In this class we started looking at requirements analysis, considered different elicitation methods, and how we might manage the process of collecting and documenting requirements. I replaced the Alan Dennis Book with a couple of different selections. Make sure you check the syllabus for an adjusted reading list.

To place requirements analysis in the appropriate context we’ll use the classic waterfall model, but I’ll simplify the context into the following basic steps for a project:

1. Define the problem
2. Describe what we need to do to solve the problem
3. Design how we will solve the problem
4. Build (or buy) something that solves the problem
5. Verify that we have a solution for the problem
6. Put the solution in place
7. Move on to the next problem

We have a lifecycle for requirements – I know everything I talk about is process…

1. Elicit (or discover) requirements
2. Analyze what we discover
3. Describe (or document) what we need to do
4. Verify that we have not missed anything that contributes to solving the problem
There are lots of ways to elicit requirements. Here is a partial list:

1. Interviews
2. Observe
3. Study the “As is”
4. Survey
5. Research
6. Prototype
7. Brainstorm

As you embark on the requirements analysis phase of your project you will need to consider a variety of different properties for the system requirements. These are referred to as “ilities” in software engineering. Here is a list, but I leave it to you to consider what they might mean:

- Usability
- Maintainability
- Flexibility
- Testability
- Scalability
- Availability
- Extensibility
- Protectability (security)
- Portability
- Compatibility
- Backwards Compatibility
- Interoperability
- Reusability
- Quality
- Marketability
- Configurability
- Auditability

Although you will undoubtedly discover many different requirements for your project you will need to construct a time budget. You must include in your planning the need to verify that you have excellence in your requirements. The ultimate quality of your project will depend on the level of quality in your requirements. Here are some properties that you can use to reason about the quality of your requirements:

1. That your requirements are **complete** – nothing is left out.
2. That you have **correctly** described the requirements.
3. That the requirements you have identified are **feasible**.
4. That you have requirements that are **necessary** and that none are unnecessary.
5. That the requirements are **prioritized**.
6. That the requirements are **unambiguous**.
7. That you have **verified** the requirements analytically and with the necessary stakeholders.
The readings describe different ways of documenting requirements, and my only requirement for the class is that you decide in advance how you will document your requirements. I prefer, however, that you develop use cases that provide sufficient detail to drive your design and implementation.

You should consider the use of personas for human users, but consider that systems also have actions that they perform. In Unified Modeling terms, both human users and systems are categorized as “Actors”, each of which takes action to accomplish a step (or set of steps) in a process.

You need to work from high level descriptions to low level descriptions. Don’t just dive into the details. See how everything hangs together before specifying in minute detail what has to happen. In all cases develop a numbering convention for your requirements. Numbering provides a way to connect everything according to a hierarchical model.

In class we worked out a simple use case for a user placing an order with an online store.

Use Case Number: 1.0
Use Case Name: Place Order
Use Case Description:
After the user has selected items to order, the user will give payment information and shipping location. When the order is placed a confirmation will be given to the user. Some users will have existing accounts.

Actors: Registered user
Non-registered user
Order system
Billing system

Triggers: The user is done shopping and indicates that they want to checkout.

Precondition: User has selected items to purchase
Post conditions: Order will be placed in system
The user will have a confirmation

Normal flow: 1: The user indicates they want to checkout
2: The order system will show previous billing and shipping info
3: The user confirms billing and shipping
4: The order system shows the total cost
5: The user confirms the order
6: The order system will provide details to the billing system
7: The billing system will verify payment and report back status
8: The order system will provide an order confirmation

Alternate flows: 3.A.1: The user enters alternate billing and/or shipping info
5.A.1: The user cancels the order
8.A.1: The order system confirm the disapproved order