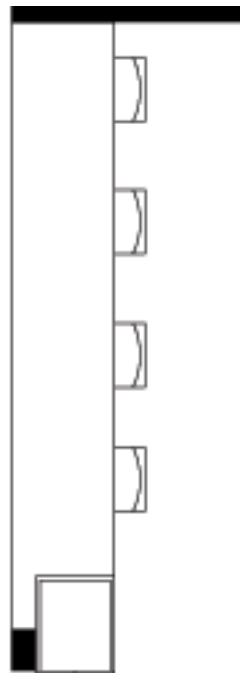
B21DN001



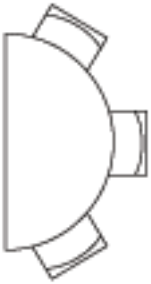
B22DN001



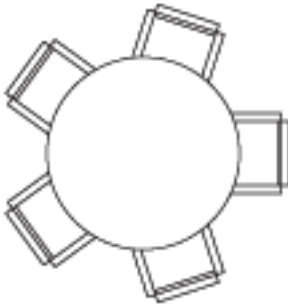
B24DN001



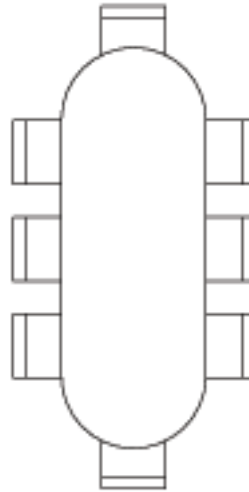
B24DN002



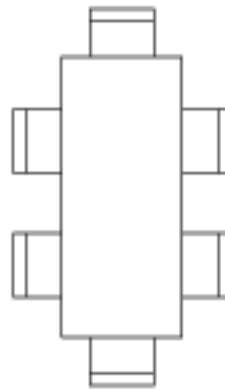
B25DN002



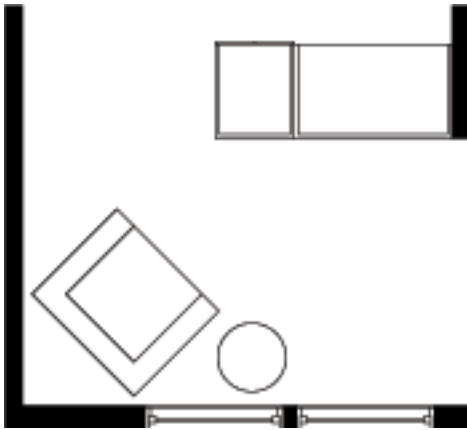
B25DN001



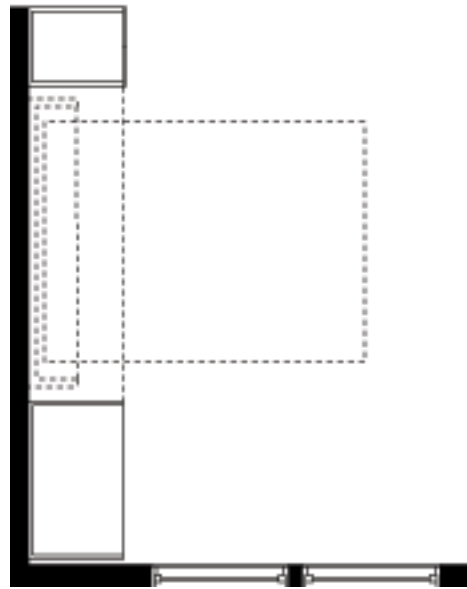
B25DN003



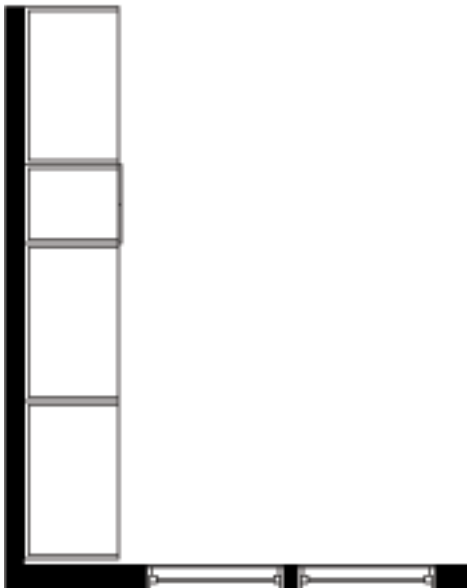
C11ST001



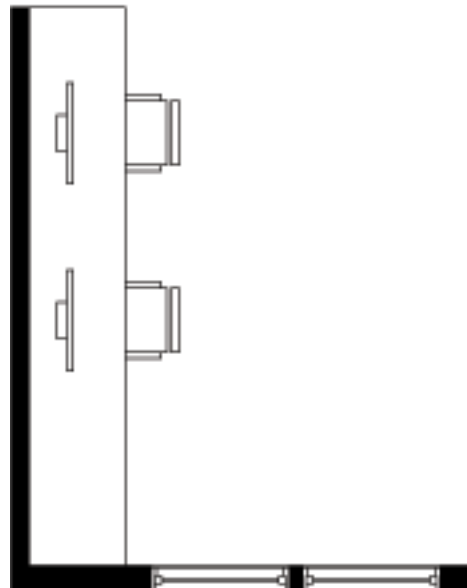
C12BD001



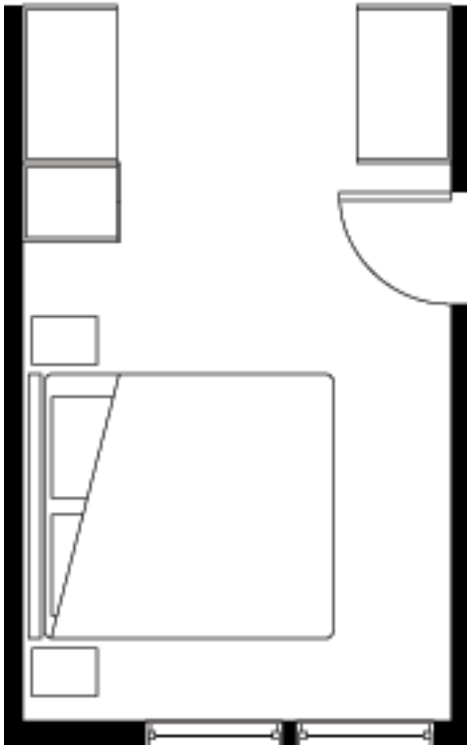
C12CL001



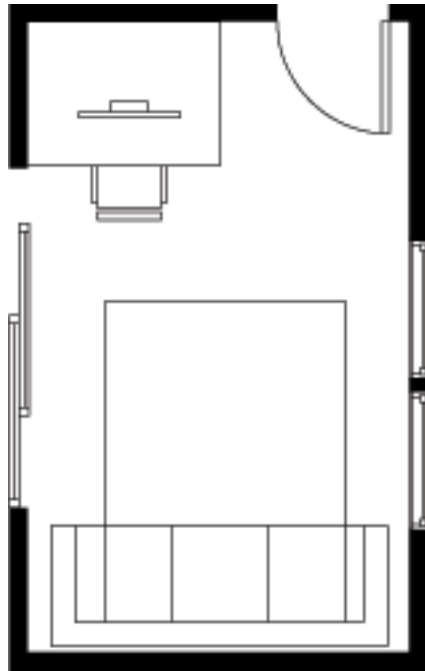
C12ST001



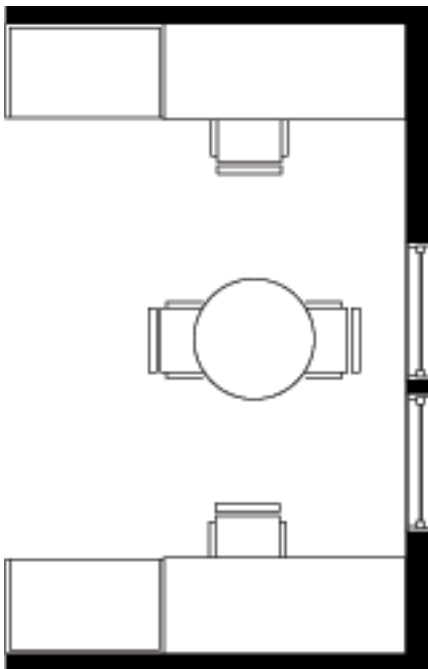
C13BD001



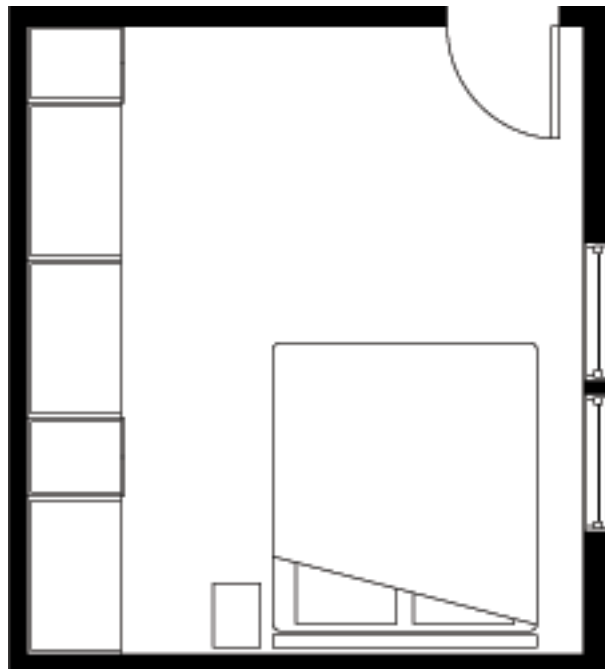
C14ST001



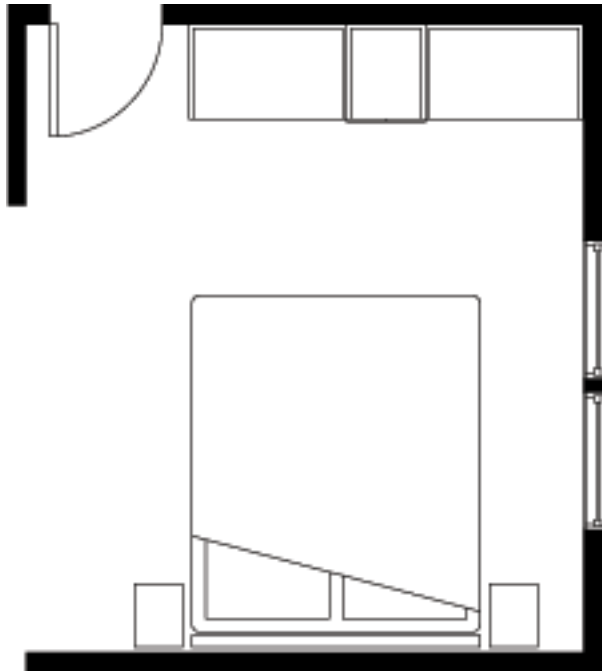
C14ST002



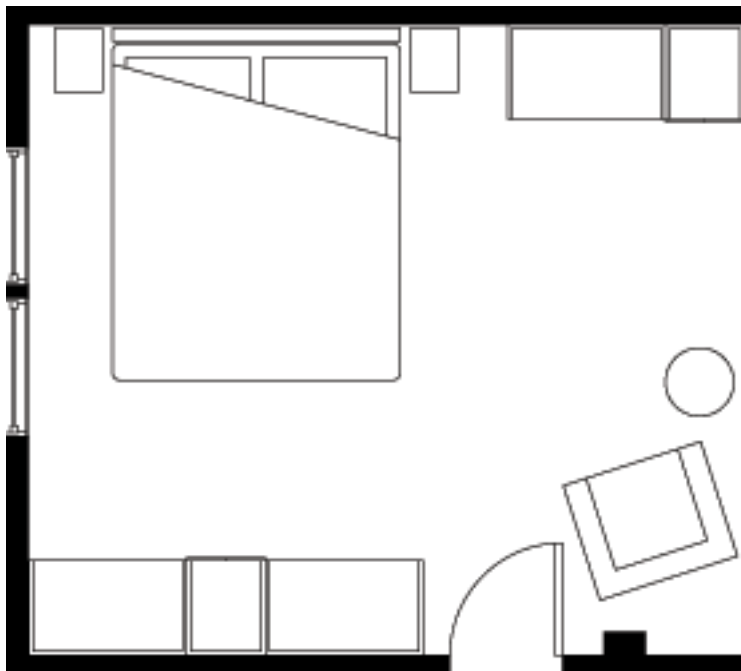
C15BD001



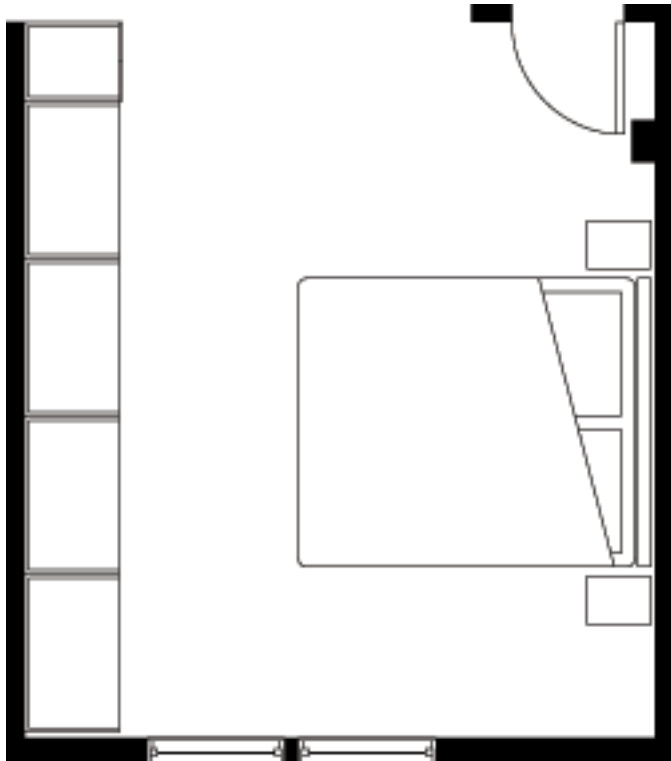
C15BD002



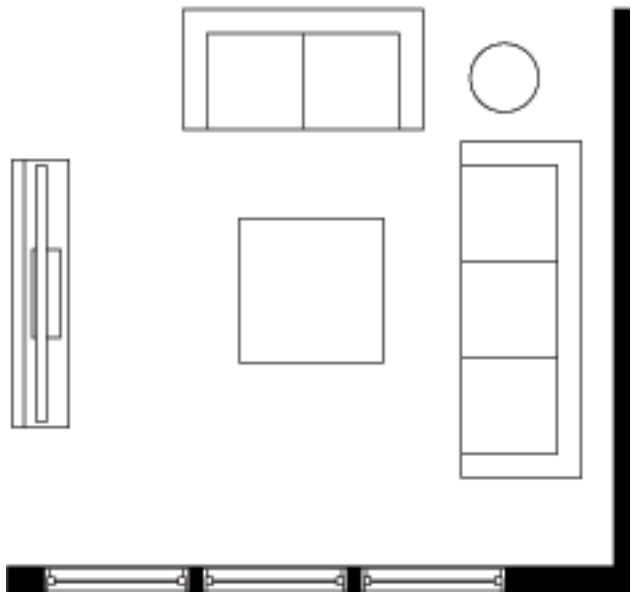
C16BD001



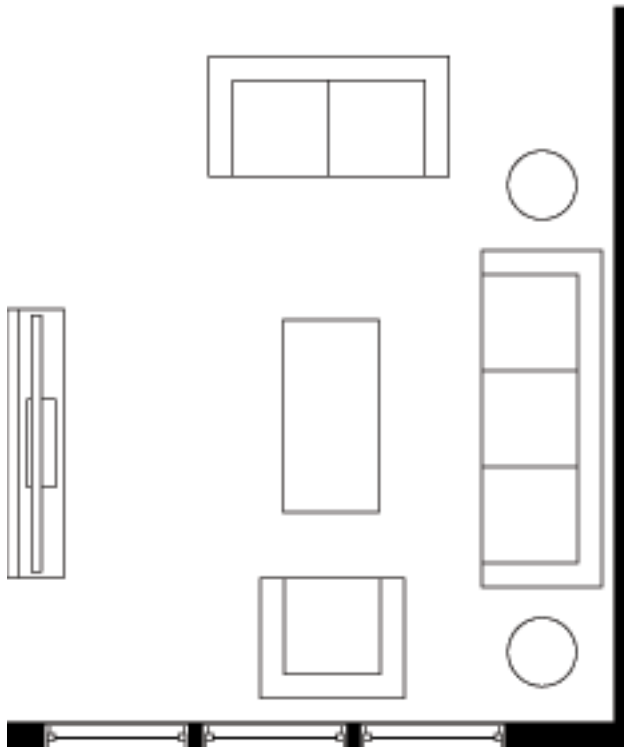
C16DB002



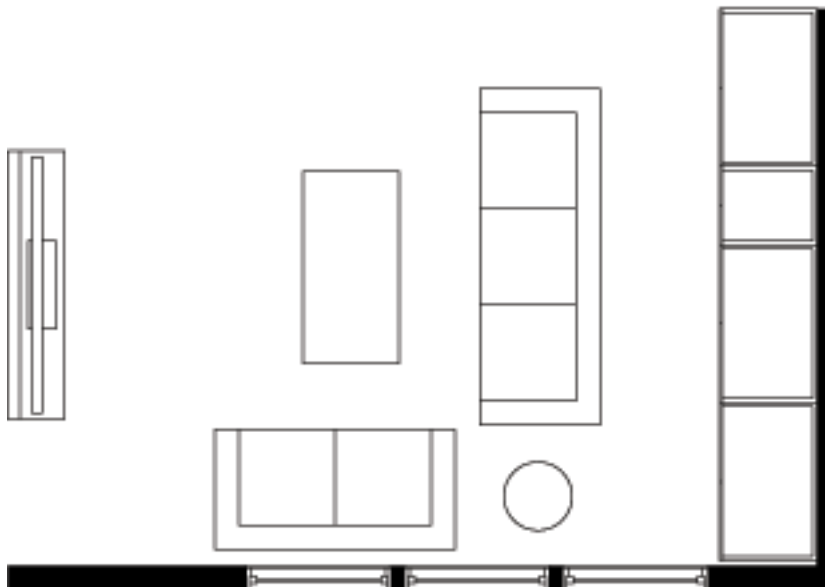
C22LV001



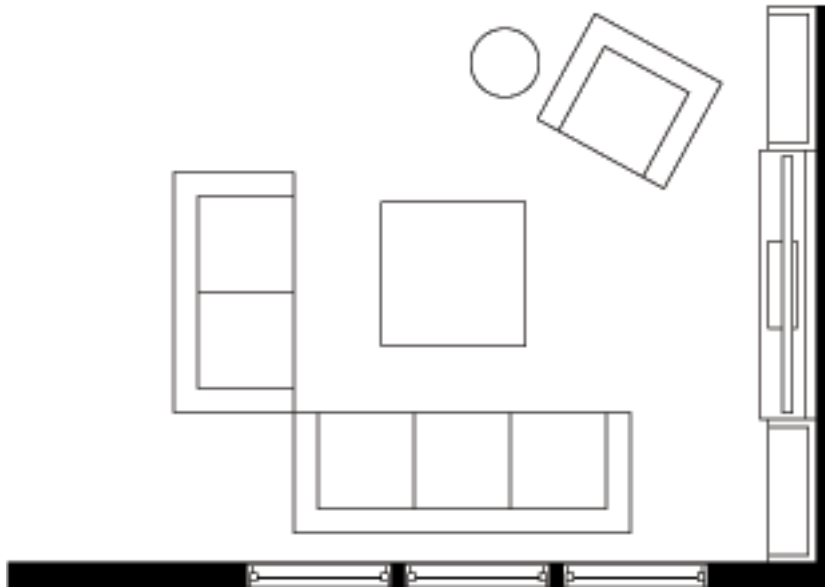
C23LV001



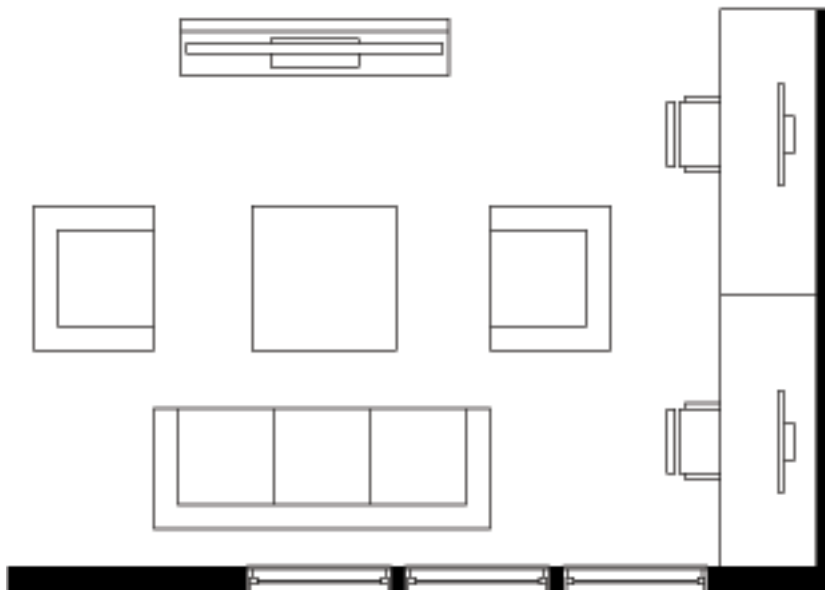
C25LV001



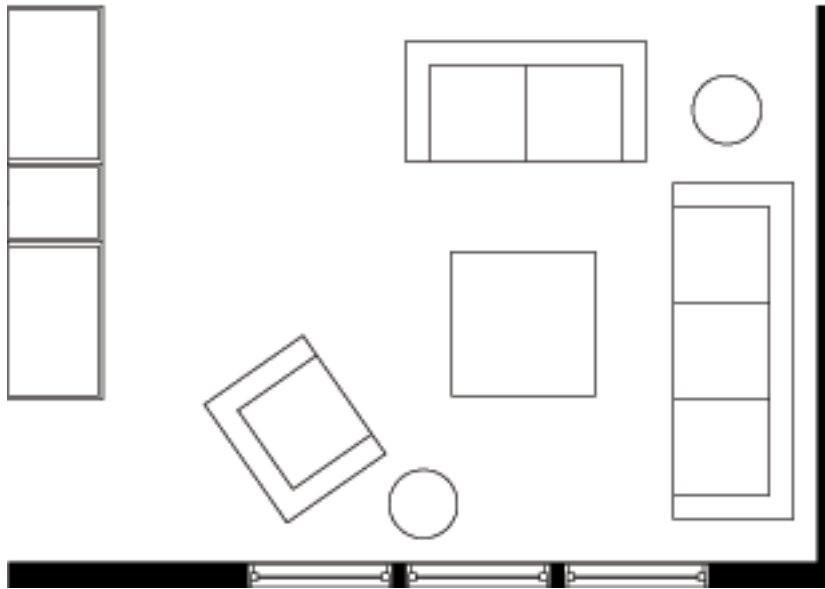
C25LV002



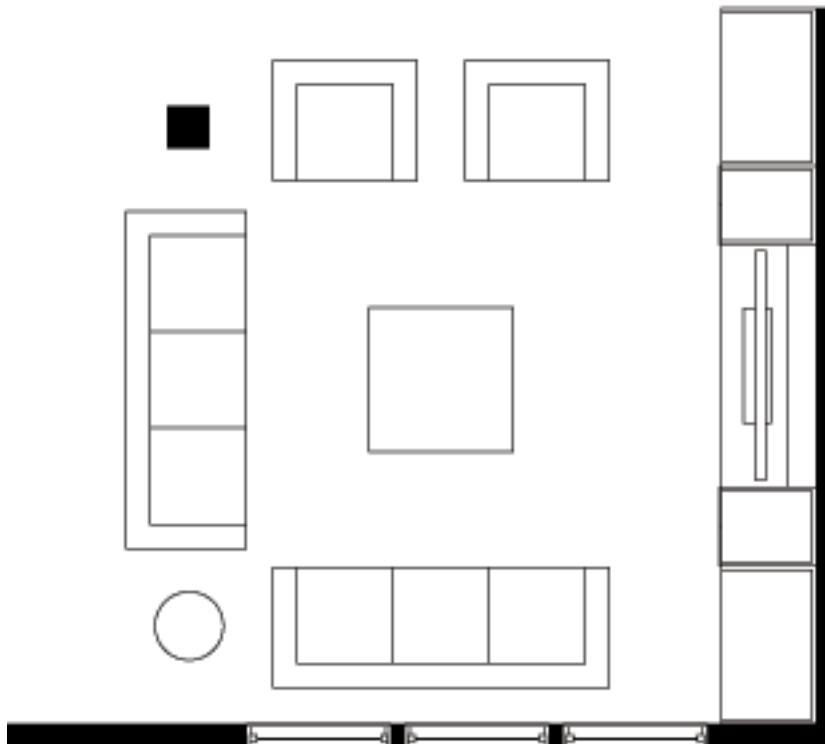
C25LV003



C25LV004

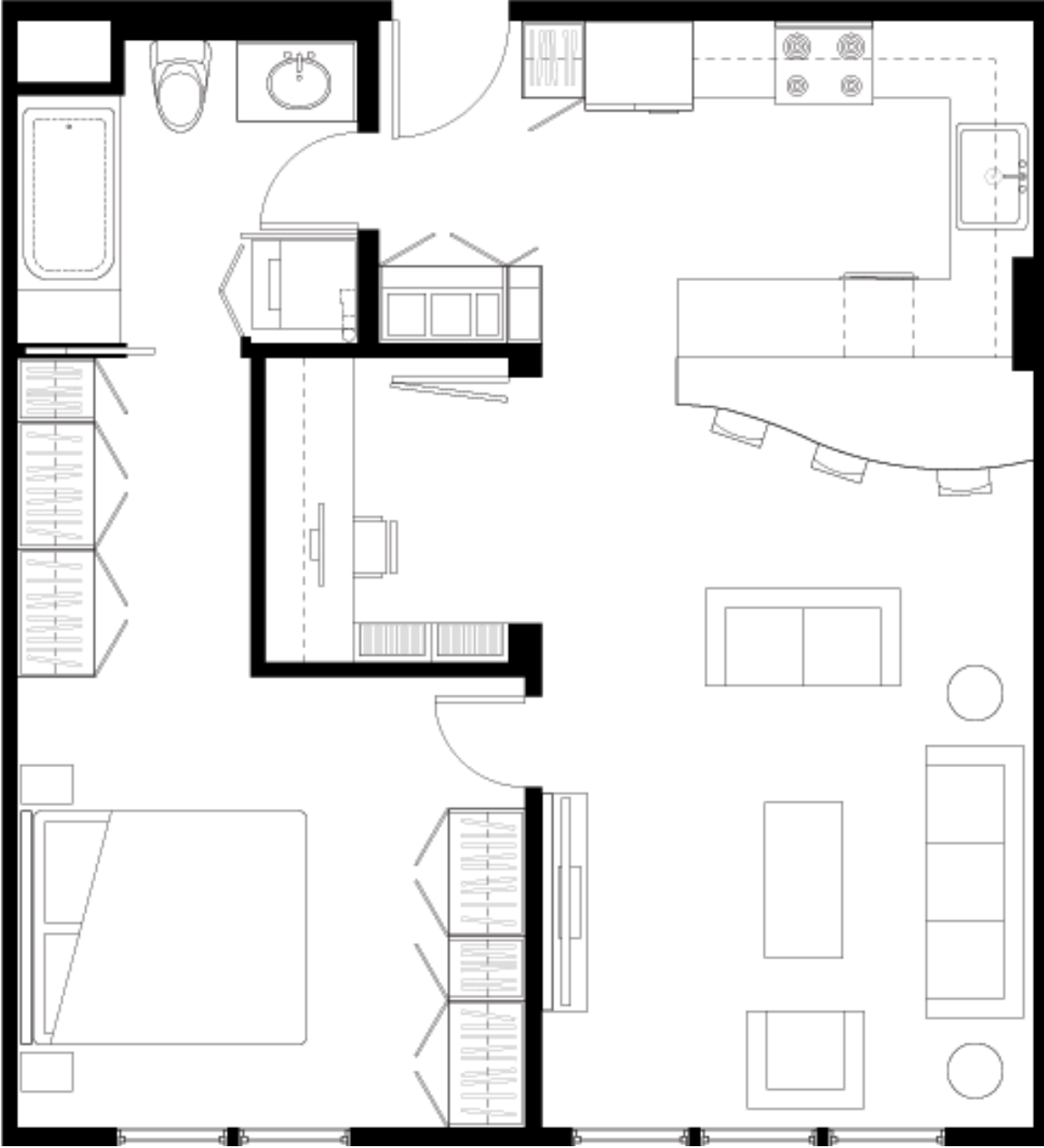


C26LV001

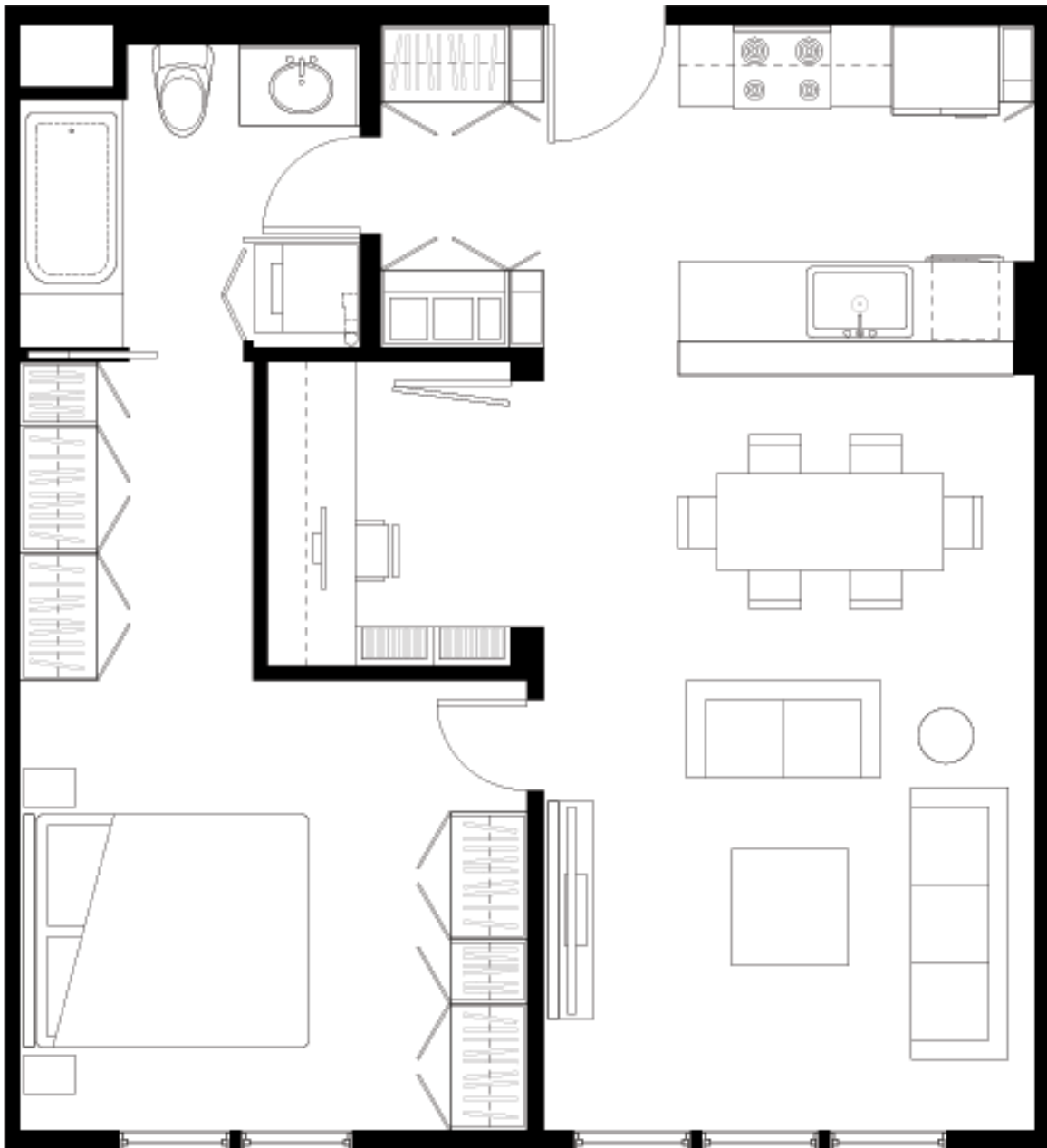


Solutions

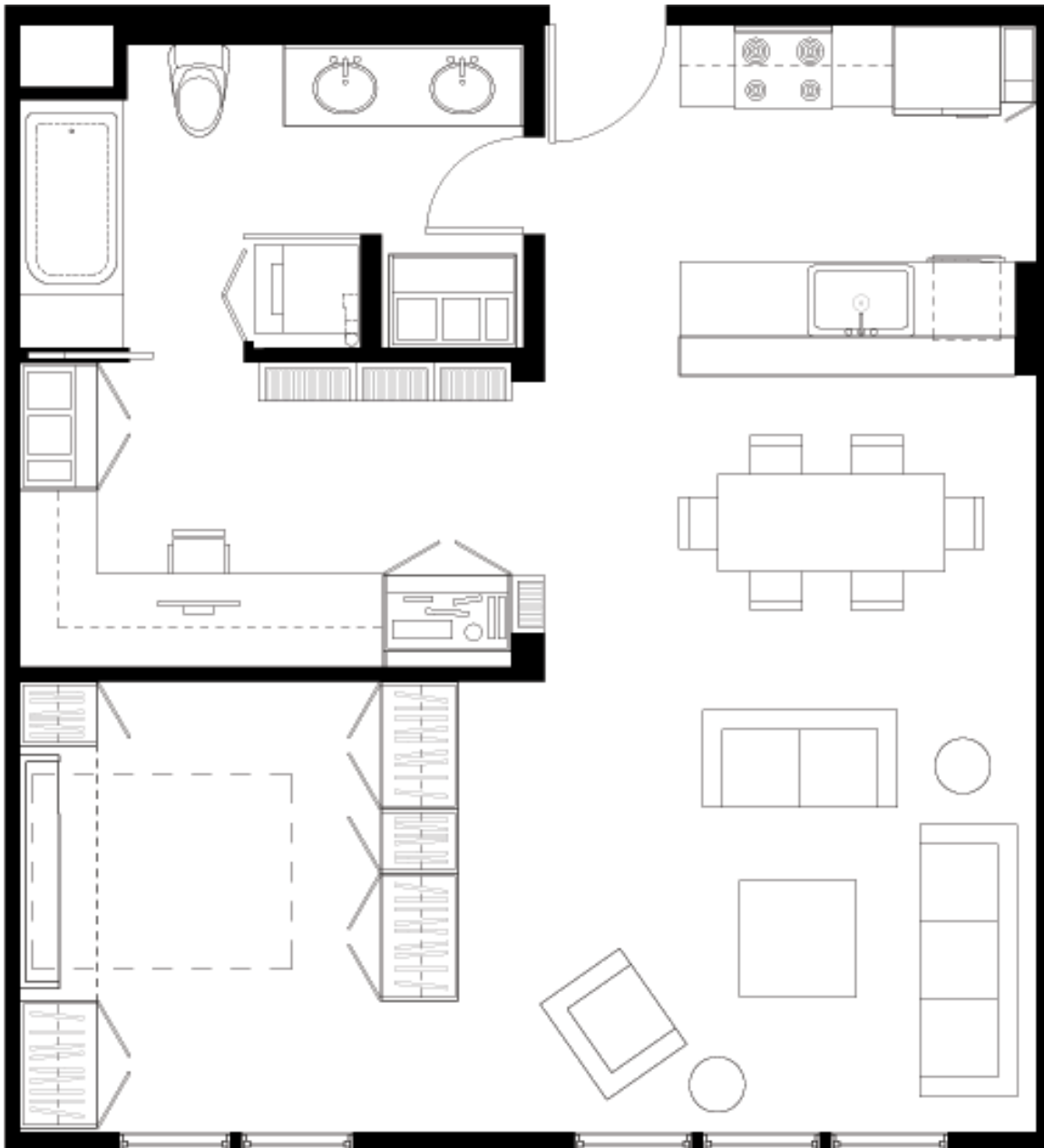
SOL9



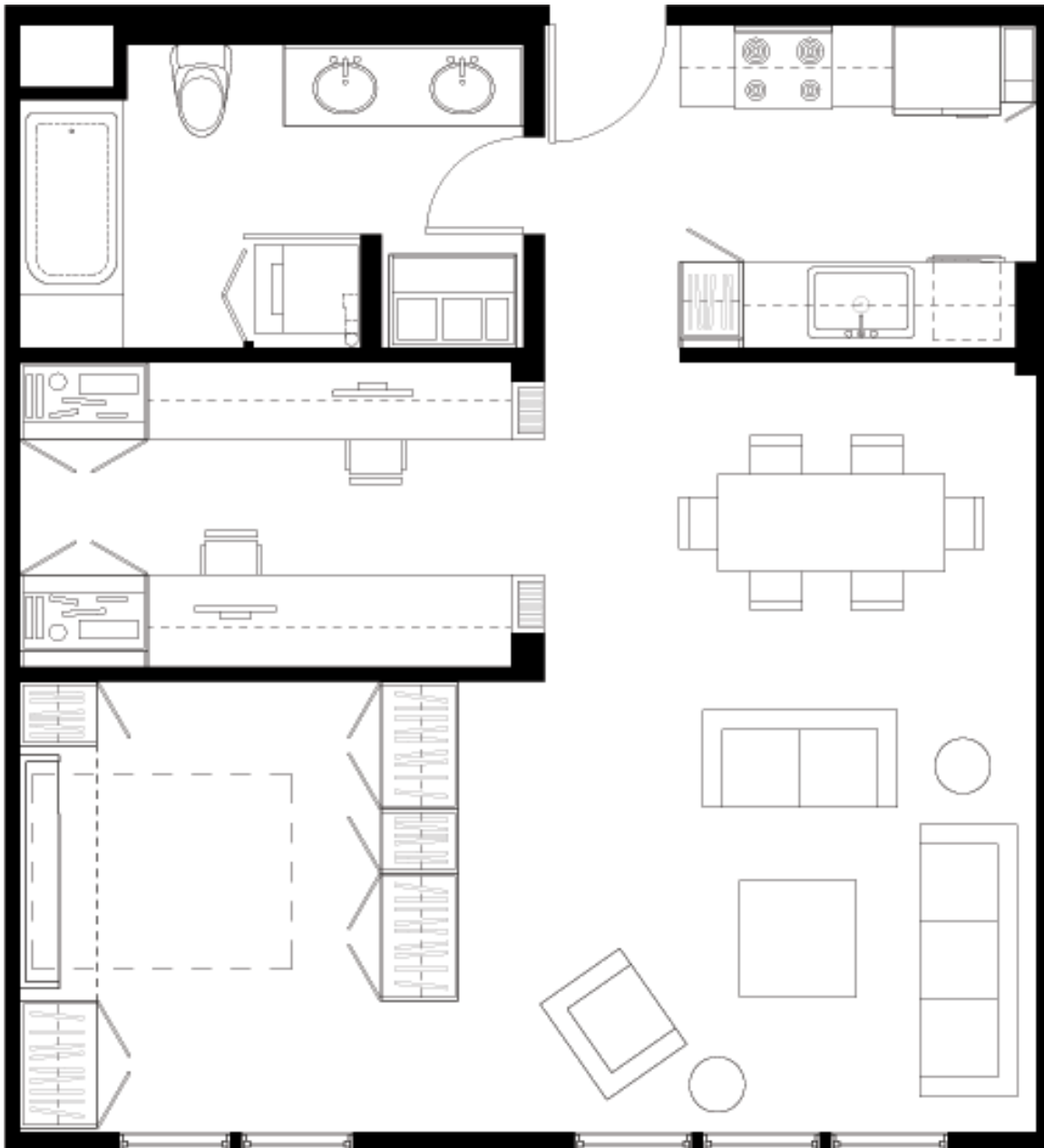
SOL25



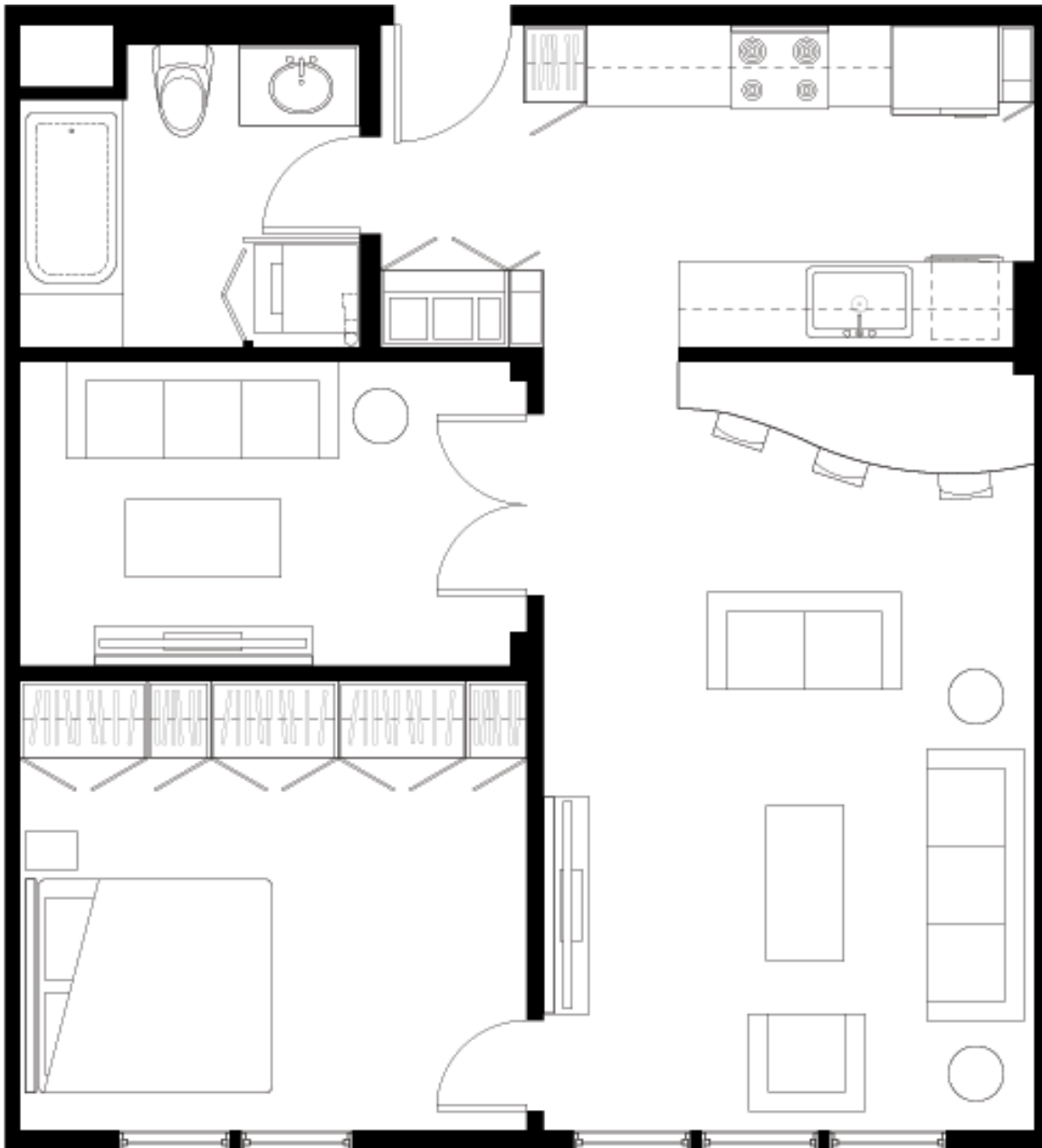
SOL33



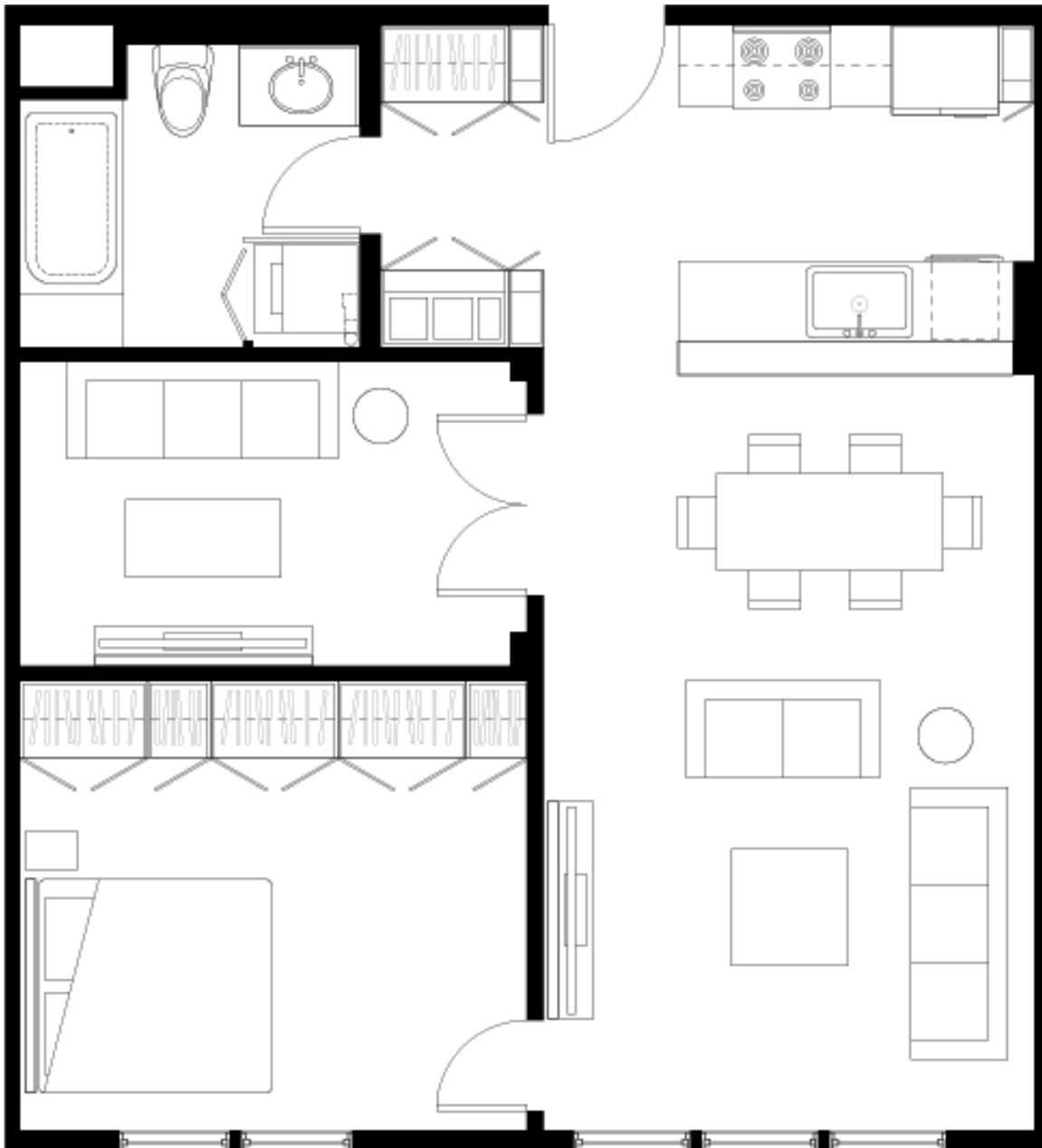
SOL38



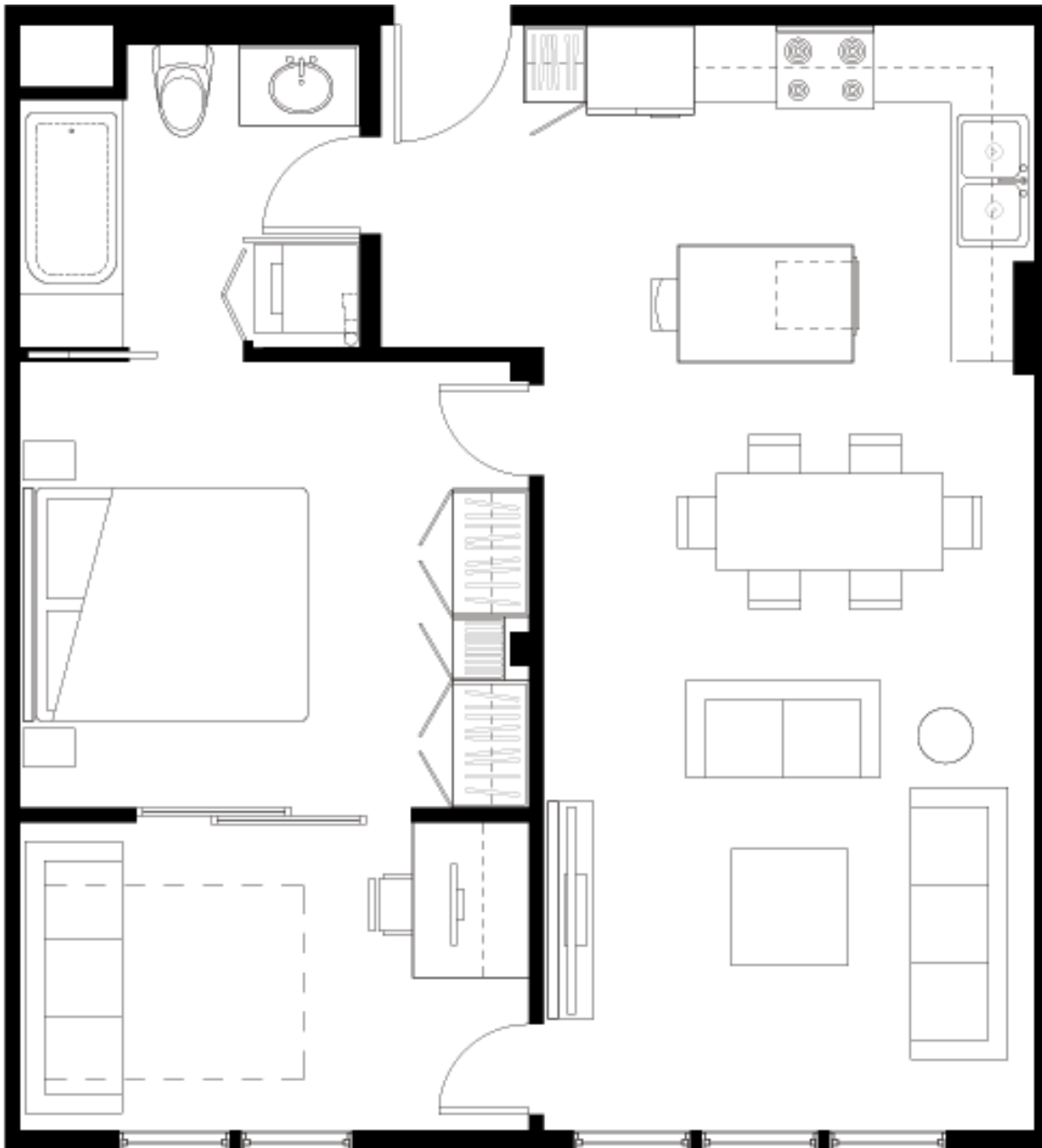
SOL54



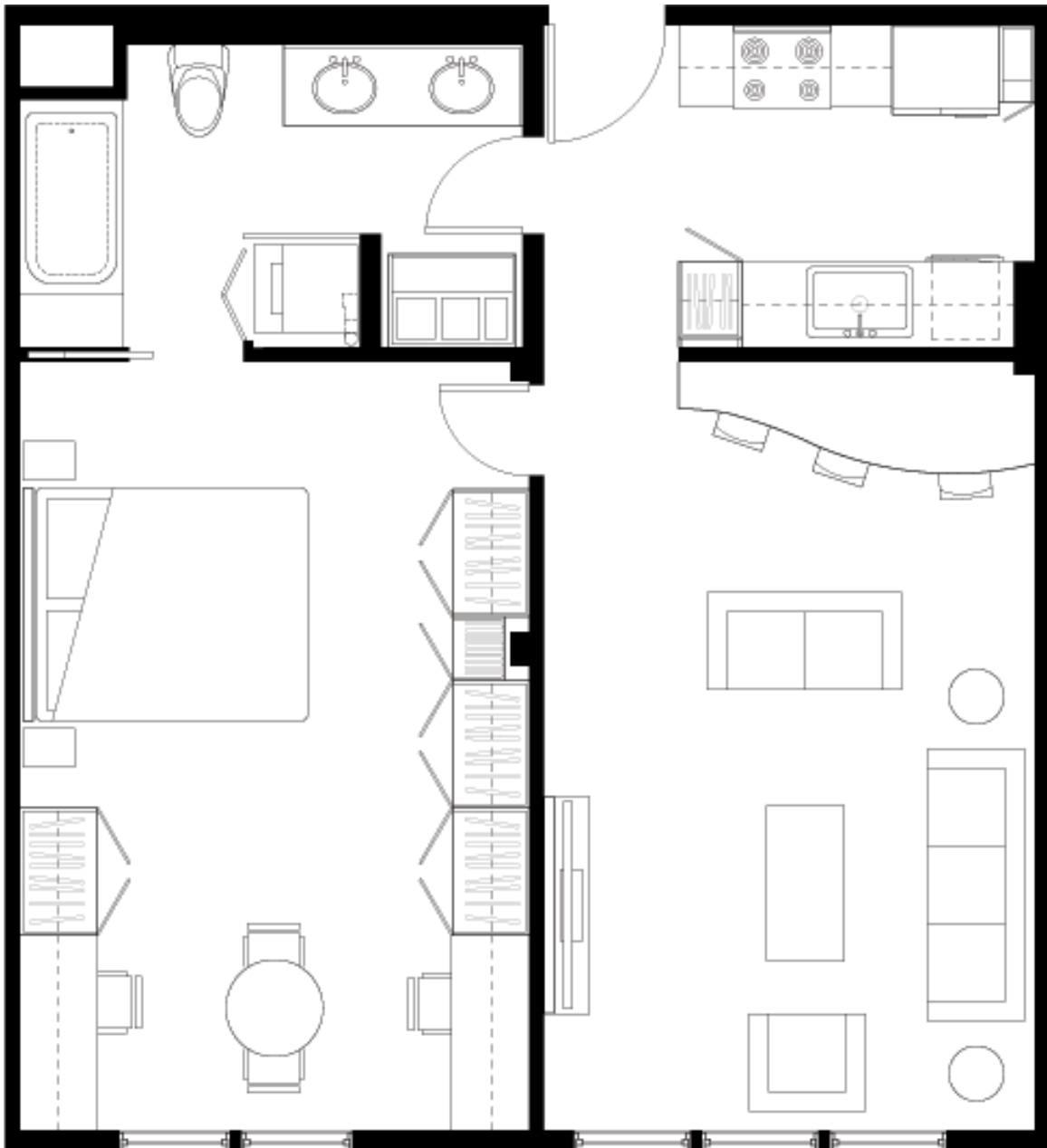
SOL59



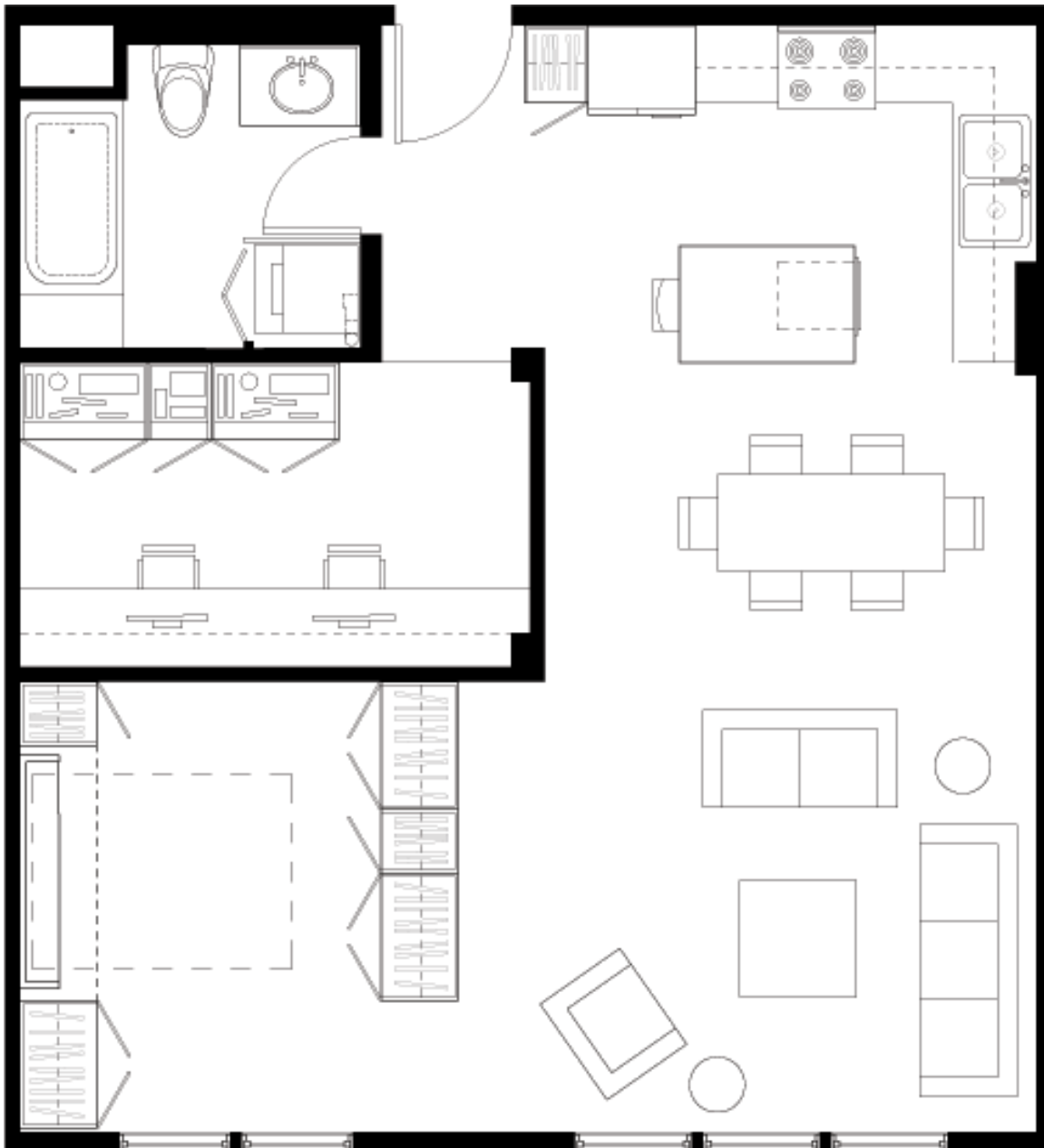
SOL74



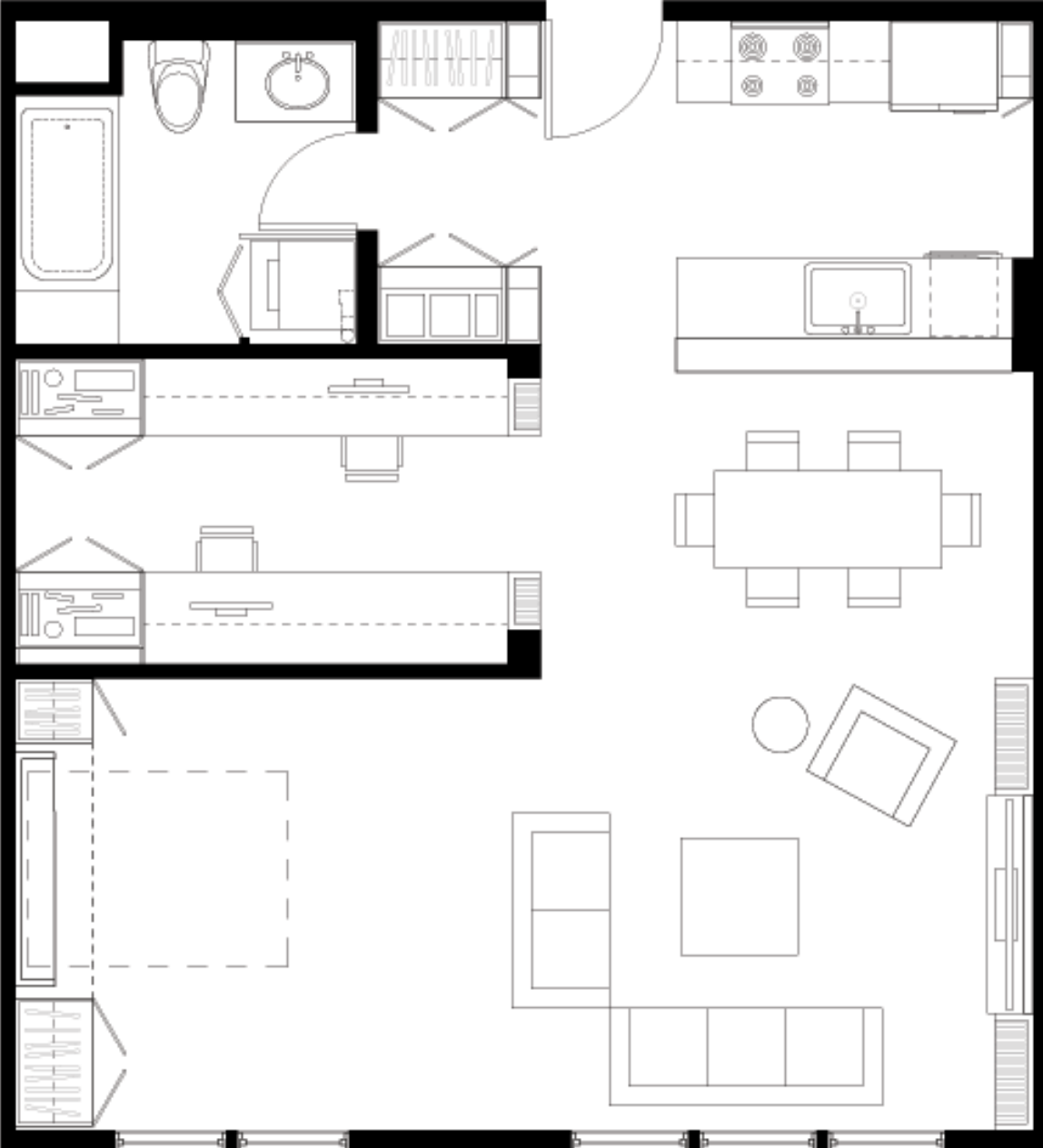
SOL81



SOL125



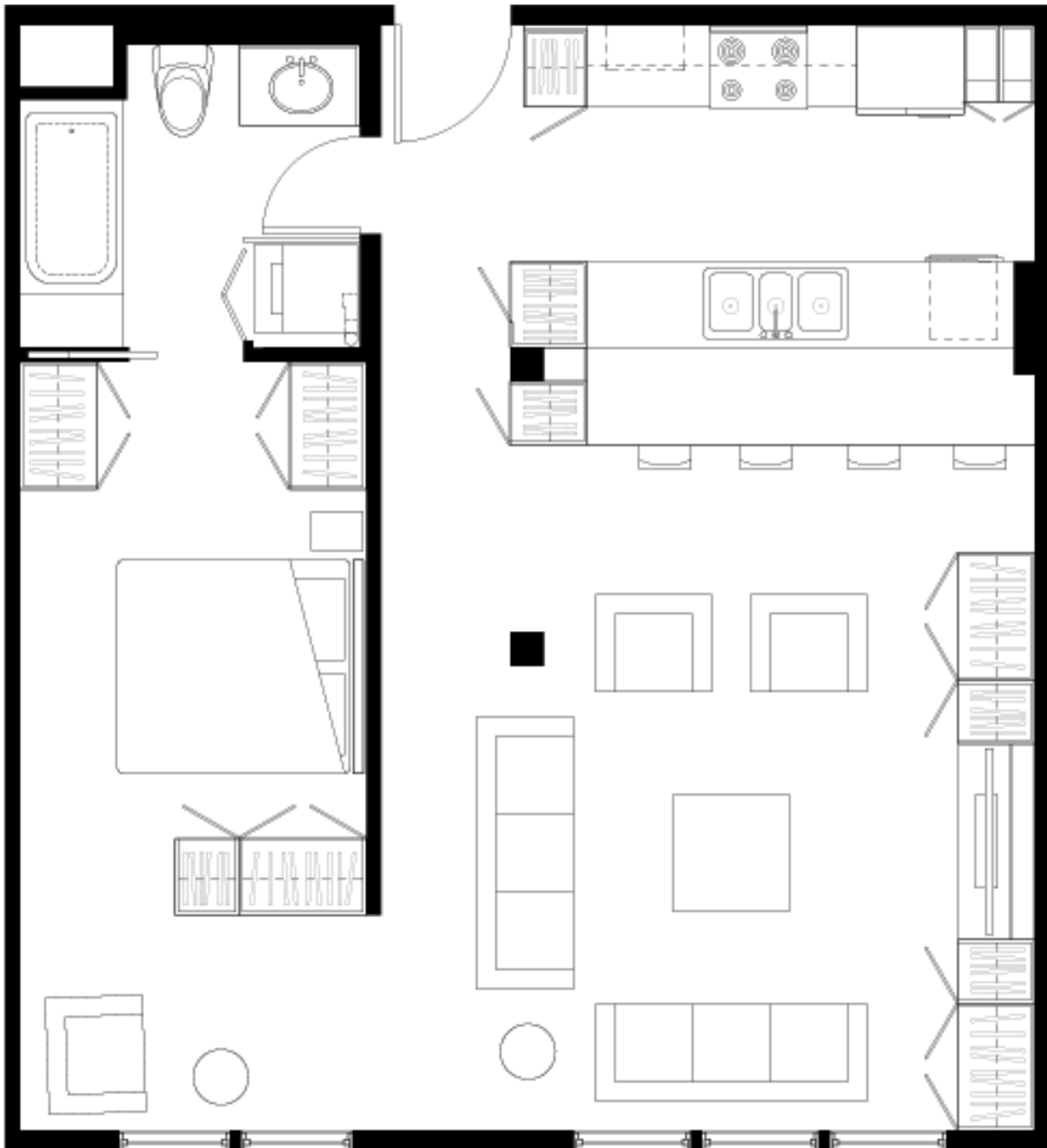
SOL139



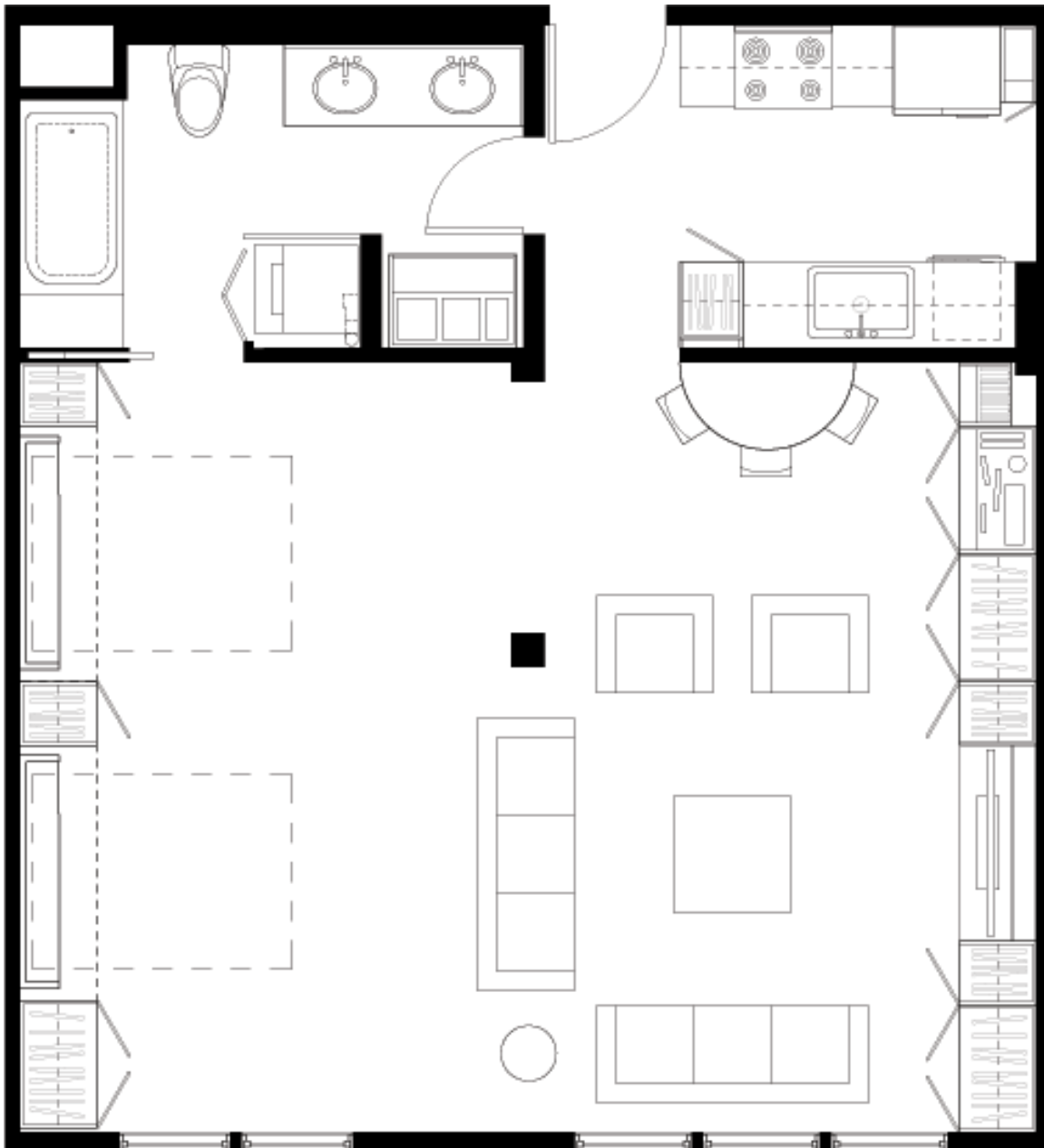
SOL144



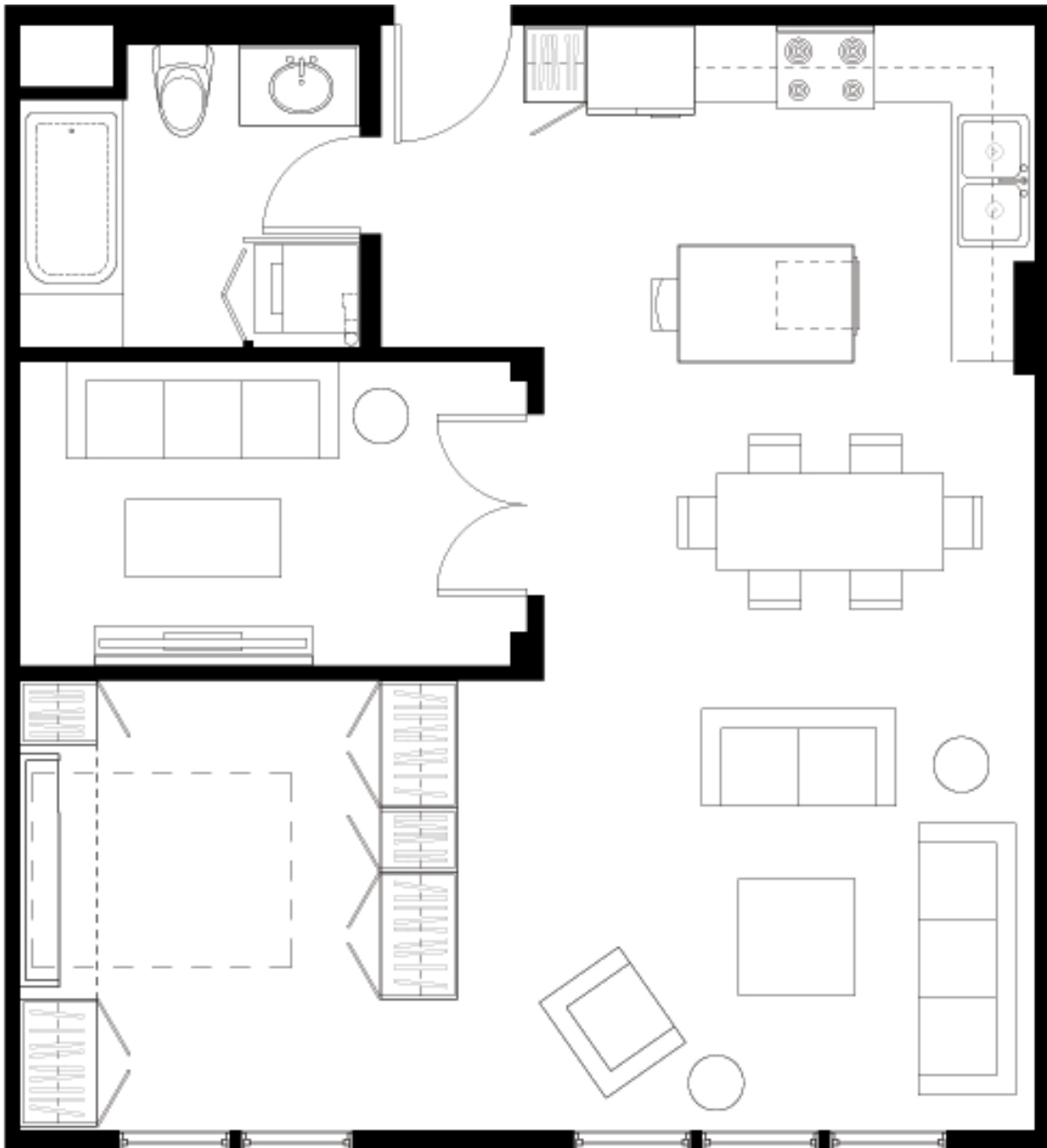
SOL149



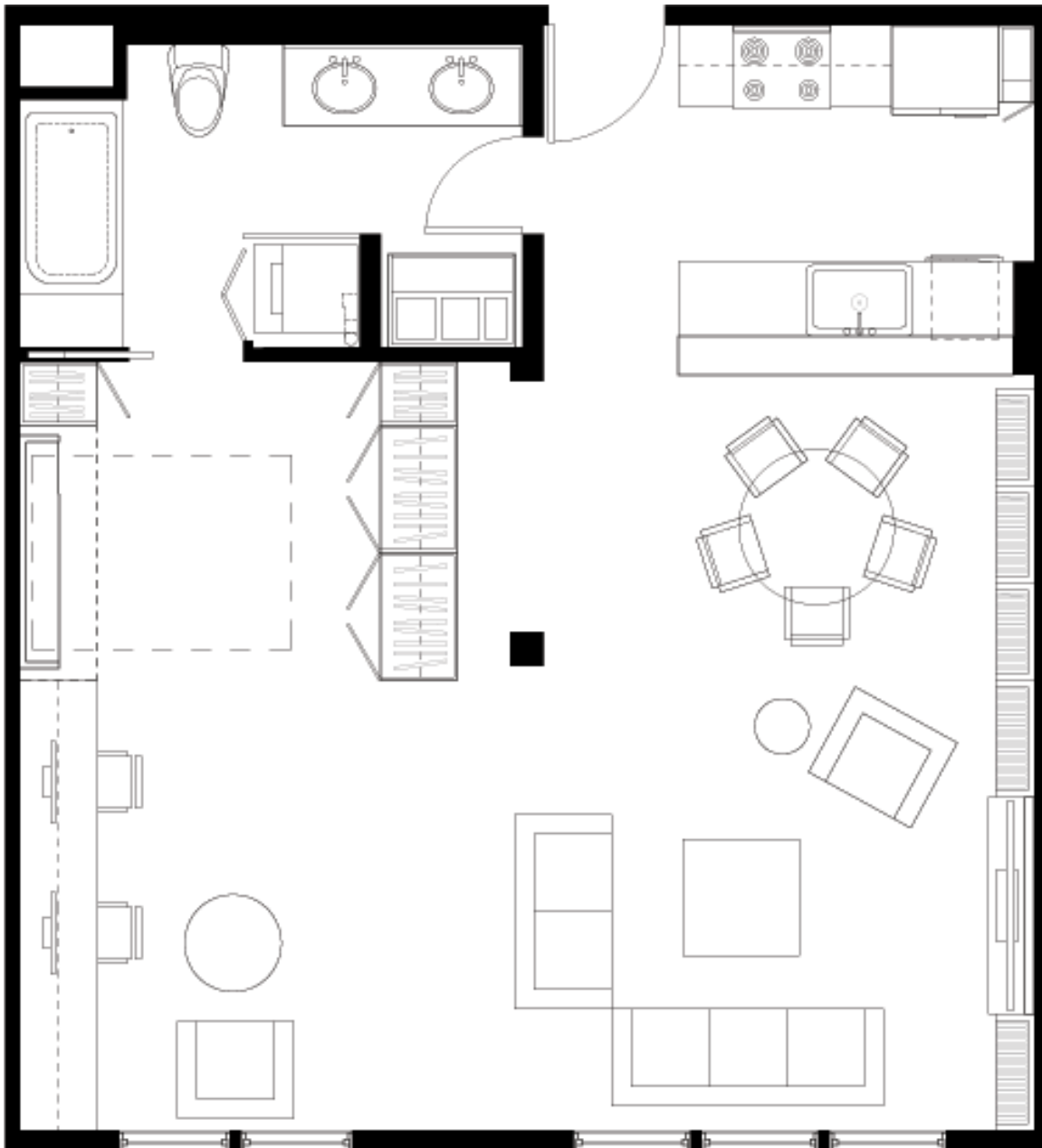
SOL150



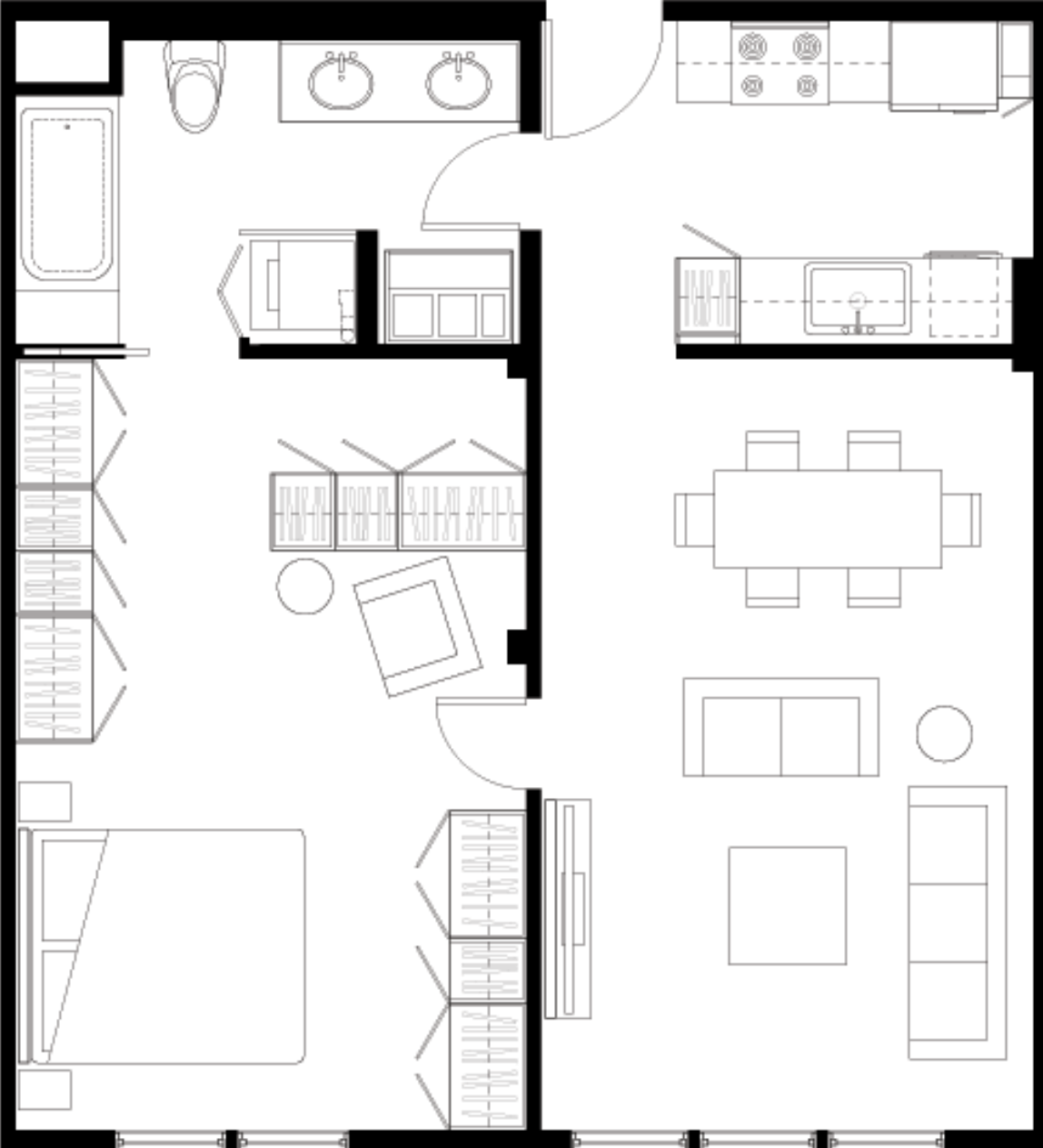
SOL158



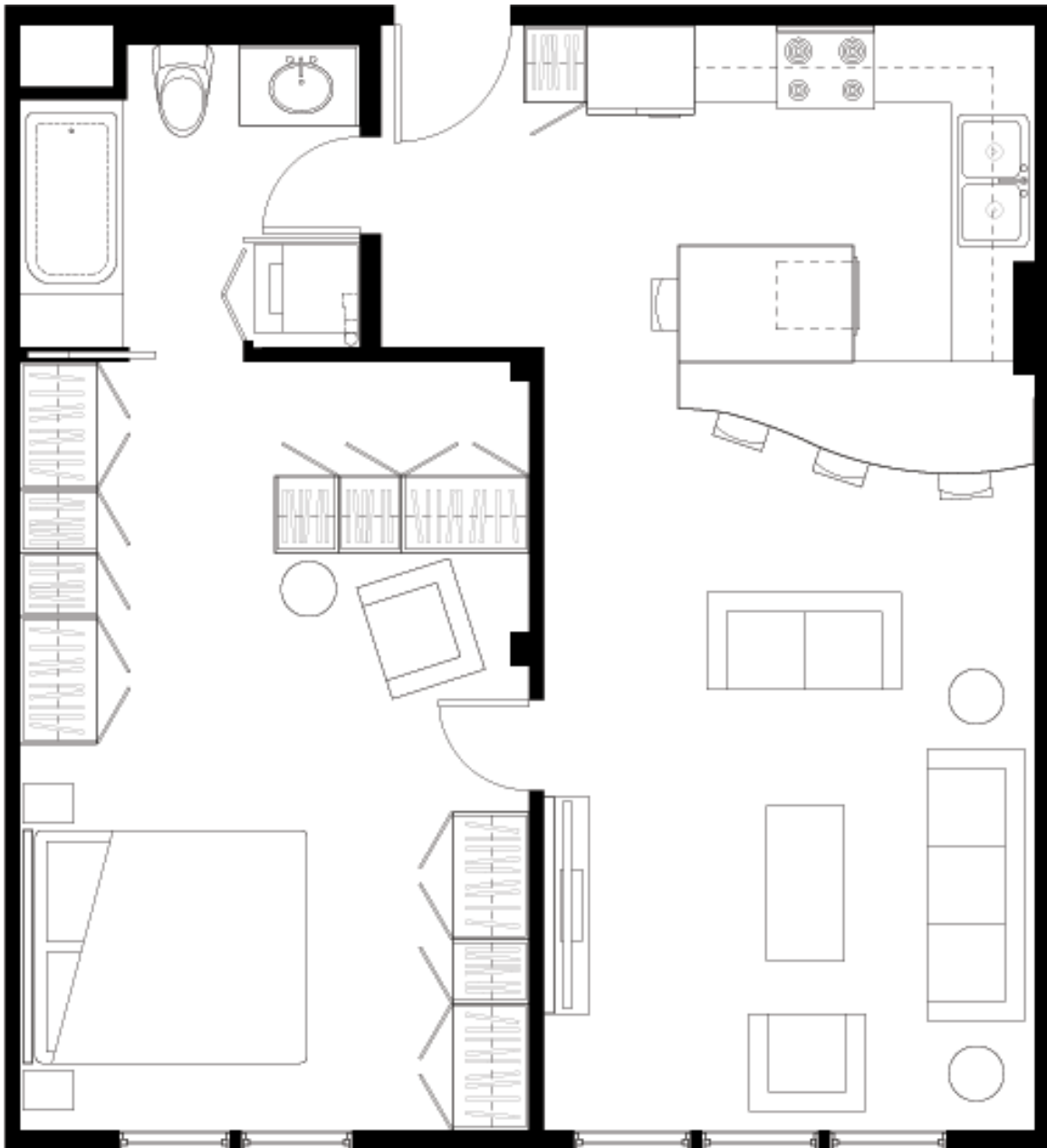
SOL169



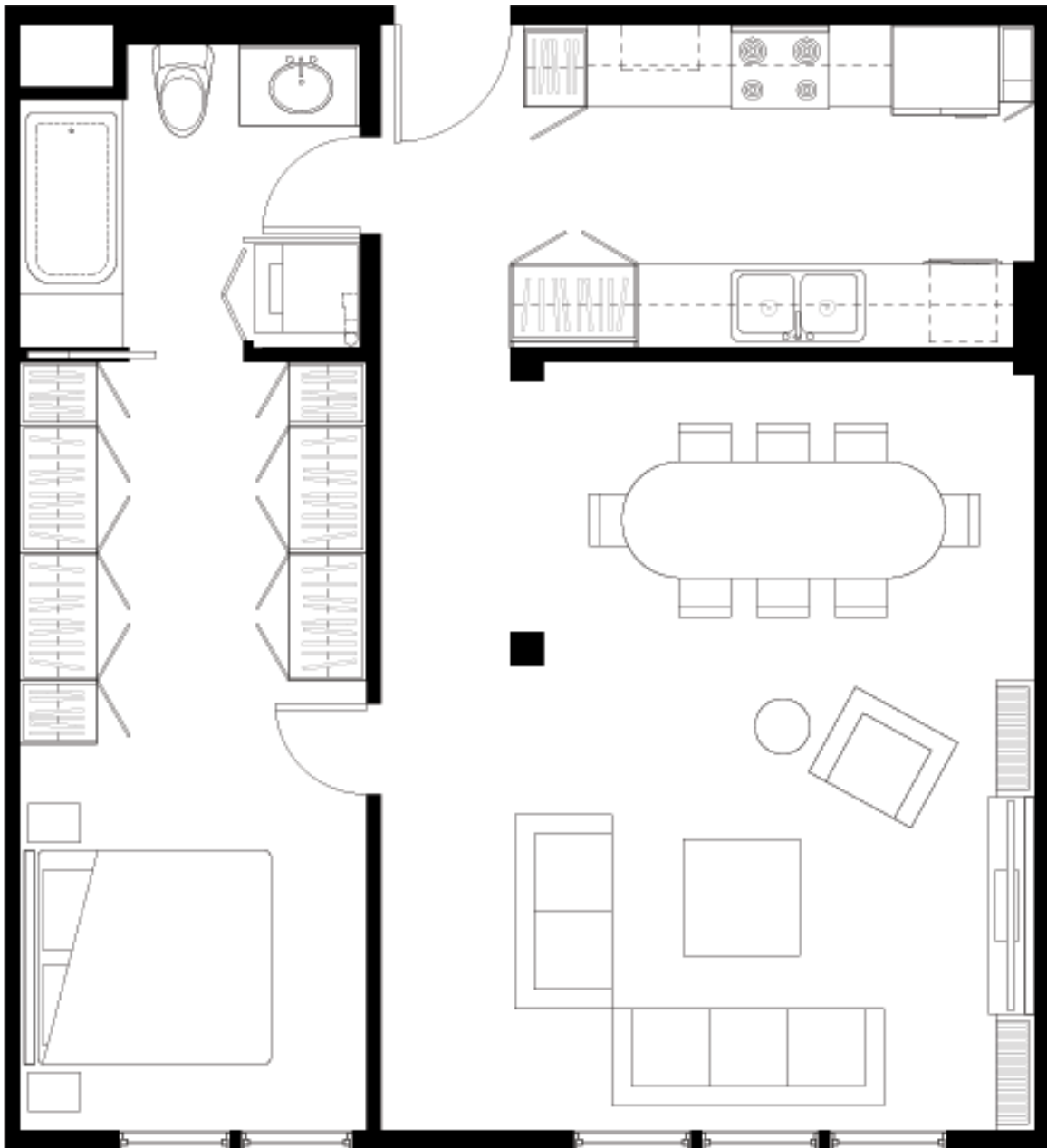
SOL202



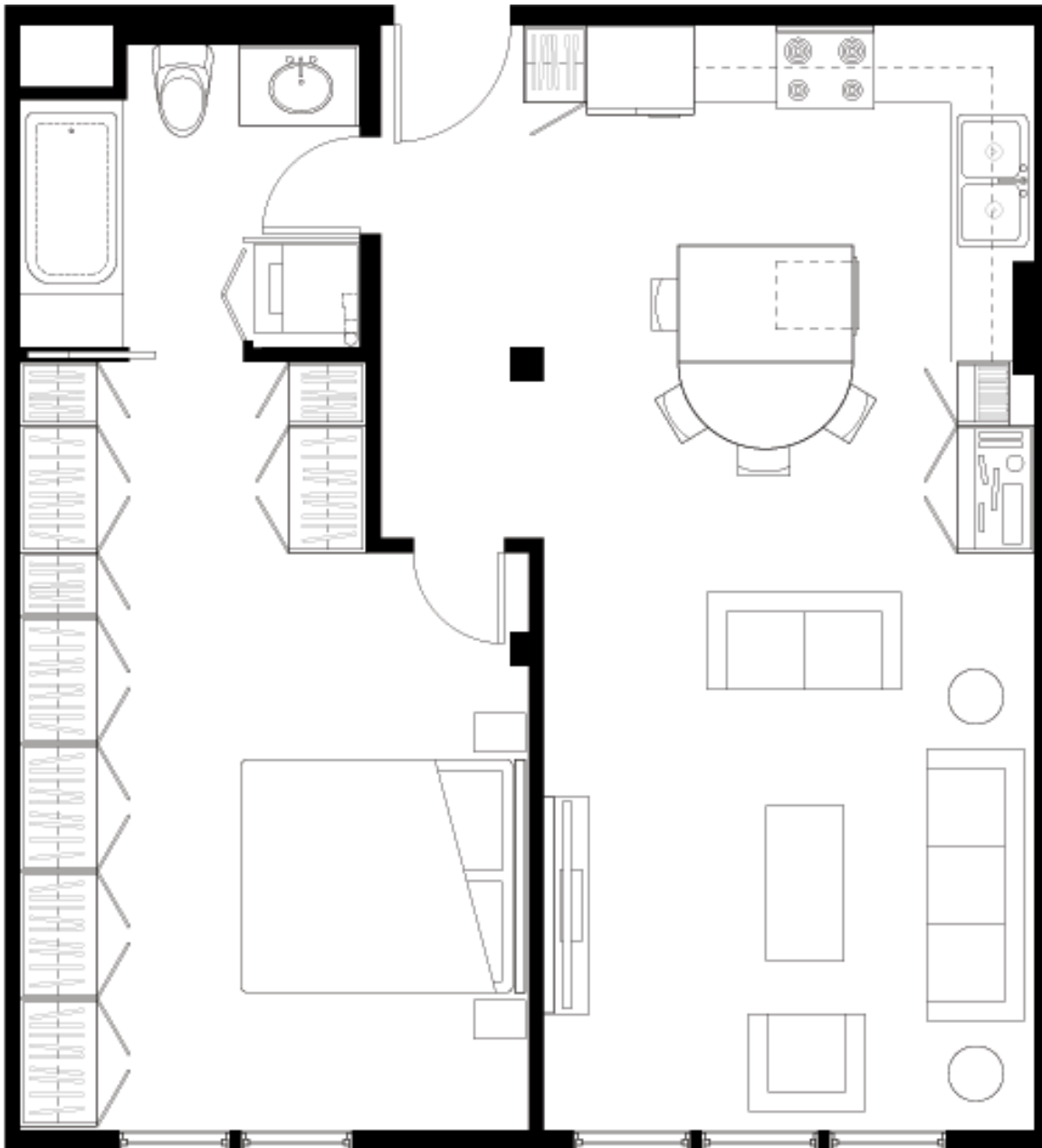
SOL212



SOL224



SOL228



SOL268

