Design Lab 1: In-Lab

Introductions and Definitions

To apply engineering principles to the ill-structured and open-ended design problems we face in real life, we must first translate the unmet needs that underlie these problems into a clear set of design specifications or requirements that can then be satisfied through engineering design and analysis. In this design lab, you will follow a process that takes you from real-world specifications to this set of system requirements. The design thinking taxonomy (Figure 1) shows us that divergence creates opportunities, while convergence supports decision making.

As a design engineer, your goal should be to create solutions to your client's problems. Remember that your client likely does not have access to the technical expertise or infrastructure required to create a solution themselves. If they did, they wouldn't need to hire you. However, **the client is an expert on the problem that they are experiencing and will always have the best understand their own needs.** The first step to addressing a client's needs is to understand what those needs are. Only then can you use your technical expertise to find a suitable solution. Effective designers can understand and interpret the needs of their clients (users and other stakeholders) and then use them to guide the creation of systems that address these needs without unnecessary features or development time. Throughout the design process, the requirements documentation serves as a roadmap, ensuring that the final design truly addresses the underlying need.

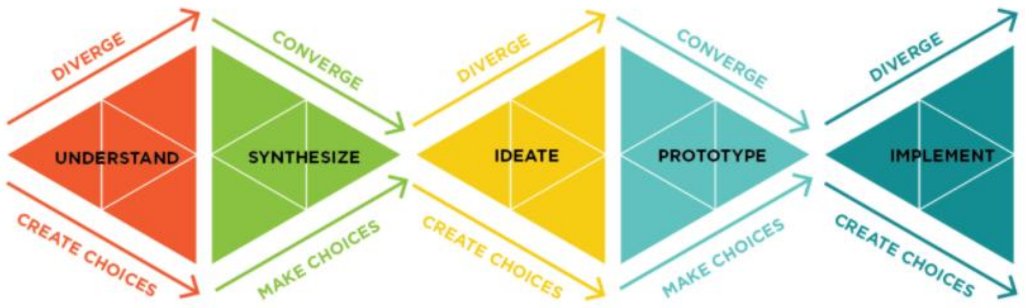

Figure 1. Design Thinking Taxonomy

This lab's objective is to develop system requirements using the user/stakeholder requirements developed in the pre-lab. The system requirements development process is as follows (Figure 2):

1. Identify users and stakeholders (from prelab)
2. Gather **User/Stakeholder Requirements** (from prelab):
   - Example: The mechanism should be durable.

3. Use **frameworks** to create an abstract model of the system to thoroughly explore the system. Ask, **"What are the constraints and capabilities of a system that can accomplish this requirement?"** (in-lab: frameworks)
    - Example: What must happen before this requirement can be satisfied? - Flow Chart
    - Example: What functions are necessary? - Functional decomposition
    - Example: What dependencies are there? - Block Diagram
4. Iteratively refine **system requirements** starting from this abstract model. There may be many system requirements that stem from a single user/stakeholder requirement.

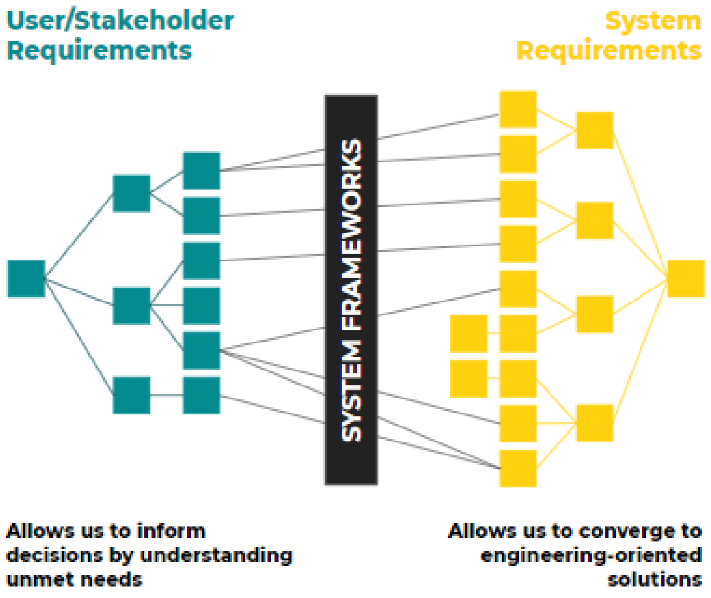
User/Stakeholder Requirements
System Requirements
SYSTEM FRAMEWORKS
Allows us to inform decisions by understanding unmet needs
Allows us to converge to engineering-oriented solutions

Figure 2. System requirements can be extracted from the system framework and informed by the user/stakeholder requirements.

Frameworks

After the user/stakeholder needs are well-described by the user requirements, it is necessary to apply your expertise to interpret the problem from an engineering perspective. This will inevitably require some decisions or assumptions about the solution, but these should be kept as abstract as possible. For example, the user may have a need for an efficient transportation system, and while both wheeled and flying systems may work, their engineering principles are vastly different, and it wouldn't be practical to consider the demands of both. While developing frameworks and system requirements, you'll often be forced to adopt one design approach or another. The key is to avoid narrowing focus too much (*e.g.,* wheeled vehicle vs. bike/car/ATV, etc.). Use your engineering expertise to consider how the system must be constrained depending on the limits of the project, but always base design decisions on the user/stakeholder requirements.

Frameworks help prevent oversights by ensuring each part of the system is represented and understood. Therefore, the type and layout of a framework must be decided by the designer to suit the demands of each project. However, a decomposition or block diagram is a good starting point to explore functionality. Non-functional requirements must also be considered, and frameworks should be developed as needed.

To get started:

1. Create a block describing the system in a general way (*e.g., a desktop toy automaton)*.
2. Subdivide this into its constituent elements.
3. For each of these, subdivide further.
4. Repeat until the system is fully represented.

Remember that at this point, we want to stay solution neutral. We want to avoid over constraining the design by specifying solutions. Including specific solutions **may prematurely limit design options**. Therefore, it is often helpful to decompose a system into its **functional components** to stay solution-neutral and avoid over constraining the design. However, great care should be taken to also consider the non-functional requirements. These are often related to the quality demanded of functional systems and should be mapped to the areas which are most impacted. For example, there may be an overall non-functional requirement of safety. This is more likely to impact the design of functional elements that require human interaction than those which do not. By considering where non-functional requirements are represented in our frameworks, we can more easily identify design opportunities to satisfy them. Note that there are other ways to develop a framework by using tools such as a block diagram or a flowchart.

System Requirements

Developing system requirements is an iterative and creative process. The intention is to define what a system must do without specifying how. This approach avoids overly constraining the design. The system requirements document serves as a guide throughout the design process. It should include a solution-neutral model of the system, which traces user requirements to features of the design. Table 1 shows a process that can help you develop system requirements:

A System Requirements Document should:

- Give a solution-neutral view of the system

- Allow exploration, trade-offs, prioritization as described in the user/stakeholder requirements, and allocation before committing to a design
- Demonstrate to users that their needs are reflected in the development process
- Provide a solid basis for the design
- Inform the testing and validation of the final system

Examples of early system requirements:

- The system should be rugged and not break during operation.
- The user should be required to apply no more than 5 pounds of force when operating the system.

Table 1. Procedures to develop system requirements

	Steps	**Example**
1.	Document user/stakeholder requirements	User's need is for human powered transportation.
2.	Identify constraints and capabilities of system that can accomplish each requirement	
3.	Create a framework or model (flow chart, block diagram, concept map) of a system that can meet the user requirements. Define what you must, but no more.	Wheeled system with no electronics
4.	Consider the influence of the system's environment/surroundings and define what is required in light of this.	Large elevation change - need mechanical advantage Long distances - reduce weight Bumpy road - vibration damping
5.	Consider the limitations and capabilities of the abstract system you've chosen and what is required for this to work	Adding mass will make it harder to start and stop Limited by the power output of the person Human factors for usability Efficiency of joints

6.	Consider the remaining faults by exploring how the system may be deliberately broken. Define what is required to ensure the system remains successful.	What if a wheel falls off? What if wheels lose traction?
7.	Iterate: Document emergent requirements in the system requirements document and update/modify frameworks and requirements as needed	

In-Lab Activities

During the lab, you will collaborate and discuss with your team and record your ideation. If you have one, bring your laptop to the lab. If you do not, work together on a computer with your teammates. There will be a total of three activities. **Make sure to save your work, as the outcome of the activities is a significant portion of your post-lab.**

We will begin this lab by meeting in our assigned teams. Your TA will provide you with a list of who your teammates are. Begin by introducing yourselves and take some time to get to know each other.

Part of your Pre-lab 3 assignment will be to complete a team contract (found on Canvas). While this assignment should be completed outside of lab, it may be worthwhile to take some time now to arrange a time and place for your team to meet and discuss the team contract together. When you work on the contract, take special care to address how and when your team will communicate, when you will meet outside of class, and if there are any penalties for failing to be a good member of the team.

Activity 1: Prelab Reflection

For the first activity of this lab session, complete the following tasks:

1. Begin by locating your prelab user point of view statement and user/stakeholder requirements.
2. Open the Example User/Stakeholder Requirements Document on Canvas.
3. A template is provided via google doc or can be found on Canvas.
4. Locate the provided template, and navigate to the slide “PRE-LAB REFLECTION 1”
    a. Copy your pre-lab’s point of view statement identifying a user or group into your team’s pre-lab reflection space
    b. Copy your pre-lab’s requirements in your team’s pre-lab reflection space.
5. Discuss your point of view statements and write your answers in the space provided:

   a. What similarities and differences are there among the users or groups and their needs that you have identified?

6. Discuss your requirements and write your answers in the space provided in "PRE-LAB REFLECTION 2":
   a. What requirements do you have in common?
   b. What requirements are different? Explain how.
   c. Are your teams' requirements verifiable, indivisible, traceable, and solution-neutral?
7. Work together to decide on a **single** user or group you will design your automaton project for in "PRE-LAB REFLECTION 3". It may be helpful to consider synthesizing, subselecting, or otherwise forming a consensus.

Activity 2: Frameworks

For the next part of the session, do the following:

1. In the provided template, navigate to the area designated "FRAMEWORKS"
2. As a team, create a functional decomposition framework.
   a. Start from your user/stakeholder requirements.
   b. Create a block that defines the overall goal of the system
   c. Subdivide this into the key functions or actions necessary to accomplish this goal.
   c. For each of these, subdivide further until the system is fully described.

Some signs that the framework is adequate:

- You can see how your user/stakeholder requirements would show up in the framework. For example, where they would be realized, or where they are most important. (traceability)
- The framework doesn't include any unnecessary suggestions or solutions which overly constrain the design (solution-neutrality).
- Your framework makes communication about the system and the requirements easier.
- The framework helps you explore the capabilities and constraints of the system.

Tips: Remember to stay solution-neutral by exploring what the system must do, but not how it will do it. Check that your user/stakeholder requirements can be represented on the framework (arrows and annotations encouraged!)

Activity 3: System Requirements

Use your existing framework as a starting point to develop system requirements. Follow the heuristic process and add requirements to the template as they become clear. Remember, this is an iterative process, and your requirements should be refined as the design becomes clearer. Some signs that the system requirements are adequate:

- Your requirements are organized in a clear way (e.g., by functionality or behavior)
- The requirements state what is wanted or needed without specifying a solution
- All relevant systems-level issues have been considered (e.g., PDS categories)
- The required system's behavior is clear
- The connection to user/stakeholder requirements is apparent and documented

Activity 4: Project summary

Create a 1 paragraph "Project Summary" that clearly communicates the target user or group you are designing for, the needs of that user group, and a few key requirements for your design.

Post-lab (30 Points will be awarded in Project 1 D2)

Your post-lab is a TEAM submission for Project 1 D2 (due the night before the next lab).

Ensure that your submission includes:

1. Prelab reflections parts 1-3
2. Frameworks
3. System requirements
4. Your 1 paragraph project summary

Make sure to export your filled in template as a PDF and submit it to Gradescope!