Australian National University

ENGN2225
Systems Engineering Design

Semester 1, 2016

Design Toolkit

Course Topics and Reading Guide

v2016.1

In this guide

This resource has been compiled to support learning in ENGN2225 Systems Engineering Design, and draws inspiration from a wide variety of sources. This document is supplemented by a detailed assessment guide and course outline

Chris.Browne@anu.edu.au

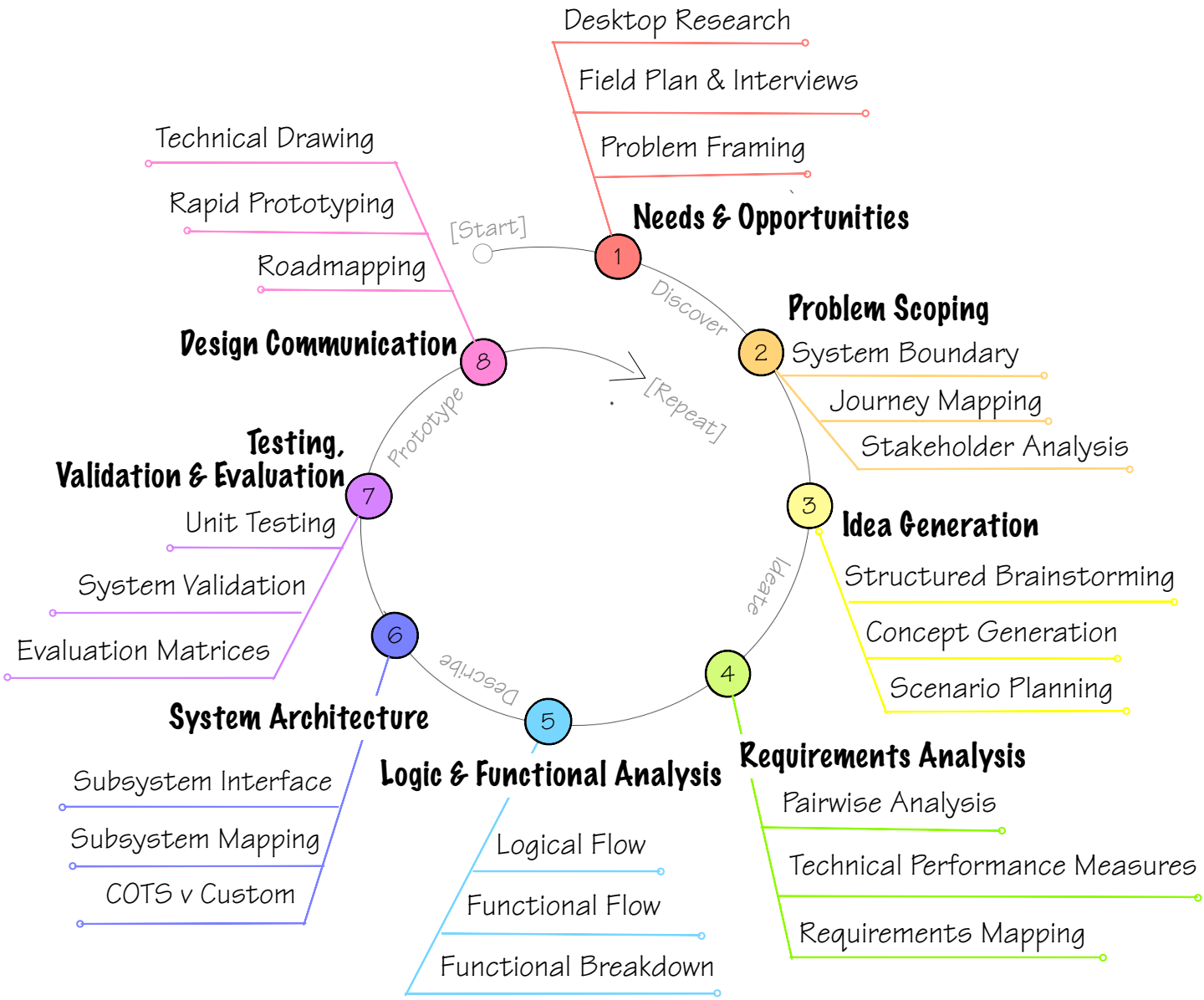
Note that ALL RESOURCES ARE IN THE RESOURCES DIRECTORY

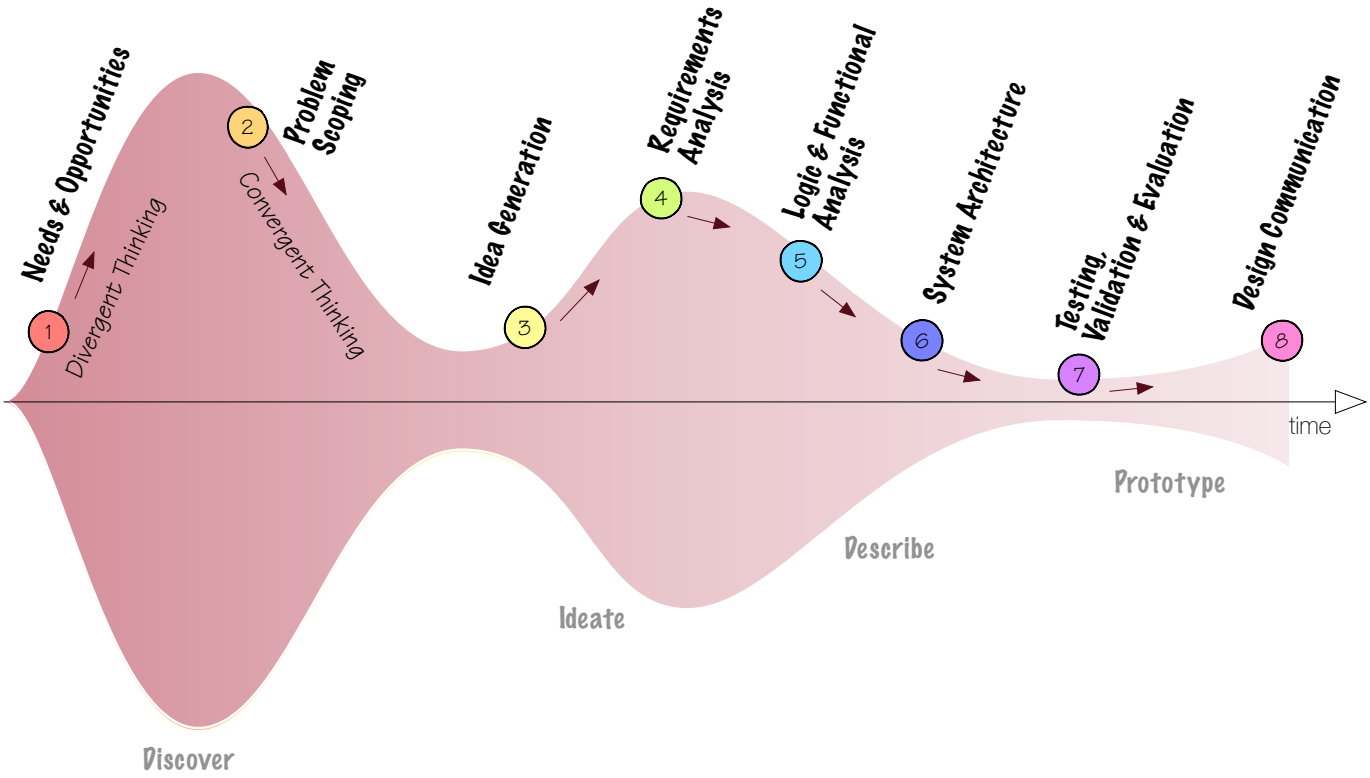
Also see the ONLINE CLASSROOM for videos on certain topics

ENGN2225 DESIGN PROCESS

The topics covered in ENGN2225 follow a spiral design process. In a spiral design process, as you move towards the centre you get closer to your eventual design.

Although formally in the course we cover the topics once, in a spiral process you should move around the topics multiple times and as quickly as possible – you might also jump around across different tools to get to where you need to go. This strategy allows you to advance your ideas faster than your competitors for the benefit of your clients.





Another way of thinking about your journey through the tools in ENGN2225 is assimilate two complementary design thinking frameworks - IDEO.org's divergent-convergent approaches, and the Double Diamond approach, where initially the divergent-convergent thinking is done to define the strategy, and then completed again to execute the solution.

See the Design Approaches lecture for even more strategies for executing design, including the traditional systems engineering Vee-model and the more competitive approach of OODA loops.

ADVICE TO THE STUDENT SYSTEMS ENGINEER

Systems engineering is both an art and a science,
and as a design science, it sits at the intersection of creativity and the natural sciences.
The ideas herein are to be nurtured and explored as you build up the model to describe your system.

However, the famous George Box quotation should be at the forefront of your thinking:
"*All models are wrong, but some are useful.*"

With that in mind, make sure your analysis is at least useful.

GREAT RESOURCES

There is no set textbook in this class. However, there are a number of great resources for investigating systems design:

- IDEO's human-centred design process (Online or download PDF - **highly recommended reading**)
- Service Design Tools has great collection of activities and ideas for design: http://www.servicedesigntools.org/tools/
- Stasinopoulos, P., Smith, M., Hargroves, K. and Desha, C., 2008. Whole System Design - An Integrated Approach to Sustainable Engineering, The Natural Edge Project, Earthscan, London. (Online resource)
- Systems Engineering Fundamentals, Department of Defence (Download PDF)
- MIT's Opencourseware subject on systems engineering (Online - check out the reading list too..)
- Hitchens, D.K., 2007, 'Systems Engineering - A 21st Century Systems Methodology', John Wiley & Sons, New Jersey.
- Blanchard, B.S., W.J. Fabrycky, Systems Engineering and Analysis, Fifth ed. Pearson, New Jersey, 2011.

TOPIC OVERVIEW

There are eight perspectives that we consider in Systems Engineering Design.

Needs and Opportunities (run by your tutor)

The best engineers don't solve problems, they find opportunities that build on the strengths of users. They do this through mapping the problem space, and taking the time to understand the needs of their users. In this topic we'll cover how to **discover** the needs and opportunities.

Problem Scoping

Now that you have explored the problem space, it's time to **empathise**; that is, to see the problem from your client's perspective. We'll use two tools that help you to understand other's perspectives: journey mapping and stakeholder analysis. Finally, we'll put the scope around the problem by defining the system boundary.

Idea Generation

Once you have empathised with your user and understand the scope of your opportunity, you're ready to start generating ideas about the opportunity. The tools in this week help your group **explore** the possibilities of design.

Requirements Analysis

Defining your requirements moves the project from exploratory into the specific. By defining the requirements, the specific parameters of the project become **measurable**. Defining the right customer requirements will ultimately push you towards a high-quality deliverable.

Logical and Functional Analysis

You can describe a system using many lenses. Here, you describe your design through logical and functional steps. A logical **breakdown** can be used to describe how a user interacts with the system to reach their goal, and a functional breakdown examines how functions can be grouped together.

System Architecture

Building on the functional analysis, we move into how the system components **interact** with each other. Designs can be highly modular or highly integrated, and there are positives and negatives for each approach. By describing the interactions between subsystems, the arrangement of the design becomes explicit.

Testing, Validation and Evaluation

Before moving to a complete design, testing needs to be carried out to ensure that each subsystem is **performing** correctly against the requirements. It's at this point you can comparatively evaluate your design against other options and the existing situation.

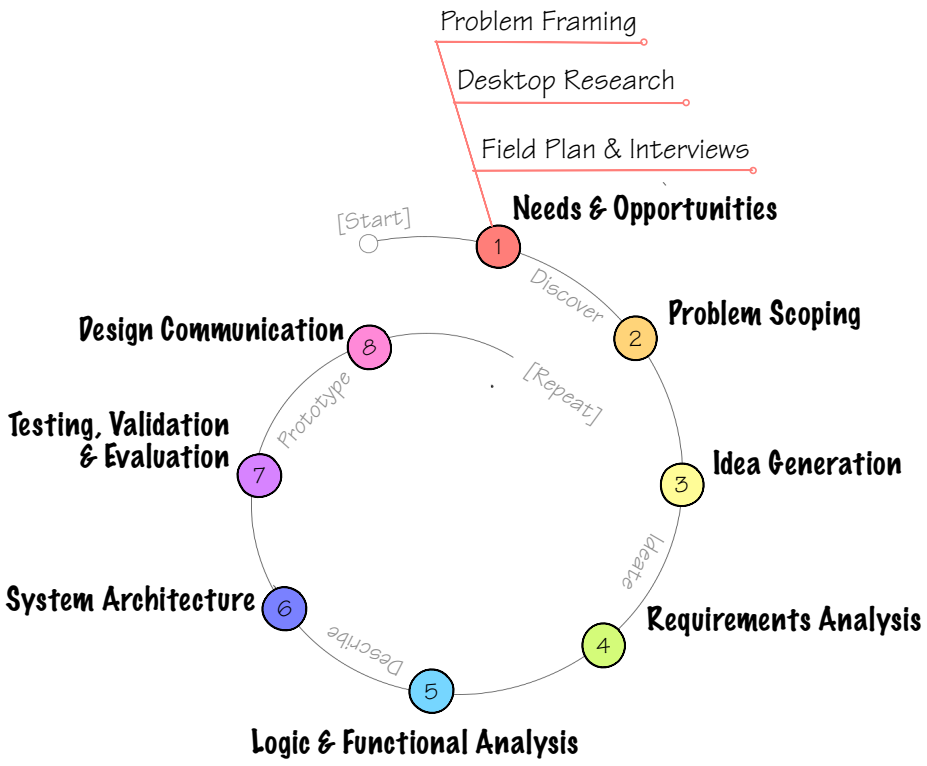
Design Communication (self-directed)

At the end of the day, there's no point creating a design if you can't communicate it - this could be through a physical or digital **prototype** of the design, or through other methods that communicate how the design works, and what opportunity the design meets.

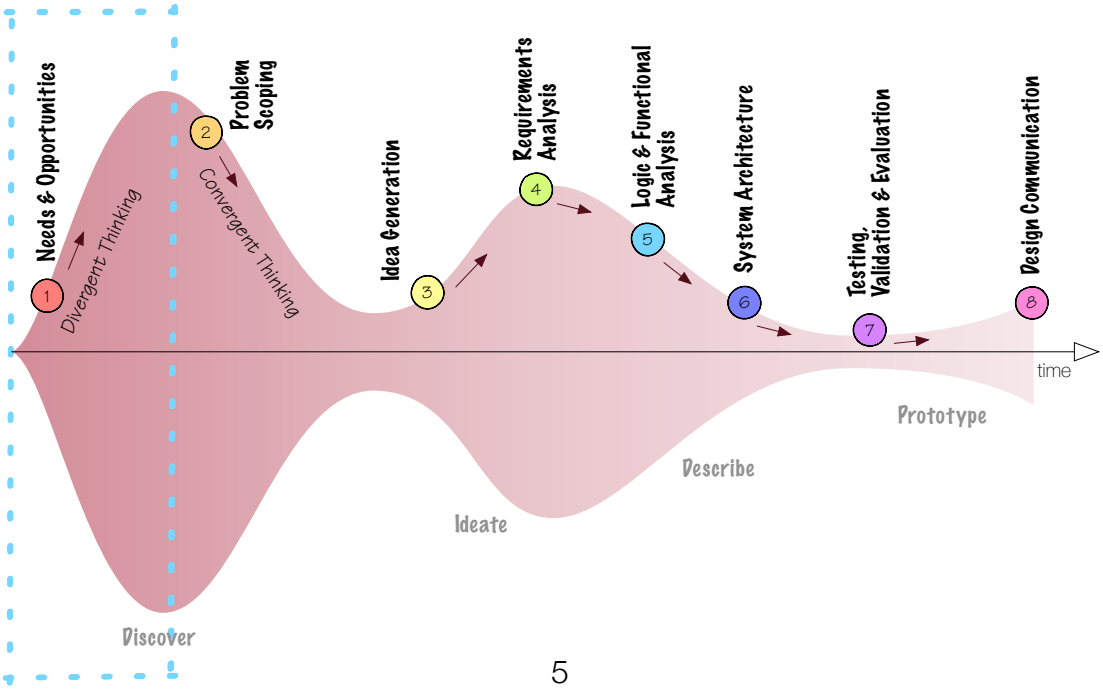
1. NEEDS AND OPPORTUNITIES

'Needs and opportunities' is the first step of our design journey. In this stage of the design cycle, we are in a 'Discover' mode, looking for opportunities that can build on the strengths of the user, client or community, and can also address their needs.

We will use three tools: Problem Framing, Desktop Research and Field Plan & Interviews.



At this stage of the process, your thinking should be diverging, ahead of refining your group's thinking in the Problem Scoping stage.



1.1 Problem Framing

Framing your problem correctly turns it into an opportunity

Framing your problem as an opportunity allows you to think in divergent ways. By putting a positive frame on your design thinking, you move from 'helping' your client to 'empowering' your user. A positive frame can inspire a virtuous cycle of good-will towards the project, and by looking for opportunities you are likely to come up with a better end result.

Example applications

Problem framing is used in a wide array of discipline areas, across the sciences and humanities. Framing a problem as an opportunity usually involves looking at the problem holistically. A classic example is the Great Horse Manure Crisis of 1894 in New York City, where the number of horses (and their excretions) were filling up the streets. The problem wasn't solved by looking at the horse, but the introduction of the automobile 'solved' this problem. Another example is the accessibility features of an iPhone, which have revolutionised the way that people with vision or hearing disabilities interact with technology.

Steps

Framing a problem is best done individually at first, then bring together all your group's ideas.

1. Write down your design problem in one sentence.
2. Turn your design problem into a How Might We.. statement
3. Generate some possible solutions to the How Might We… statement
4. What are some constraints in the given project context?
5. Rewrite your design problem [repeat as necessary!]

Reframing problems is seen all the time in politics. George Lakoff (personal website) describes how Conservative politicians in the US frame problems to their political advantage - taxes are vitally important in paying for the infrastructure needed, but it's hard not to support 'tax relief',

Hints

- A good How Might We… statement requires thinking broadly about your problem, and can be very lofty and difficult to achieve
- During the problem framing, you should encourage different ideas and creativity. Don't get bogged down in the detail, and avoid negative attitudes
- Build on the strengths of your user, rather than focussing on the negatives. See how you could transfer one skill to help them achieve the broader goal

Core resources

- IDEO's *Field Guide to Human-Centred Research,* p.31-33 on Problem Framing. [Online or PDF (framing sections only)]
- Tina Seelig's article in InGenius, *How reframing a problem unlocks innovation*. [Online or PDF]

Similar tools…

As the concepts behind problem framing are used widely, there are a few related tools worth considering alongside how you frame your problem. Problem setting (SpringerLink) is a way to influence the direction of a design approach, often used in policy and economics. Humanitarian, development and community practitioners rely on using a 'strength-based approach' (see Dr Graeme Stuart's reflection on this approach).

1.2 Desktop Research

You need to complete some desktop research about your problem before you try to solve it.

Primary research, such as going out and conducting interviews or user testing, is typically an expensive and time-consuming exercise. Desktop research can help you identify secondary sources that can point you in the right direction before going out into the field. In desktop research, your sources may include journal articles, conference papers, government and NGO reports and statistics, and other (less) reputable sources, such as newspapers.

Example applications

You will need to conduct desktop research for all the work in this course. For the TCs, you can generally refer to this guide and the core resources, as well as exploring resources related to your portfolio topic (eg autonomous braking). For the ETSs, you will need to research your topic and methodology a bit more widely, and in the Portfolio it will need to be considerably researched.

You may also choose to conduct some primary research to strengthen your argument, and this might include conducting your own testing of product for comparisons.

Steps

There are two broad goals of desktop research. The first is to discover your topic, and once you have learnt about it sufficiently, find some reputable sources to help you construct an argument (it's bad practice to build your argument first and then find sources - usually you'll miss the point).

As a very general guide, you might:

Discover

1. Consult this guide about potential methodology sources. Avoid using these sources in your bibliography in favour of demonstrating your own independent research
2. Search Wikipedia, a collaborative encyclopaedia, to find out about the topic. Read widely on Wikipedia to discover the topic, but do not cite Wikipedia in your bibliography
3. Explore the (reputable) sources referenced in Wikipedia - every article that meets Wikipedia's editing standards has a list of references. You can definitely use any reputable sources from here in your bibliography

Research

4. Once you know a bit more about the topic, explore the area on Google Scholar, or similar search engine. Use phrases you encounter in the *discover* steps. Many of these sources will be reputable.
5. Explore the ANU library journal databases. The ANU library has access to thousands of specialist journals, and these resources, used correctly, will influence your argument and help you to demonstrate scholarly research.

Hints

- Be skeptical during the desktop research phase. Don't believe everything you read.
- Use the desktop research as a way to understand more about your topic. Read 'around' topics not just 'in' them, as reading around will help you to see new aspects or perspectives
- Look for reputable sources. Reputable sources are typically peer reviewed and not supported by lobbyist funding. This is sometimes tricky to navigate, so err on the side of caution
- Be aware of bias in your sources, and seek alternative viewpoints (good research would reveal both sides of an argument.

Core resources

- University of Southern Queensland has a great list of example Harvard citations [Online or PDF]
- See the ANU Academic Skills and Learning Centre's guide to generic report structure [Online or PDF]
- CECS has a generic report structure too [PDF]
- There are generic templates for the ETSs available in many formats in the Resources directory (note you do not have to use these)

Library resources

- If you're unsure about Harvard referencing, and would like to know more by consulting a real book, the place to go is the *Style manual for authors, editors and printers* published by the Australian Government Publishing Service (ANU Library)

Similar tools…

Desktop research is also known as secondary research. Secondary research involves sources where the analysis has been done for you. In secondary research, the goal is to synthesise what has already been done. This should then prepare you for moving into the field to interview, or for conducting your own experiments.

Case studies

In your research, you will have to find sources to make your argument credible and convincing. These are examples of references from real ETSs in previous years.

This is an example of a poor choice of reference, as this information would be easily gathered from government and non-government sources, such as the United Nations:

> How much money does a person from Nepal Earn of make a day? *The Longest Way Home*, 2014 Available at: http://www.thelongestwayhome.com/blog/nepal/how-much-money-does-a-person-from-nepal-earn/, last accessed 15/03/2015

This is an example of not actually doing research. Avoid referencing the course material, as it shows your haven't actually done any research! Both are fairly abysmal references in the context of the course.

> - Systems Engineering Design-ENGN2225 Semester 1 2015 [course guide]
> - Browne, C 2013, ENGN2225 OC – Introduction, Video, Youtube, Australia

This is an example of not referencing the right resource. The Guide Dogs are probably a good source for basic information about using and accessing guide dogs; however, the author has referenced a website. On the same page is a PDF of the report with the same information - instead, reference this as a report.

> Guide Dogs NSW/ACT, 2015. Low Vision Services Paper. [Online] Available at: http://www.guidedogs.com.au/what-we-do/low-vision-services/low-vision-services-paper [Accessed 28 March 2015].

These are examples of the type of standard you should be aiming for in your research:

> - B. E. Johnson. The speed and accuracy of voice recognition software-assisted transcription versus the listen-and-type method: a research note. Technical report, Indiana University, 2011.
> - George Saon and Jen-Tzung Chien. Large-vocabulary continuous speech recognition systems: A look at some recent advances. IEEE Signal Processing Magazine, 2012

1.3 Field Plan and Interviews

Asking good questions will help you to see the problem differently

Before interviewing your client or 'entering the field' it's not always possible to sit down and conduct extensive desktop research. However, there should always be time to think of a field plan before starting the interview. Having a field plan can help you to get the most out of collecting primary data.

Example applications

A field plan is particularly useful when working in a team that requires a visit to a client or community (such as your group project). It's a chance to think through the logistics of the visit, and identify who in the team will take on which role, and what outcomes you hope to achieve. As it's likely that you'll only meet the clients once during the ENGN2225 project, it's worth planning ahead to make the most of this session.

Steps

A field plan for an interview is best constructed as a group.

1. Conduct desktop research ahead of meeting with your client. Understand the organisation that your client represents, and look for examples from other organisations in the field.
2. Check to find out how many people should come to the meeting, and then identify roles in your team. One person could take on the role of interviewer, note-taker, observer, introductions/thankyous etc).
3. Plan to start your interview by asking open-ended questions, not digging out specifics. When you start by asking specifics. Open-ended questions, or 'why' questions, can open up avenues of inquiry far richer than you could have planned.
4. If you have the opportunity, ask questions that allow the interviewee to describe a process, such as "how do you normally fix the water pump?" rather than assuming that your interviewee does it the same way you would. Even better, get them to show you.
5. Always make sure that you end the interview by saying thank you. If appropriate, a small, relevant gift might be appropriate (such as an ANU bookmark, etc).
6. Download your learnings shortly after the visit while the information is still fresh in your mind.

Hints

- Be conscious of the time that you spend with your client - make sure that they know how long you will have, and be mindful not to exceed this time
- Remember that an field trip or interview is a discovery learning trip, and so try to keep your mind as open as possible to all sorts of design opportunities
- Practice asking questions with friends and family.

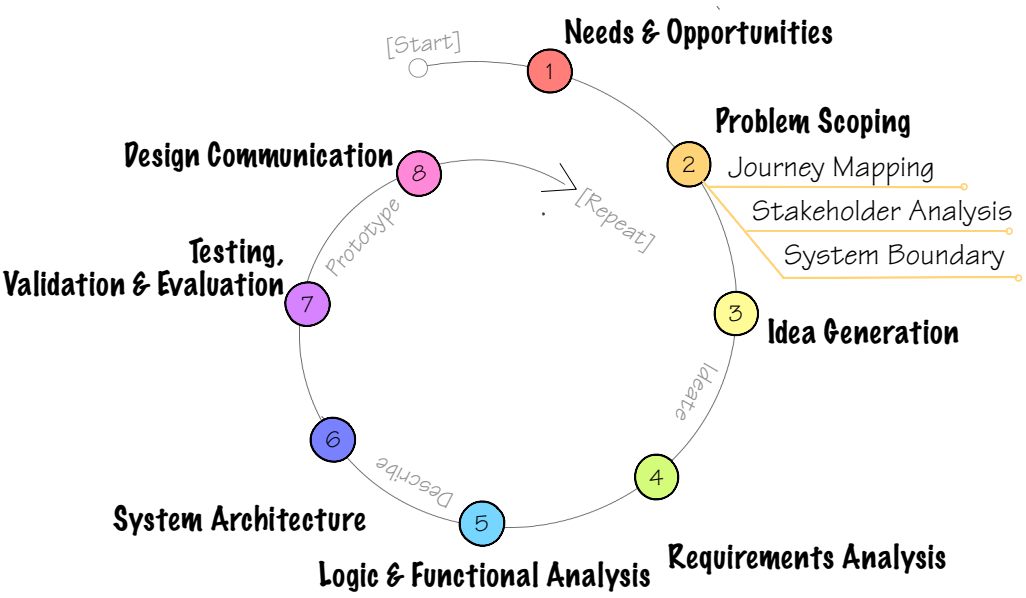
Core resources

- IDEO.org's *Field Guide to Human-Centred Design* has a great section on Individual, Group and Expert interviews. The plan above has been assimilated from this resource. [Online or section as PDF]
- Medecins Sans Frontiers, *A Guide to Using Qualitative Research Methodology* [Online or as PDF]. Particularly relevant is section 3 (pp11-21), which discusses interview techniques for generating data

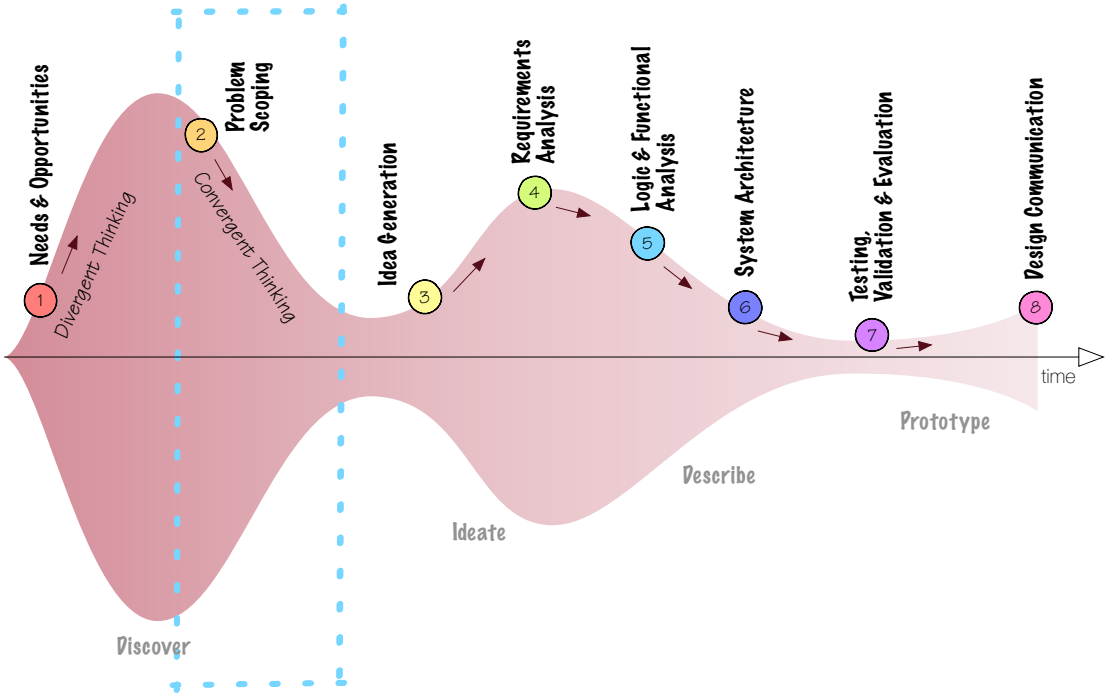
2. PROBLEM SCOPING

The tools in problem scoping assist you to further empathise with your client or users perspective. Problem scoping then allows you to start to put parameters around your project, and scope the boundary of your investigation.

We will use three tools: Journey Mapping, Stakeholder Analysis and System Boundary Mapping.



At this stage of the process, your thinking should be starting to converge, not entirely, to focus on the specific design opportunity you plan to address, ahead of the Idea Generation stage.



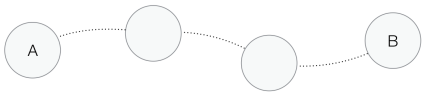
2.1 Journey mapping

A journey map helps you to understand how the user interacts with the system.

A journey map is a way of collecting stories about a process. The stories can be short or long, and can span minutes or decades. Important to the journey mapping process is understanding why people do things that way, not just what they do.

Example applications

Journey maps can take lots of forms, usually exposing an activity over time. A typical use of journey mapping might be a trip from A to B, such as driving to uni. A journey map could also be used to describe how a user interacts with a system, such as planning a uni timetable.



Steps

A journey map is best done individually before coming together to share how different stories overlap.

1. Decide the scope of your journey, or agree on the start and end points. For example, from A to B.
2. Individually fill out all the steps that are on the journey. You should be detailed at this stage
3. Once all the journey maps are completed, come together and use the individual maps to prompt a discussion
4. Synthesise all of the ideas around the journey, and agree on one journey map. This may need to be less detailed to account for all the different perspectives.

A journey map is also a great supplement to a formal or informal interview. You could ask questions to fill out your understanding of the interviewee's story, and then check the finished diagram with the interviewee before completing the exercise.

Hints

- It's important to collect all the details about a journey from all different perspectives. Try not to complete your first journey map as a group
- Complete a journey map with different roles (such as those identified in a stakeholder analysis). For example, getting to A to B, talk to a pedestrian, cyclist, car owner, bus driver, etc.
- During the journey mapping, you could wear different hats (see De Bono's six-hat thinking). One person could wear the Information (white) hat, and another could wear the Emotions (red) hat.
- After you've completed your journey map, look at how the stakeholders interact with the journey, or the different systems that the user interacts with.

Core resources

- Service Design Tools - Journey Mapping (Online or as PDF)
- Cockburn, A., 'Writing Effective Use Cases', Addison Wesley Introduction to Use Cases [Chapter 1, PDF 13 pages]

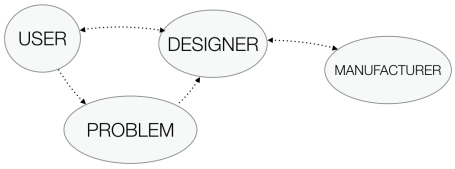
Similar tools…

A journey map is similar to a use case scenario (Wikipedia) or storyboarding (Wikipedia) . A journey map tends to be directed more at understanding why the user is doing something, whereas a use-case scenario tends to be more of a description of what the user does. Both are similar to flow or process diagrams, which we cover in a later topic.

2.2 Stakeholder analysis

A stakeholder analysis is used to understand how different actors/groups interact with the system

A stakeholder analysis is a way of making sure that all the people involved in a system, design or project are identified. Stakeholders could be directly involved with the system, or on the periphery.



Similar tools

There are many different tools that you could use to understand the way that stakeholders interact.

- a mud-map of the stakeholder relationships shows how different stakeholders are connected
- an influence-interest grid shows the balance between influence and interest between stakeholders in a 2-axis grid
- a connection circle maps out different stakeholders and the power relationship they have
- an organisational hierarchy shows the structure of an organisation, but is less useful at identifying how they are related around a problem

Example applications

An example situation of using a stakeholder analysis would be in a community consultation about a new engineering project. For example, the light rail project in Canberra has many different stakeholders with differing powers of influence, such as government, residents, commuters. These stakeholders can be broken down further - commuters could be cyclists, motorists, users of public transport, etc.

Steps

We will construct a stakeholder analysis using two tools - a mud-map and then a influence interest grid.

1. Construct a mud-map of stakeholder relations first by identifying the stakeholders at the core of the problem.
2. From this core, work out from the centre and connect all the stakeholders that influence or are influenced by that stakeholder using lines or arrows. Indicate the relationships using labels if needed.
3. Take a sensible number of stakeholders from this mud-map, and create a influence-interest grid (low to high on each axis)

Hints

- Invite people you don't think about immediately to be involved with the stakeholder analysis — you might find out they know a lot more about it than you
- Keep ideas broad. Remember this is a tool to understand relationships, and there is no one correct answer.
- Think about scenarios in which other stakeholders may be involved. For example, what would happen if the project was funded by different external stakeholders? How would that change the dynamics of the project?

Core resources

- Service Design Tools - Actors Map (Online or as PDF)
- Cockburn, A., 'Writing Effective Use Cases', Addison Wesley Introduction to Use Cases [Chapter 1, PDF 13 pages]

2.3 System boundary chart

Defining the boundaries clearly describes the scope of your current investigation into the system.

Now that you've had a look at the how your users interact with your problem, it's worth defining your scope. Constructing a system boundary chart is the first step in defining what's in and out of your system. The system boundary chart is flexible and changeable, but it explicitly defines how you consider the components, areas or influences of your system.

Example applications

This tool can be used as a brainstorming activity for many things. Typical uses include helping you to define your scope, or deciding what components are inside or outside your concern in a subsystem interface diagram.

Steps

1. Brainstorm all of the variables you can think of about the problem. Think widely, and add in everything you can think of.
2. Categorise all of the variables into one of the three aspects:
    - Internal (inside) - these are variables that are part of your system of interest, and you have direct (or mostly direct) control over them (for example, the components of the system)
    - External (outside) - these are variables that interact with your system of interest, but you do not have direct control over them (for example, the user interacting with the system)
    - Excluded (exogenous) - these are variables that might be important to consider at some stage, but for the time being are considered outside of the scope (for example, the weather)

For example, if you were describing the timetabling system at a university, you might brainstorm things like students, classes, timetabling staff, timetabling software, existing infrastructure, local admin staff, web browsers, etc.

Hints

- It's important to make sure that you revisit problem scoping throughout the design process to make sure that you haven't scoped the problem incorrectly
- The excluded list could be everything-in-the-world-that's-not-inside the system. Be judicious but comprehensive with your list.

Core resources

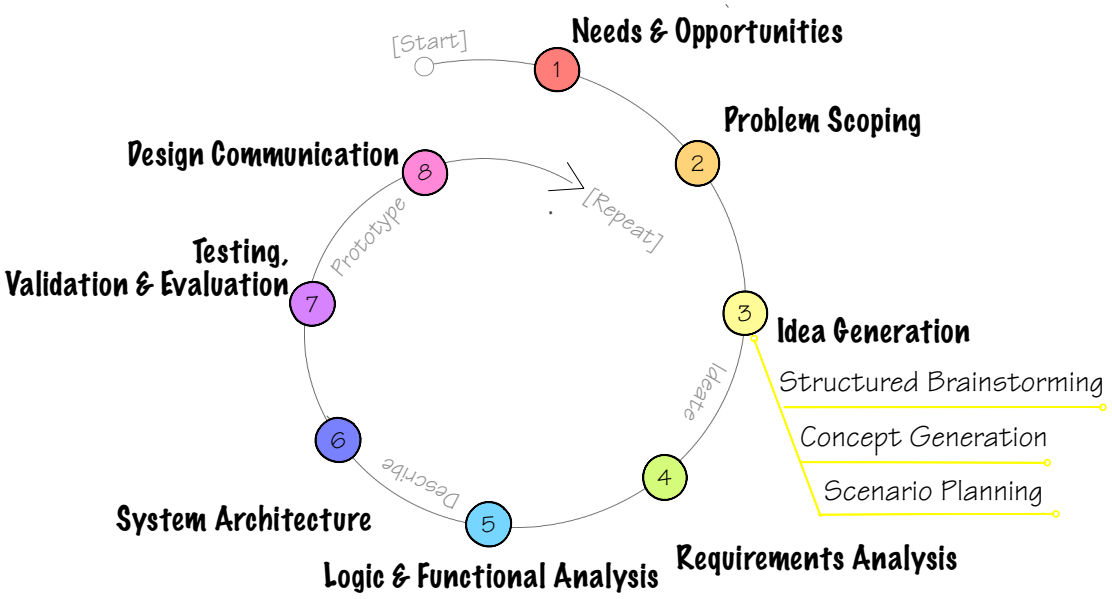
- Sterman, J.D., 2000, 'Business Dynamics - Systems Thinking and Modelling for a Complex World', McGraw Hill The model boundary chart (p97-98) [PDF, 6 pages]
- Define the System Boundaries A Practical Guide to Security Engineering and Information Assurance [PDF, note read especially p67-72]

Inside(/Internal)	Outside(/External)	Excluded(/Exogenous)
Components or actors that are inside your system or under your control	Components or actors that interact with your system but are not under your control	Components or actors that influence your system but are considered outside the scope at this stage

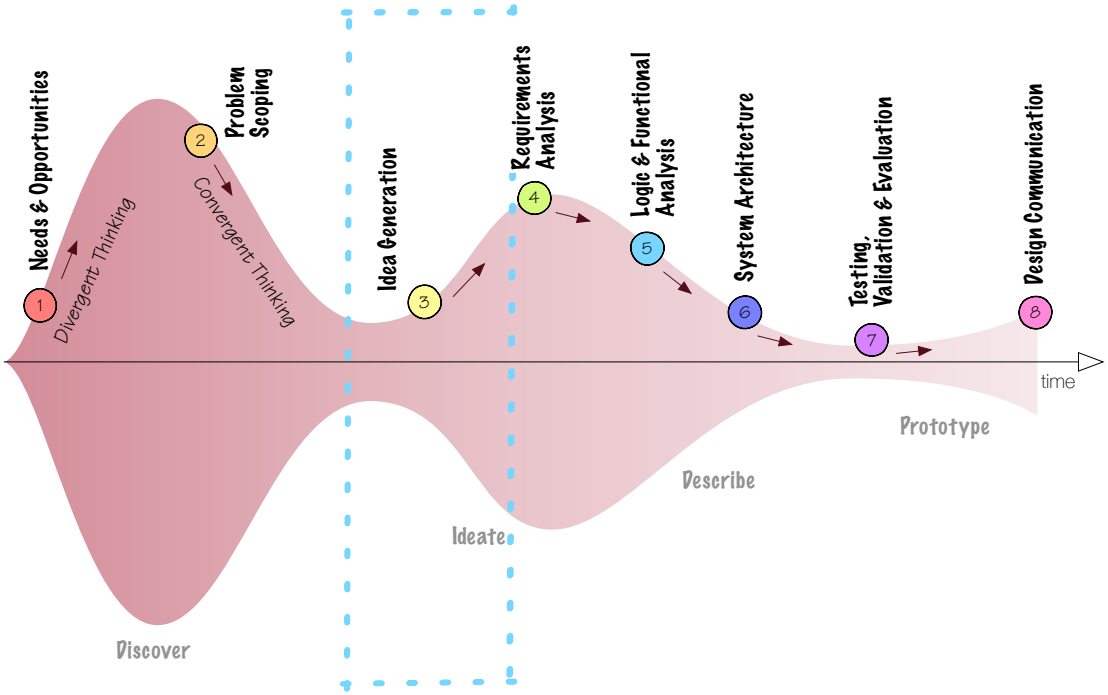
3. IDEA GENERATION

The tools in idea generation move you into divergent thinking again around the problem that you defined in your problem scoping. In idea generation, you are looking to build on opportunities to meet the problem scope based on the strengths of you user or client.

We will use three tools: structured brainstorming, concept generation and scenario planning.



At this stage of the process, your thinking should be allowed to diverge within the scope of the problem. Once you've explored the problem space, we move to the traditional systems engineering tools starting with Requirements Analysis.



3.1 Structured Brainstorming

Structured brainstorming allows you to brainstorm from different perspectives.

Brainstorming is a standard idea-generation technique. In a group setting, though, groupthink can set in, and truly innovative ideas can be suppressed or missed completely. In structured brainstorming, you are encouraged to take on different roles and remove constraints that might be limiting your thinking.

Example applications

Structured brainstorming can be used at any time during the design process, especially when you're trying to generate ideas. Brainstorming works best when ideas are open and everyone in the group is has agreed to the process.

Steps

The goal of brainstorming is to generate as many ideas as possible. Brainstorming individually, and then in a group, is a great way to make sure you come up with a bunch of different ideas.

1. Agree to the rules of brainstorming as a group. This might involve valuing creativity over reality, and encouraging broad ideas over detailed ideas.
2. Agree on a question or problem that you're brainstorming. Using a How Might We.. question (see Problem Framing is a handy way to frame a brainstorming session. Write down the statement so that it can be referred to during the brainstorming.
3. Assign each member of the team a different brainstorming role. There are two good ways to do this:
    - break up in stakeholders - assign each person in your team a different stakeholder perspective to brainstorm from. For example, user, designer, manufacturer, owner, etc.
    - break up by adding/removing constraints - assign each person in your team a different hypothetical constraint. For example, what could you do if you… had unlimited/no money; had unlimited/reduced technology capacity; had unlimited/reduced resources, etc.
4. Give a short amount of time for each team member writes down 5-10 ideas (agree on a number) from that perspective. Post-it notes are essential here, and different colours are handy to identify different stakeholders
5. Then, go around the group taking turns to map out ideas. Overlapping ideas can be joined together. See what comes out, and then discuss the possibilities.

Hints

- Once you've generated all your different ideas, you could try wearing De Bono's six thinking hats (see De Bono's six-hat thinking) to critique the different ideas
- Avoid taking brainstorming personally. The reason that taking different roles can work well is that it separates you from being personally responsible for the idea, as you are roleplaying that perspective
- If the brainstorming isn't getting your group anywhere, take a break. Go for a walk, or do something social, and come back to the idea generation fresh or explore a different question to brainstorm

Core resources

- Stanford's D.School note on effective brainstorming (Online or as PDF)
- IDEO.org's *Field Guide to Human-Centred Design* provides some good suggestions on general brainstorming. [Online or section as PDF]

3.2 Concept Generation

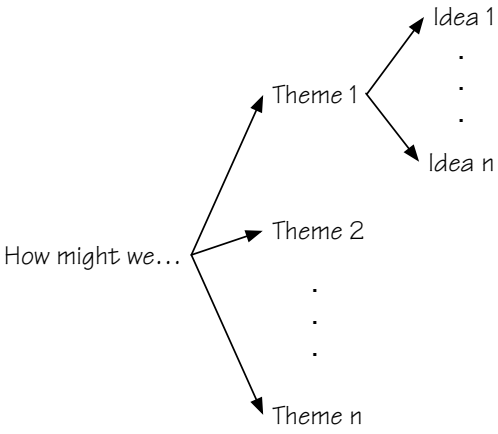
Concept generation is a systematic tool for generating and classifying ideas

A concept classification tree can be used to classify your group's ideas. It is often a good tool to complement the structured brainstorming step, and the systematic process can help to highlight more ideas. Concept generation sits in the ideation phase, as it is a structured tool to explore every single opportunity or idea.

Example applications

Ulrich and Eppinger (see core resources) spend their chapter explaining the different ways that you could hammer a nail. Now, this sounds like an extremely trivial activity, but when explored fully becomes quite a powerful tool for organising thoughts and generating ideas. For example, when classifying ideas to 'store or accept energy', they propose five ways (some are more feasible than others) to do this in a nail gun: chemical, pneumatic, hydraulic, electrical, and nuclear, with further sub-ideas under chemical and electrical.

A key output of a concept generation process should be the concept classification tree. Once this is described, a number of promising 'branches' should be identified to allow for more thorough exploration.



Steps

As concept generation can go hand-in-hand with structured brainstorming, the approach described here assumes that you've completed the structured brainstorming in §3.1 around a specific *How Might We…* question:

1. Group similar ideas as in step 5 of structured brainstorming (§3.1)
2. Look for common themes within or between the similar ideas. These themes will become the branches
3. Start to draw the concept classification tree. Start with the *How Might We…* question, and draw branches to the themes identified in step 2. If there are ideas that need to be broken down further, then create new branches until all ideas are exhausted.
4. Once all ideas are mapped, use the advice in Ulrich and Eppinger: prune less promising branches, identify independent approaches to the problem, expose inappropriate emphasis on certain branches, refine the problem decomposition on particular branches

Hints

- A single concept classification tree is useful, but to piece together a whole solution multiple classification trees are needed. Once you have explored the options around a number of different subsystems, you can easily piece these together, like in Exhibit 10 in the Ulrich and Eppinger reading (p.97)
- The classification tree is also a good way to break down ideas into functions and subsystem - a topic we'll examine in the Logic & Function analysis and System Architecture topics.

Core resources

- Ulrich, K.T., and S.D. Eppinger, 1995. Product Design and Development. McGraw-Hill. Chapter 5 - Concept Generation [PDF, 25 pages]. This is the ultimate guide to using a systems engineering approach to generating ideas. In particular, see p.93 for the concept classification tree.

3.3 Scenario Planning

Scenario planning helps you to imagine the boundary conditions of your ideas.

Scenario planning is a tool used to solve large, complex problems that concern a range of stakeholders, often with a range of perspectives. It is common a common tool used in organisational management and strategic policy making. We will use scenario planning to stress-test some ideas before we bed down the requirements of your system as a final way to ensure that you have adequately explored the problem space.

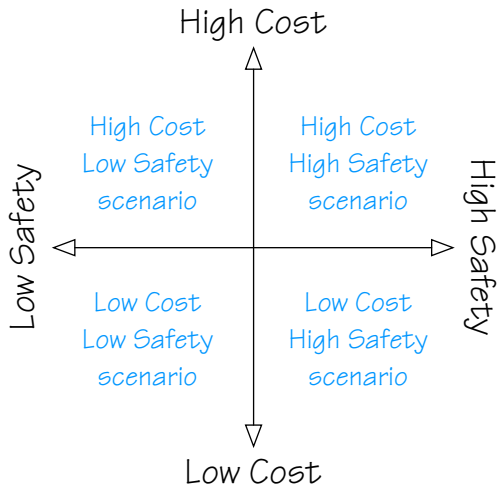
Example applications

The classic use of scenario planning was by Royal Dutch Shell, which used scenario planning to their strategic advantage in the 1970s to out-forecast other oil companies. They were able to react to the predicted overcapacity in the tanker business, and moved out of that area before the oil crisis.

Steps

Like all the tools we use in our interdisciplinary swamp of systems problem solving, constructing a good scenario plan is as much an art as it is a science. We are going to utilise only a part of the suite of tools and approaches used in scenario planning:

1. Consider two possible variables that are important to your client/project. Some examples could be be cost, safety, reliability, performance, etc. These need not be the two most important 'requirements' but rather should be things that you have found to be important to the success of the project. (Let's consider cost and safety here.)
2. Take the two variables, and place them on two perpendicular axes. Put low and high at either end (i.e. low cost to high cost, low safety to high safety).
3. Consider your main ideas from Concept Generation step. What would this idea look like in each of the four quadrants.
4. Question your ideas. Are any ideas more robust than others? Keep the ideas that still seem to look promising after this stage.



Note: in the next stage we're going to start to be specific about the design of our system. It's worth taking a number (three is good for ENGN2225) of independent ideas to the following stages.

Hints

- It's good to stress-test your ideas at this stage, but be prepared to change your thinking completely - we're still ideating, and the purpose of stress-testing your ideas is to have a broad sense of possibilities as you go into the requirements phase.
- Try a different set of axes to stress-test your idea further.

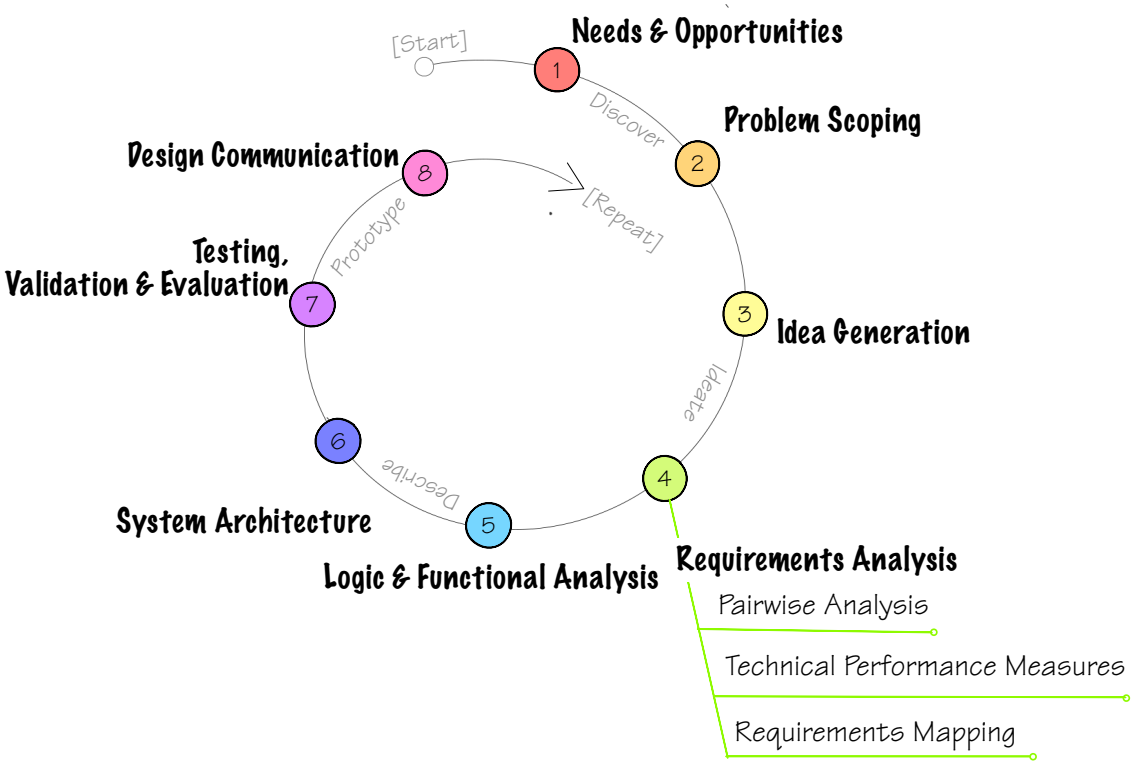
Core resources

- Wulf, T., Brands, C., and Meissner, P., 2010, *A Scenario-based Approach to Strategic Planning,* Center for Scenario Planning [PDF, specifically pp.8-16, alternative download]

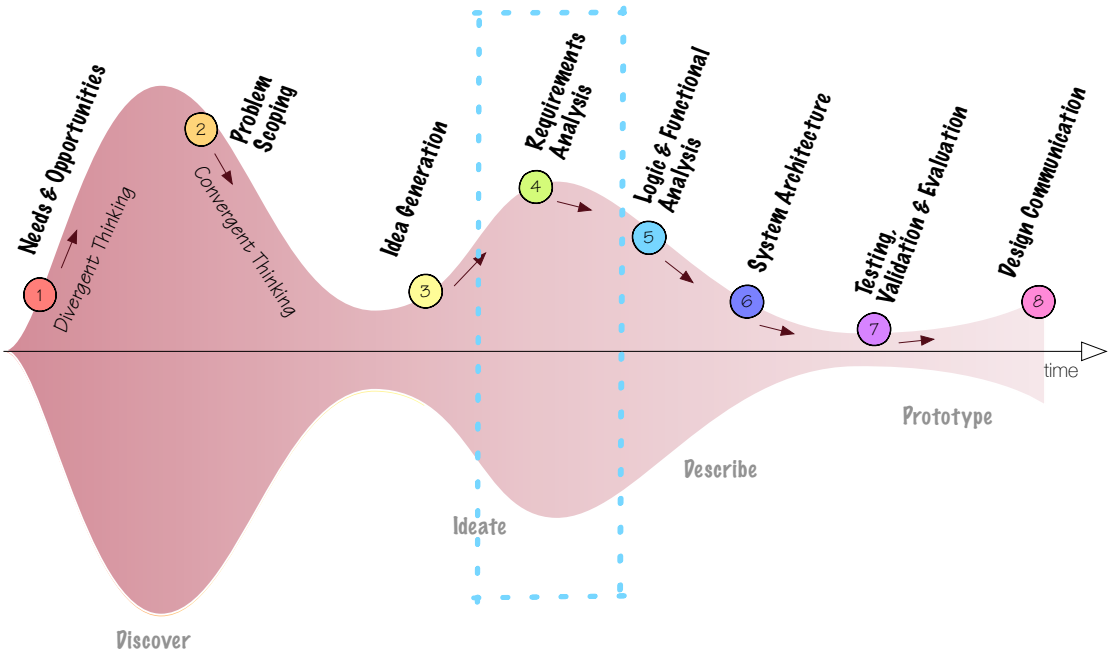
4. REQUIREMENTS ANALYSIS

By the time that you are ready to generate requirements, you should have a clear idea about the scope of the project and the problem that you plan to address. Coming into the requirements phase, it's a good idea to bring some ideas from the Idea Generation phase to contrast and compare.

We will use three tools: pairwise analysis, technical performance measures and requirements mapping.



At this stage of the process, you will start to define the scope of the rest of the investigation. Your requirements will become the reference point for the rest of the process.



4.1 Pairwise Analysis

A pairwise is a surefire tool for establishing design priorities.

The pair-wise analysis is a simple tool to rank competing requirements. Each requirement is tabulated for importance using a binary comparison. This will help your understanding of which requirements are more important than others.

Example applications

The pairwise technique is a useful technique for ranking all sorts of things, such as applicants for a job or entires in a design competition. Pairwise approaches are used to in probabilistic models to aid decision-making, and can be used to aggregate a range of opinions.

Customer requirements v Design requirements

It's important to note in the requirements phase the difference between customer and design requirements. Customer requirements tend to be 'wants', 'needs', or 'desires' from the client. Design requirements tend to be 'whats', or the measurable aspects of the design. At this stage, we are concerned with prioritising the customer's desires, and so will use the customer's perspective here.

	R1	R2	R3	R4	R5	Sum	Rank
Req. 1		1	0	0	1	2	3
Req. 2	0		0	0	0	0	5
Req. 3	1	1		1	1	4	1
Req. 4	1	1	0		1	3	2
Req. 5	0	1	0	0		1	4

Steps

Construct a list of potential customer requirements (for this exercise, less than 5 becomes trivial, and more than 10 becomes unmanageable) to bring into the activity.

1. Create a table with the customer requirements listed in the rows and columns in the same order
2. Black out the diagonal.
3. For each row, compare the requirement in the row to the requirement in the column.
    - award a 1 if the row requirement is more important than the column requirement
    - award a 0 if the row requirement is less important than the column requirement
    - in this process, rows and columns cannot be equal
4. Tally up the rows and calculate the sum. The most important customer requirement will have the highest sum.
5. If you have equal rankings, split them by comparing them to each other

Hints

- If you have equal ratings, make sure you split them - the purpose of the tool is to rank requirements
- Complete your pairwise individually, and combine them using the Borda count technique in Dym reading on rank ordering

Core resources

- Dym, C.L., P. Little, E.J. Orwin and R.E. Spjut, 2009. Engineering Design - A Project-Based Introduction, Third ed., John Wiley & Sons. pages 60-62 provide an example of a pairwise using a ladder as an example [PDF, 3 pages]
- Dym, C.L., et al. Rank ordering engineering designs in Research in Engineering Design November 2002, Volume 13, Issue 4, pp 236-242 [PDF, 7 pages] Note that this also introduces a more complicated idea of group aggregation

4.2 Technical Performance Measures

Using TPMs we take the customer wants, and determine the measurable design requirements

Once your technical performance measures (TPMs) have been established, they becomes the key to understanding whether or not our design meets or exceeds the requirements, and competing products. Setting the TPMs or 'metrics' allows us to compare designs against a common scale. It's important to make sure that your technical performance measures are limited to the key needs of your design.

Example applications

Finding the right metrics to judge your performance is key to a successful project. It is also important to explore exactly what the metric means. For example, if your client wants you to design a 'fast' car, does that mean more engine power (Watts), more torque (Newton-metres), top engine speed (RPMs), lightweight (kg) or reduction of the drag co-efficient. Exploring which TPMs you choose to measure your project will completely change the direction of your optimisation.

Steps

Take your customer requirements, and convert them into a manageable number of design requirements:

1. Design requirements should be neutral and measurable:
   - CR: faster car; DR: speed, engine torque, mass
   - CH: cheap; DR: cost of materials, fuel consumption, ongoing cost, capital cost, etc
2. With your design requirements, establish the metrics that each DR can be measured against (each DR could probably be measured at least 5 ways)
3. Establish whether the desired direction of the TPM is to increase, decrease or optimise the metric value, and set any minimum/maximum requirement or benchmark.
4. Take your expanded list of metrics, and decide on a subset of TPMs to measure your project again (at least one per requirement is a good guide)

Hints

- Listing all the ways that a customer requirement could be measured will help you to be explicit about the project's requirements
- Explore the TPMs with your client - get them to help you identify exactly what they mean.

Core resources

- Blanchard, B.S., W.J. Fabrycky, Systems Engineering and Analysis, Fifth ed. Pearson, New Jersey, 2011. Chapter 3.6, pp96-100 [PDF, 5 pages]
  Glossary for that reading: MTBF is Mean Time Before Failure, MTBM is Mean Time Before Maintenance, LCC is Life-Cycle Cost.
- INCOSE Technical Measurement Guide [PDF, 65 pages, read particularly 3.2.3 (pp10-11) and Figure 3.3 (p16) for examples]

Customer Requirement	Design Requirement	Metric
Fast car	Engine power	Horsepower/Watts
	Engine torque (acceleration)	Newton-metres
	Engine speed (top speed)	RPM

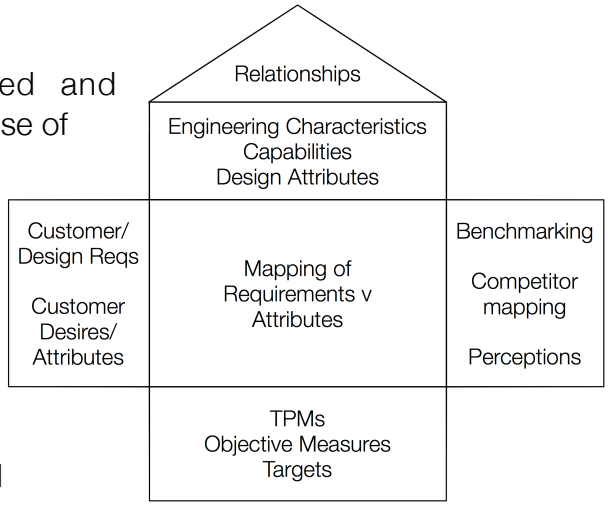
4.3 Requirements Mapping

Requirements mapping establishes the relationships between requirements and attributes of the system

Requirements mapping is an approach for organising the design priorities, trade-offs and benchmarking. The House of Quality is a way of organising and comparing the design requirements and the TPMs. By going through this process, we begin to see the important parts of the design and - more importantly - the relationships between the parts.

Glossary

The vocabulary around requirements is wide, varied and interchangeable. There is no right or wrong usage, as each use of language is likely to be slicing the system in different ways. See an example of the different vocabulary on the right.



Steps

Building a HoQ is an iterative process, and will keep evolving. It's a useful tool to track changes in a project, and good version control is important.

- list the design requirements an the left of the table, and the design attributes along the top of the table
- map the relationship between the design requirements and the design attributes (0, 1, 3, 9) in the centre of the house
- indicate the relationships between design attributes at the top of the house
- indicate the TPMs, direction of improvement and targets at the base of the house
- benchmark competitor products on the right of the house
- revise the mappings continuously as the project develops and the requirements change

Note on standards
When you're mapping the relationships in the middle of the HoQ, please use numbers rather than symbols - 1: Weak, 3: Medium, 9: Strong.
In the relationships mapping in the 'roof', please use + positive and - negative symbols.

Hints

At the core of the mapping is a relationship between two aspects of the problem. Some ideas are:

- customer requirements v design requirements/attributes
- design requirements v design attributes / subsystems / work teams
- staff/team members v capabilities/professional attributes
- pizzas/cakes v toppings/ingredients

Find what are the most useful aspects to map for your problem.

Core resources

- Hauser, J.R. and D. Clausing, The House of Quality in Harvard Business Review, May 1988. The House of Quality) free reprint [PDF, 13 pages]

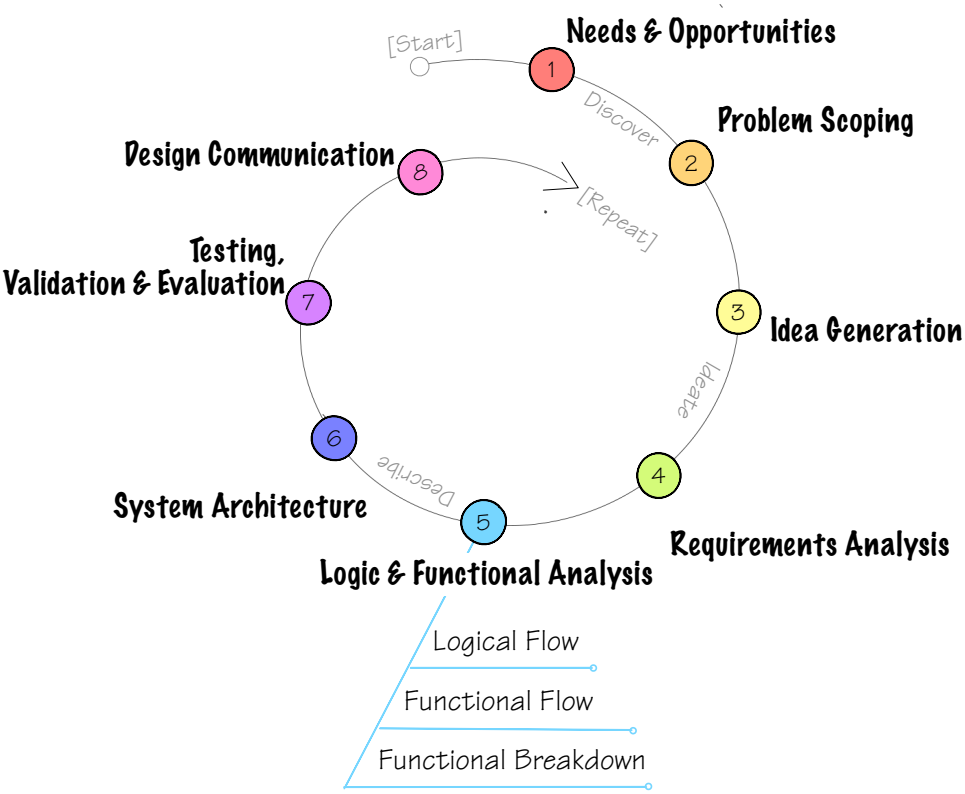
Useful resources

- QFD - HoQ Tutorial by Dr A Lowe [Flash file]
- Quality Function Deployment (QFD) and House of Quality - MIT - goes through the process step-by-step, by Dr. Michael Short [video, 44m]

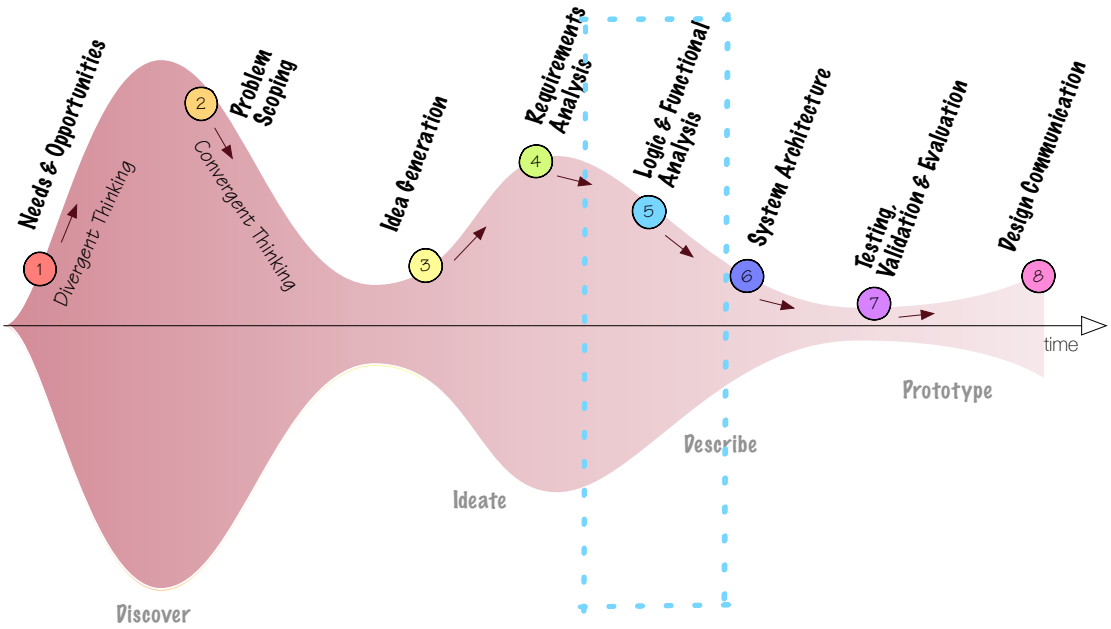
5. LOGIC AND FUNCTIONAL ANALYSIS

A system (remember: something a system has components, relationships and goals) can be broken down in many ways. We're going to break down our system by logic and function, before examining the subsystems in the System Architecture topic.

We will use three tools: logical flow diagrams, functional flow diagrams, and a functional breakdown.



At this stage of the process, you are starting to describe the way that your system or design will work. By describing how your system should work, this sets the scene to abstract it into subsystems.



5.1 Logical Flow

A tool for showing how the decision-making process around a system or design works

A logical flow diagram maps the decision-making steps of a system, supplementing the functional analysis. The FFBDs shown in §5.2 map how the system works functionally; a logical flowchart helps to identify where decisions are made and what inputs are required.

Example applications

Logical flow diagrams are essentially decision trees: do this step, if A happens do B, if C happens do D, else E. Logical flow diagrams are commonly used to show business processes, such as processing an online order, or mapping out a call centre complaints procedure. A flowchart can be broken into users, hardware, resources or systems to highlight the roles and actors in a process (see p108 of Hurt, 2012).

Similar to…

The logical flow and functional flows are very similar, but focus on two different things. A logical flow emphasises the decisions and resulting actions of a process, whereas a functional flow describes a manual-style procedure that should be followed.

Steps

A good logical flowchart should help the user understand the logical progression of the required steps to follow in a process. A comprehensive flow chart will have all of the contingencies clearly covered.

- Establish the start and finish points of the process that you're mapping. Scope the system boundary to a useful level.
- Determine column headings. These could be systems, users that interact with the system, or different plant/machines that are required.
- List the required actions for each column, and where the interactions between columns occur
- Seek feedback, discuss and revise

Hints

- you should be able to read a flowchart from left-to-right, top-to-bottom. Give your flowchart a title, and don't cram everything in. Position the flowchart at a useful level within the system - too detailed and it will become too difficult to use, and too high-level and it will become trivial
- organising your flowchart in columns by responsibility or resource can help to identify issues in your processes
- get feedback on preliminary flowcharts from stakeholders before delving into detail

Core resources

- Hurt, R., 2012, *Accounting Information Systems,* Chapter 6 [PDF]
- The Ultimate Flowchart Guide - on Creately, and informal but informative source [Website]
- Classic engineering case study: WD40 or Tape [Image]

Software

Choosing the right software for flowcharts and other diagrams can make a big visual impact. Gliffy is popular online service, and Omnigraffle is my favourite on a Mac. Visio is the Windows-standard. Avoid using diagramming tools in a word processor - they almost always look second-rate.

5.2 Functional Flow (FFBDs)

A tool for showing the functions of a system, step-by-step.

A functional analysis looks at the 'functional' steps that are required to use or interact with a system. We will use the traditional technique Functional Flow-Block Diagrams (FFBD) for our functional analysis. FFBDs are useful to the design process in as much as they describe the intended use of the system. FFBDs can be hierarchical, and an important consideration is to keep the steps in an FFBD at useful task sizes. Some blocks will have sub-functions, which can then be broken down into a sub-level.

Example applications

FFBDs are used in large projects to highlight both the procedure of functions within a system. An Operational FFBD could be constructed from any step-by-step procedure that you can lay your hands on: baking a cake, putting together IKEA furniture, fixing a car, or sewing a superhero outfit from a pattern. A maintenance FFBD can be constructed from any troubleshooting guide: for example, fixing a phone, repairing a tyre, or tuning a piano.

Note on standards
There are also many ways to represent Functional Flow Block Diagrams (FFBDs). Please use Systems Engineering Fundamentals reading as a guiding standard.

Similar to…

A journey map or use case diagram. There is a big difference, though. A journey map is a discovery tool for empathising with your user, whereas an FFBD is used to specifically demonstrate the intended procedure for a system. For those who are coders, an FFBD follows similar rules to a procedural programming language: one process must be completed before moving to the next process.

Steps

- Create a bullet list of top-level steps from the start to finish of a procedure. Try to group the steps in the list approximately the same size
- Break the top-level list into more detail by filling in all the second- and third-level steps that are required to use the system
- Look for any maintenance steps. These are steps that require a logical 'check', where requiring a check means that a maintenance subroutine will be followed
- Map all of the steps out using the diagramming conventions.

Hints

- Listing the steps in a bullet list can save you heaps of time before moving to a diagram (much easier to change and move around).
- Always draw your diagram left-to-right, use orthogonal lines, and partition the levels clearly
- Once you have created an FFBD, do some user testing and see if the user does exactly what you expect using the steps.

Core resources

- Systems Engineering Fundamentals, Department of Defense Chapter 5 and Supplement 5A [PDF, 7 pages]
- Examples of FFBDs from Blanchard, B.S. and W.J. Fabrycky, Systems Engineering and Analysis, Fifth ed. Pearson, New Jersey, 2011. [PDF, 8 pages]

5.3 Functional Allocation

Functional allocation organises your system into functional groups

Using a functional allocation, we take the key functions identified in the Functional Flow, and aggregates them into a hierarchical tree. The purpose of this process is to ensure that all the required functions of a system are mapped into the subsystems.

Similar tools

Hierarchical breakdowns of system are a common method used in a 'reductionist' approach. In this topic we break the functions into the likely subsystems, but there are many other ways that a system could be broken down. For example, a Work Breakdown Structure is a hierarchical map of allocating work between departments or team members.

Steps

Start with the key functions from your FFBD.

- Group the key functions into similar functional groups. Try to make each group as self-contained as possible: this promotes modularity and reduces interdependence of a system
- Look for themes around functional groups, and assign them to an actor, resource or subsystem; for example, electrical system, mechanical system, or control system.
- These units become the building blocks for the functional packaging of your design, and ultimately shape the characteristics of the relationships in your system.

Hints

- Ensure as much as possible that the functions within a group are self-sufficient, requiring as little interaction with other systems as possible. There are, however, trade-offs. For example, a mechanical car lock will operate independently, whereas central locking relies on the electrical system. If your battery fails, then you can be locked out of your car, which can be a good thing if you are trying to keep people out, or a bad thing if you're trying to get in.
- If you're considering functional allocation in a work group, there may need to be high duplication in order to make a unit self sufficient. Take, for example, a company with global offices. It might make sense to house all of the IT support in one location, but it might reduce the responsiveness to IT-related problems, driving down productivity. Instead, it might make more sense to distribute the IT support between offices so that each office has the capacity to function. In reality, it might be a combination of centralised and distributed support.

Core resources

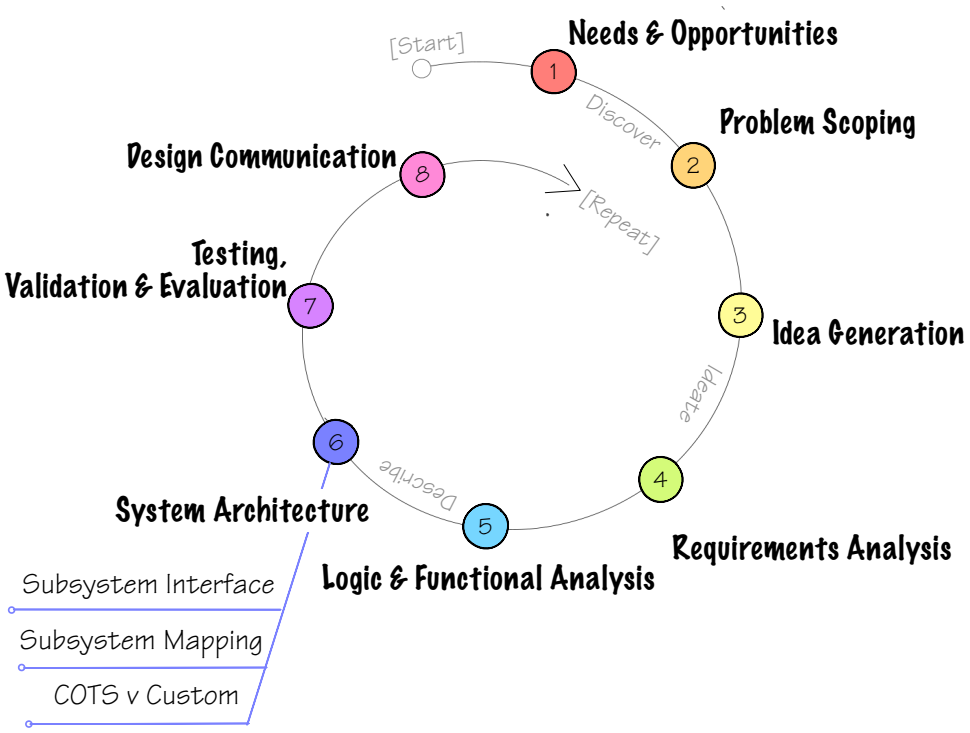
Blanchard, B.S. and W.J. Fabrycky, Systems Engineering and Analysis, Fifth ed. Pearson, New Jersey, 2011

- *Functional Allocation* - organise the functions from the FFBDs into hierarchical groupings [PDF]
- *Functional Packaging* - organise the functions into functional units [PDF]

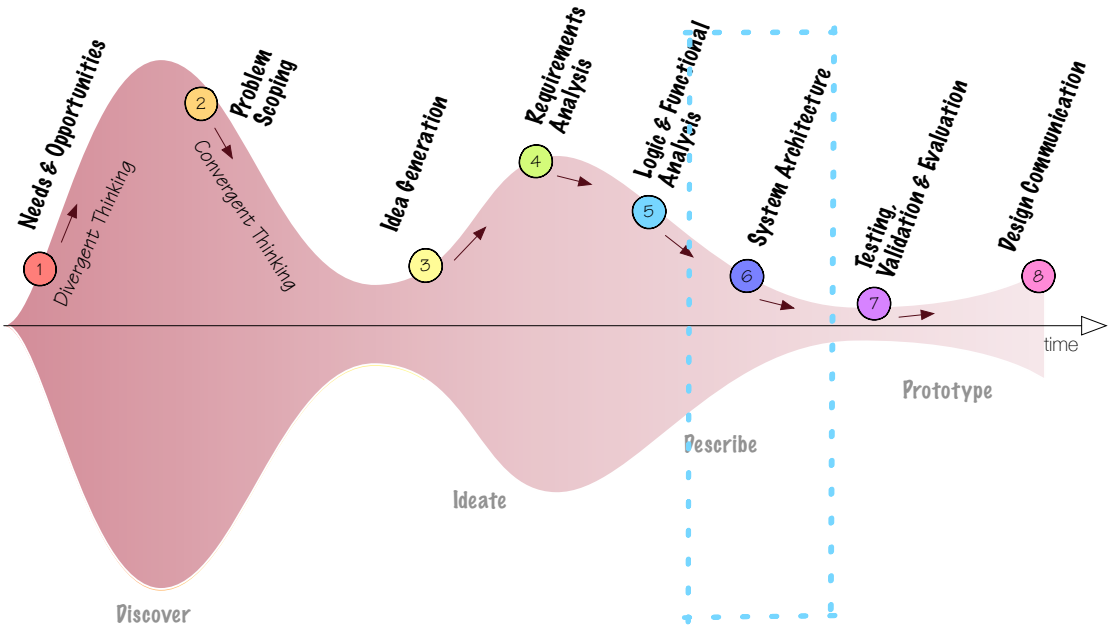
6. SYSTEM ARCHITECTURE

In the system architecture topic, we define the physical subsystems of your design in a system down to the component level, and describe the interactions between subsystems and components.

We will use two tools: a subsystem interface diagram, and a subsystem mapping back to the original requirements. The COTS (commercial-off-the-shelf) and Custom topic looks at the trade-off between utilising existing modules that are readily available, or developing your own modules.



At this stage of the process, you are narrowing your design down further. You have probably favoured one design from your idea generation, but it's good to take at least three alternatives through to the Testing phase.



6.1 Subsystem Interface

An approach for demonstrating the inputs and outputs of the system and subsystems.

In this topic, we will define the boundary of your design, the subsystems, and the relationships and interactions between them. A subsystem interface can be used as a control tool to manage how different subsystems interact, and what signals the modules can expect to send and receive.

In some cases, it may be easier to complete the process described in §6.2 before starting the subsystem interface map.

Example applications

Subsystem interface maps are used frequently in electronics in the form of schematic diagrams. These schematics trace the expected inputs and outputs of components, and by studying the schematic you can figure out how the circuit works. Some components act as 'black boxes', where it is not necessary to describe the internal behaviour. This is exactly the same principle for Subsystem Interfaces.

Similar to…

Subsystem interfaces are similar to classes and subroutines in software programming. A function or class will receive an input, conduct some processing, and send an output. If a data structure changes, it will have an effect on the whole program, and so establishing the expected input and output types is very important.

Steps

Constructing a Functional Block Diagram is the first step, describing what functions are packaged into each subsystem. This is easily created from the Functional Allocation step in §5.3.

- cluster the brainstormed system elements from your model boundary chart and functional analysis
- decide what the subsystems are, keeping in mind that you are aiming to reduce interdependence between subsystems

The second step is to move from functions to components, and to start describing the physical components:

- replace the functions with physical subsystems and components
- each system and subsystem should have inputs and outputs, and that relationship should be drawn using an arrow. Each arrow should have a label describing what is travelling (eg wireless signal, heat) and what units/standards are involved (eg 802.11n, joules)
- consider might be external elements that the system relies on (eg. wireless transceiver)

Hints

- If your system is large, it may be worthwhile constructing a system-level interface, and then drawing the subsystems separately down to the component level
- Ensure that you are using standard interfaces between subsystems
- Consider modularity in your design - a goal to strive for is for each subsystem to be replaced without requiring re-engineering of other subsystems.
- Is your system prone to failure? What redundancies may be required to ensure safe operation?

Core resources

- Principles of System Architecture - this is a how-to on the principles. These are actually good principles to apply to all of your systems engineering endeavours [Online or PDF]
- Examples of subsystem interface diagrams [PDF]

6.2 Subsystem Mapping (to Functions and Design Requirements)

Trace the customer requirements to functions and the functions to subsystems

Systems engineering is a control process for a project. It's a way to document design decisions, and make informed choices about design. Mapping the design requirements to functions, and functions to subsystems and ultimately stakeholders is important for traceability. If a design requirement changes, it is important to know exactly how this change will manifest through the design, and inform the appropriate stakeholders. In the reverse situation, if stakeholders change, it is important to know which subsystems are likely to be affected.

Example applications

This traceability should be flexible to your needs. We are looking at design requirements, functions, subsystems and stakeholders because these are the factors that we've examined. If your project has gone through an extensive design attributes or work breakdown structure phase, these might be useful considerations to map. The key idea is that you can trace your process backwards and forwards.

Steps

Described here is the process for the mapping we're using in ENGN2225. Adjust and apply as required for your project's instance.

- List all the design requirements established during your Requirements Analysis (note: these may be customer requirements, secondary or tertiary attributes, engineering characteristics - whatever is more useful for your project. Put this in the first column of the table.
- For each requirement, list the functional groups that meet that requirement in the second column. There may be functions that are outside your scope, but include these in this process
- Then list the corresponding components that are responsible for delivering that function
- If appropriate, list the stakeholder or team responsible for delivering the component

Hints

- Where a function or component sits across multiple requirements, ensure that these are listed multiple times. A single requirement may have many important components - make sure that you keep the mapping at a useful level for its purpose.
- It may be easier to consider top-level requirements for the purpose of this exercise, as the documentation may get unwieldy as the number of requirements increase
- It may be more appropriate to use tertiary attributes rather than functional groups - this is what is done in the example readings. The functions should be relatively easy to substitute in, as you should already have them at hand.
- Ensure that you use the mapping to construct insights about your system - is there one crucial element that may cause a problem if it fails or is not delivered? Does one requirement determine the choices around too much of your system?

Core resources

SlalomTech, 2009, Attributes Cascade of an Instrumented Whitewater Slalom Gate System [PDF]

Further resources

- Smith, E., and Bahill, T., 2008, Attribute Substitution in Systems Engineering, InterScience. [PDF]
- Birkhofer, H., and Waeldele, M., 2008, Properties and Characteristics and Attributes and… An Approach on Structuring the Description of Technical Systems, AEDS [PDF]

6.3 COTS v Custom

Should you design using existing components, or develop your own?

Before testing and evaluating your design, and ahead of communicating your design back to stakeholders, an important decision needs to be made: can your system be built using existing components, or does it require developing new designs. Using commercial-off-the-shelf (COTS) products can speed up your development, but will likely bring in constraints and trade-offs that weren't accounted for in your design to date. Alternatively, custom production may remove these constraints, but are generally considered higher-risk, more costly and likely to take longer to produce.

Ultimately, the decision to use COTS, modified COTS or custom development should reflect the importance of your customer requirements.

Example applications

There are three common situations where this decision is key:

Software development: often companies have specific needs for software based on their own internal processes. COTS software often lacks the additional functionality required, and custom development is often too costly if the skills aren't already in-house. Often plug-ins can be a way to bridge this gap.

Electronics hardware development: the cost of electronics has dropped drastically over recent years with a great increase in function. Devices such as Arduinos and the Raspberry Pi allow for rapid prototyping, but often come with more functionality than required. Using this COTS product to prototype an idea before moving to custom production

Physical prototyping: when designing a physical prototype, using material that you can readily get your hands on is usually a cheaper way to develop your prototype, rather than spending money tooling and configuring machinery. Once you've tested the prototype, you'll have a better idea of how the design might perform and you've hopefully figured out some improvements.

Steps

To help you decide whether COTS, modified COTS or custom design is required, you can go back to the concept generation tool: concept combination table (see Ulrich & Eppinger 1995, p96).

- Create columns for the subsystems described in §6.2
- List all the COTS, modified COTS or custom options to achieve the functions required of that subsystem
- Choose an assembly that best helps you to meet your customer requirements

Note: it may be worthwhile taking these ideas to an evaluation matrix (see §7.3)

Hints

- Comparing COTS, modified COTS and custom design becomes a good comparison for the evaluation of designs if your group has narrowed to a single design.
- Consider the topic in relation to different phases of production: the needs of proof-of-concept and development stages are very different

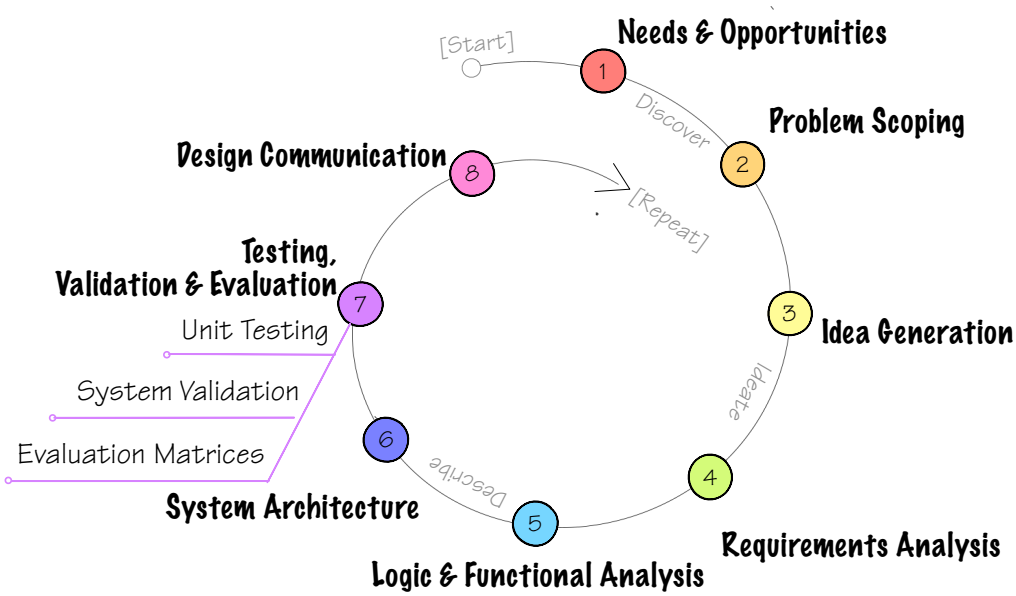
Core resources

- McKinney, D., Impact of COTS Software and Technology on Systems Engineering, presentation to INCOSE Chapters [PDF]
- Ulrich, K.T., and S.D. Eppinger, 1995. Product Design and Development. McGraw-Hill. Chapter 5 - Concept Generation [PDF, 25 pages] (from concept generation topic)

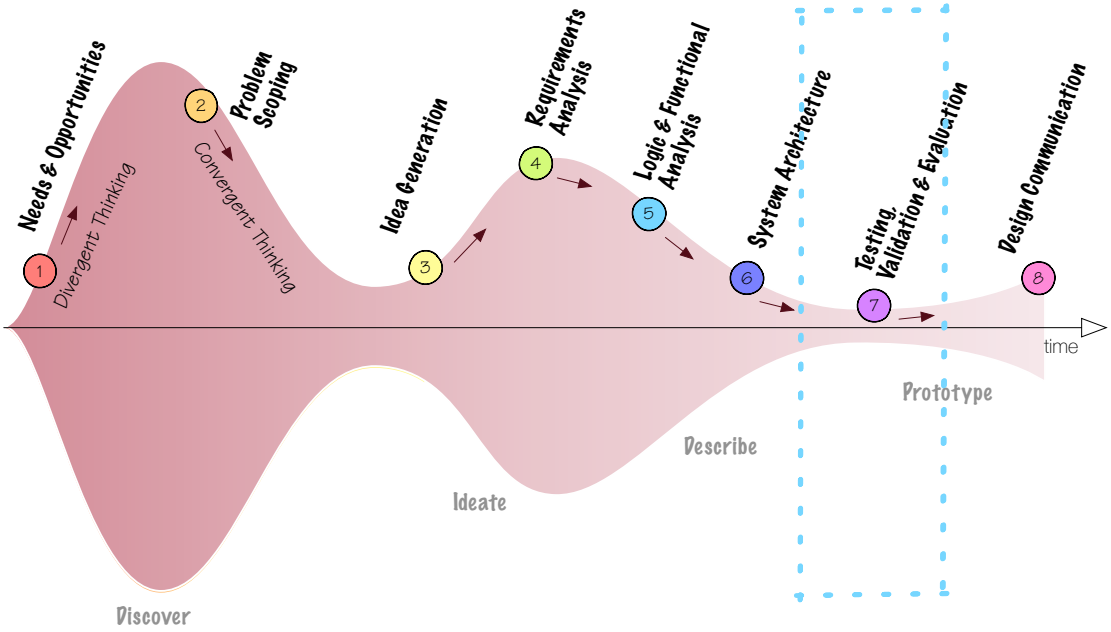
7. TESTING, VALIDATION AND EVALUATION

In the testing, validation and evaluation topic, we measure your design against the specified customer and design requirements. Unit testing is a process used to compare a design against predetermined requirements, validation occurs at a higher level, ensuring that subsystems deliver the required output. Evaluation is a tool for comparing alternatives.

We will use three tools: unit testing, system validation, and evaluation matrices.



At this stage of the process, you are finalising your design decisions, and deciding on a final design (or at least a preferred design to develop further).



7.1 Unit Testing

Unit testing is used to test components of your design against your requirements

Testing and verification of modules in your system is important to understand whether or not your design matches or will match up with your requirements. There are a number of different methods that involve testing, you should be able to identify which type of testing applies to each design requirement or attribute.

Testing Types

There are five types of unit testing that we will consider in this topic. It is likely for your project, that only the first three are relevant:

- Analytical testing, such as using analytical models (CAD, CAM) or textbook equations
- Proof-of-concept testing (Type I), such as using breadboards, mock-ups and rapid prototyping
- Model(/prototype) testing (Type II), such as the construction of prototype
- Operational testing (Type III), for when the design introduced into the field
- Support testing (Type IV), for when the design is in operational use

For your project, Operational and Support testing are likely to be sometime into the future if the project continues. At this stage of your project, you should almost certainly be able to engage with Analytical and Proof-of-concept testing.

Goals for this topic

Depending on the scope and progress of your project, your goals at this stage will be different. The purpose of testing in this course is to demonstrate to your client that they should have confidence in your work.

- at minimum, design a test procedure for a subsystem or function in your design at a future proof-of-concept stage
- if possible, demonstrate how your design will complete an analytical test. For example, using basic physics equations, or computer aided design, models, schematics to demonstrate how the proposed design will function
- if possible, demonstrate at a proof-of-concept level that a subsystem functions as expected. For example, a breadboard design of a single subsystem or function

Steps

A test design should include:

- A descriptive title, test scope, version number of the test, and responsible officer
- Any specialist terminology or calculations required for interpreting the test
- The procedure, apparatus and safety measures
- Analysis and interpretation of the results

Hints

- Just like the experimental method, tests should be robust, objective and repeatable.
- Document your test procedure and results, and use the results in your evaluation process.

Core resources

- Blanchard, B.S., W.J. Fabrycky, Systems Engineering and Analysis, Fifth ed. Pearson, New Jersey, 2011. Testing types [PDF, pp166-171]
- Australian Design Rules examples of design standards of cars [Third Edition ADRs, look up some ADRs to get an understanding of testing procedures, eg: ADR 4/00 Seatbelts, ADR 6/00 Direction Indicators, ADR 79/00 Emissions Control for Light Vehicles]

7.2 System Validation

System validation tests the interoperability of your subsystems against requirements

System validation of your system tests the interaction between modules, as described in the system interface. Unit testing should have ensured that the individual modules operate as expected, and system validation tests that they work together.

Testing Types

Of the five types of unit testing described in unit testing, it is the later three that are relevant at this stage:

- Model(/prototype) testing (Type II), such as the construction of prototype
- Operational testing (Type III), for when the design introduced into the field
- Support testing (Type IV), for when the design is in operational use

For your project, it is likely that only Model/prototype testing will be possible, if at all. You may also take a 'proof-of-concept' approach to validating the system interactions. The important difference between unit testing and system is that the unit testing occurs inside modules, and the system testing occurs between modules.

Goals for this topic

Depending on the scope and progress of your project, your goals at this stage will be different. The purpose of testing in this course is to demonstrate to your client that they should have confidence in your work.

- at minimum, design a test procedure to test at least one subsystem interaction at a future proof-of-concept stage
- if possible, demonstrate at a proof-of-concept or prototype level that subsystems interact as expected. For example, using a breadboard design of a single subsystem to manipulate a mechanical subsystem

Steps

A test design should follow a similar procedure as for Unit Testing.

Hints

- Just like the experimental method, tests should be robust, objective and repeatable.
- Test interactions between individual subsystems before testing the whole system at once. As subsystems are included, the dynamic complexity of the system increases, making it difficult to isolate problems.
- If you do have a complete prototype, consider conducting isolation testing to ensure that the expected behaviour occurs between modules

Core resources

As for Unit testing

7.3 Evaluation Matrices

A process for evaluating designs against the requirements, and making a design selection

An evaluation process takes us back to the customer requirements, and evaluates the options against these requirements. Ideally, the evaluation is informed by testing. The evaluation process itself is arbitrary, and there are many different evaluation rules that you can use - selecting the right procedure is important to get the best results for your end design.

Evaluation with multiple criteria

Choose the most appropriate evaluation method, given the requirements for your project:

- direct ranking of alternatives: using a pairwise to determine the ordered rank
- comparing alternatives against each other: choosing the best between alternatives
- comparing alternatives against a standard: the alternatives must meet a minimum standard
- comparing criteria across alternatives: do the alternatives meet the criteria by importance
- weighted evaluation*: such as the tabular additive method, which takes into consideration the relative importance of each criteria
- under risk: which compares the possible future or expected value of the alternative
- under uncertainty*: which allows for comparison where the results are uncertain

* these methods are recommended, but are not necessarily the right choice for all evaluation processes

Steps

As with testing, an evaluation process should be transparent and repeatable. Each evaluation method requires tabulating alternatives against requirements.

- Construct a table with the relevant customer or design requirements in the first column, the relative importance of the requirement in the second column
- Place the alternative designs (usually at least three) in the following columns, and put the benchmark or TPM as a reference in the last column.
- Fill in the relevant performance data for the alternative designs against the requirements

If completing a direct comparison:

- choose the best alternative against the benchmark using the chosen decision rules

If completing a weighted evaluation:

- score the alternative on a defined scale (usually between 0-5) against the requirement
- multiply out and tally the alternatives, selecting the alternative that performs best against the requirements

If completing an evaluation under uncertainty:

- present the results in a graph with all possible values, and indicate the values that likely or suitable

Hints

- Make sure that you use the evaluation process to objectively show the best result, not show the result that you want.
- The design performs best against the criteria shouldn't necessarily be the design to choose - use the evaluation to inform your decision, not as the decision

Core resources

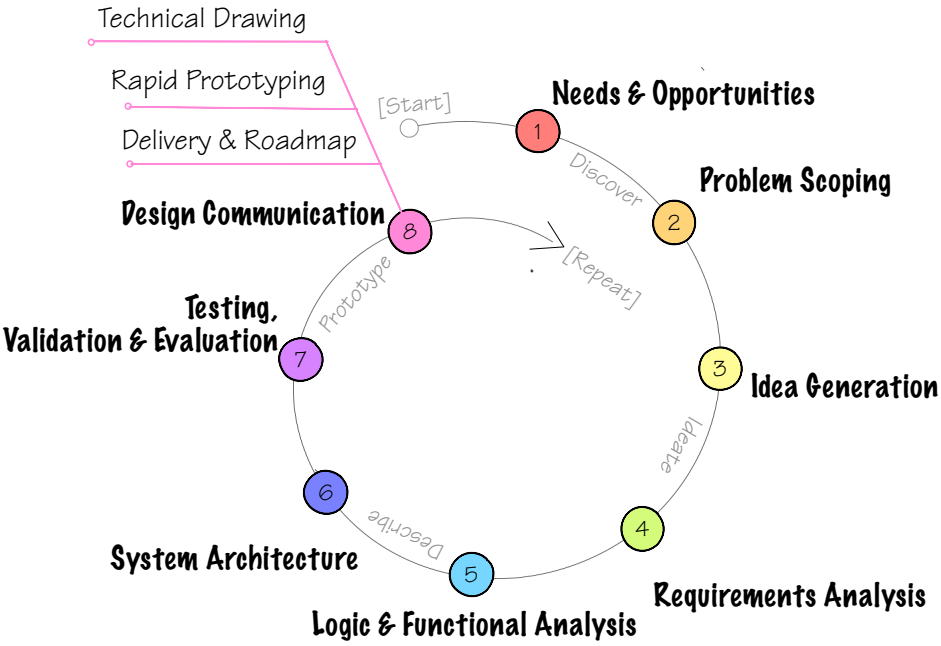
- Evaluation Matrices worksheet. This is a good demonstration on how you can use your customer requirements to identify the best solution [PDF worksheet]
- Chris' 2012 ENGN2226 Lecture Slides on evaluation of models, includes a demonstration of a number of design evaluation models [PDF slides]

8. DESIGN COMMUNICATION

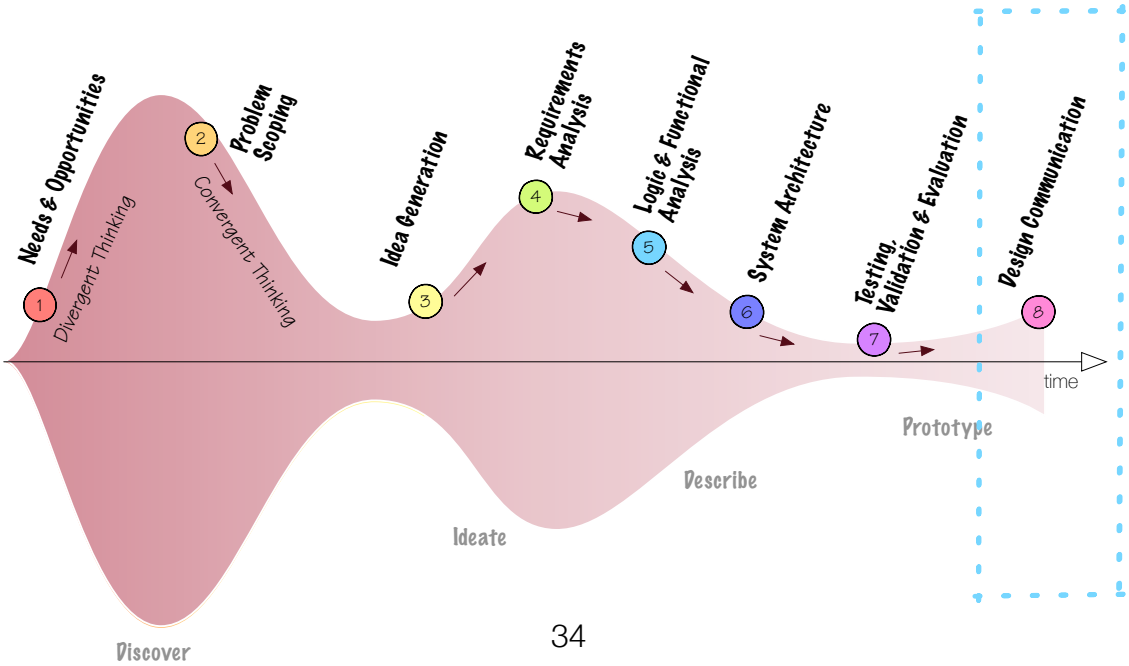
There is no point going through the design process if the design isn't communicated to a general audience. The Design Communication topic provides some tools to communicate your design, but depending on your design, there are a number of other ways that you could use - explore the topics that best suit your design.

Note that this topic is not part of the Jigsaw topics. This is intentional, as it's quite important that the team works together towards this goal alongside the topics (for example, it would be unreasonable to give the role of prototyping to a single team member).

We will use three tools: technical drawing, rapid prototyping and delivery & roadmap.



This is the final stage in our design cycle. If this were a real-world project, it would likely be the stage gate for the start of the next design cycle.



8.1 Technical Drawing

A technical drawing will show your design clearly in a graphical form

Technical drawings have been used to describe designs for centuries. A modern equivalent is a Computer-Aided Drawing (CAD), which can be readily drawn using software packages such as SolidWorks or free alternatives such as Google SketchUp.

Technical drawings should clearly help the reader or audience understand how your design works.

Example application

If your design is a physical object, technical drawings are used when registering a design patent. Design patents are used to protect your design so that you can gain a commercial advantage, and might be a consideration if your design is novel and has broad potential. Google maintains a Patent database search via Google Scholar, and many ANU engineering alumni have worked at IPAustralia, our patent office (see guest lecture slides from 2015)

Technical drawings are often shown in views: top, side, front and isometric. If your design has multiple components, it might be good to show a cross-section or exploded view of your design.

Alternative representations

The purpose of this topic is to communicate your design. You might find these alternatives more relevant depending on your project or idea:

- Wireframes are useful for showing Graphical User Interface (GUI) design. Many GUIs have design standards, such as Apple's Human Interface Guidelines [website]
- Videos or storyboards are a great way to show how a design will work. A good way to structure a video is using the Minto method described below
- Think about using rapid prototyping to communicate your design (see §8.2).

Minto method

The Minto method is a useful way to structure a narrative around your design

- Situation: what is the situation you are addressing?
- Complication: why is this an issue that needs to be addressed? Problematise your question
- Resolution: what should we do to resolve the problem?
- Governing Thought: show how your design will fulfil the design need

Golden Circle

Another way to communicate your design is using the Golden Circle technique, shown by Simon Sinek in his TED talk [video 3m39s]

- start with 'Why'- few organisations understand why they do something
- then 'How' - some organisations know how they do what they do differently
- then 'What' - all organisations know what they offer

Core resources

- MITOpenCourseware Design Handbook: Engineering Drawing and Sketching [Online or as PDF]

Further resources

- See the MITOpenCourseware page for further examples, such as Creating CAD components and Assemblies [Online]

8.2 Rapid Prototyping

Rapid prototyping is prototyping at the speed of thought

Rapid prototyping is a process that tests out an idea, and uses the process to learn. The purpose is not to have a fully functioning device or design, but rather demonstrate and test out an idea. Prototyping is also one of the most effective activities you can undertake to communicate your idea. This could be a physical artefact, a mock-up, sketches, role-playing, demonstrations - anything to help you and your audience understand how your idea will work, get feedback, and iterate.

Example application

History will likely tell us that Google Glass was just another technology failure from Google, just like Google Wave. But, in the world of innovation, failure only occurs if you don't learn from the process. Rapid prototyping takes on the same philosophy - you should push your idea to failure, because you should learn from it, and make it better.

In prototyping Google Glass, Tom Chi developed his rules for prototyping:

- Find the quickest path to experience
- Doing is the best kind of thinking
- Use materials that move at the speed of thought to maximise your rate of learning

Steps

A good way to approach prototyping is to use IDEO.org's tools for prototyping in their Human Centred Design Guide:

- Determine what to prototype (p111)
- Mock-up a physical (or digital, if you're well-versed in the technology) rapid prototype. Think about using one of these techniques, and follow Tom Chi's rules:
    - storyboard a process or interaction (p113)
    - role-play an interaction or exchange (p118)
    - make a model or mock-up using materials you have to hand (eg, string, paper, glue, wire, etc)
- Test it out. Take the rapid prototype and give it to your user and get feedback
- Integrate the feedback into your next rapid prototype, and document what you've learnt

Hints

- Rapid prototypes don't have to be perfect - just good enough to test out the idea
- Rapid prototypes shouldn't be expensive to make
- I always think of Play School when I think of rapid prototyping - a box and paper plates can quickly become a bus, train, stage, drum, percussion set, mobile, magic box, etc. Rapid prototyping works best when you let your imagination run free
- Once you've rapid prototyped ideas with simple materials, think about developing the prototype further using techniques from the Maker and Hacker movement, such as 3D printing, or breadboarding with an Arduino.

Core resources

- IDEO.org's *Field Guide to Human-Centred Design* has a great section on Ideation through rapid prototyping. The plan above has been assimilated from this resource. [Online or relevant sections as PDF]
- Tom Chi's TED Ed lesson on rapid prototyping [Online]

8.3 Delivery & Roadmap

After you deliver your design, a roadmap will help you map the future with your design

The delivery of your ideas back to the client might be the end of your team's project, but it's not the end for your client. It's important that the project is handed back to your client in a consultative process, and use this as process to learn more about your project. When you deliver your project, it's important that it goes back to your client alive, and with a future.

So, what happens now? Many university projects fail to go forward because the original team moves on and the new team doesn't know where to go. As the current experts on the project, it is an important legacy to outline the future direction of the project.

When framing your roadmap, think big picture. Imagine what you would do now if you were to start again today with all the knowledge you've accumulated about the project.

Delivery

For your ENGN2225 project, the delivery of your project is through the poster, poster session and design recommendations. These will be handed back to the client (unless there is a reason not to share your design back to your client).

When you deliver your project back to your client, you should avoid using technical jargon or presenting the systems engineering approach back to them. Deliver your design described in a way that the client can use and understand.

Roadmap

As part of the delivery back to your client, you should include a roadmap for future development. This should serve as a guide to the next project team. You should also not expect them to continue *your* project - the roadmap should provide the next project team with the inspiration to take the project further.

- Plot out a plan for the next six months of the project
    - What were some of the good ideas that should be continued?
    - What were some of the ideas that should be explored further, but you haven't had a chance?
    - How could some of your ideas be piloted?
    - How did you build on feedback from the client? What advice would you give to the next team?
- Chunk out some milestones that might be useful for the future of the project
- Identify the skill sets that might be useful for the future project team. Were there any avenues in your design that you didn't follow up because your team didn't have the skills
- Identify resources that might be required to take the project to the next stage
- Identify the partners that will be required to develop the future project. Who would you advise your the next project team to work with?
- Identify the funding requirements to continue the project into the next phase

Core resources

- IDEO.org's *Field Guide to Human-Centred Design* has a section on Implementation. The plan above has been assimilated from this resource. [Online or relevant sections as PDF]