SOFTWARE ENGINEERING

On the importance of planning...
COMPLIMENT

From division head of Microsoft in Boulder (extracted from email sent to Bill Hoff):

Second, you and your peers at CSM produce incredible students. More importantly for us, they step into the work environment more quickly and gracefully than students from any other university regardless of how prestigious. You guys are doing something very right at CSM and you should be proud of it.
Steve McConnell describes how small projects aren't necessarily representative of the problems you'll encounter on larger projects:

People who have written a few small programs in college sometimes think that writing large, professional programs is the same kind of work -- only on a larger scale. It is not the same kind of work. I can *build a beautiful doghouse in my backyard in a few hours*. It might even take first prize at the county fair's doghouse competition. But that *does not imply that I have the expertise to build a skyscraper*. The skyscraper project requires an entirely more sophisticated kind of expertise.

WHAT DOESN’T WORK…

Dr. Paul Dorsey:

Projects are frequently built using a strategy that almost guarantees failure.

Building a large information system is like constructing a 20-story office building. If a bunch of electricians, plumbers, carpenters and contractors meet in a field, talk for a few hours and then start building, the building will be unstable if it even gets built at all. At one of my presentations, an audience member shared the quip that:

“If building engineers built buildings with the same care as software engineers build systems, the first woodpecker to come along would be the end of civilization as we know it.”

SOFTWARE ENGINEERING FAILURES

IBM survey in the success / failure rates of “change” projects finds;

1. Only 40% of projects met schedule, budget and quality goals
2. Best organizations are 10 times more successful than worst organizations
3. Biggest barriers to success listed as people factors:
   - Changing mindsets and attitudes - 58%
   - Corporate culture - 49%.
   - Lack of senior management support - 32%.
4. Underestimation of complexity listed as a factor in 35% of projects

http://calleam.com/WTPF/?page_id=1445
BEST PRACTICES

1. Development process. Make this a conscious choice. Consider size and scope of project. Agile is not always the answer.

2. Requirements. Are you creating what the customer wants? Are there non-functional requirements? (efficiency etc.)

3. Architecture. How do the pieces fit together?

4. Design. Agile does *not* mean no planning! (or no documentation) Guiding principle: keep it simple (You Ain’t Gonna Need It – YAGNI). How much design before coding? Agile: just enough...

6. Peer reviews of code.
7. Testing.

Other steps listed in web article, not covered here.
SYSTEM ARCHITECTURE
Making a plan
III. SOFTWARE ARCHITECTURE

The software architecture of a system is the set of structures needed to reason about the system, which comprise software elements, relations among them, and properties of both. The term also refers to documentation of a system's "software architecture." Documenting software architecture:

- facilitates communication between stakeholders,
- documents decisions about high-level design, and
- allows reuse of design components and patterns between projects.
SOFTWARE ARCHITECTURE – WHY?

- The software architecture discipline is centered on the idea of **reducing complexity through abstraction and separation of concerns**.
- The software architecture of a program or computing system is a *depiction* of the system that *aids in the understanding of how the system will behave*.
- Need a unifying architectural vision to ensure system qualities such as performance, modifiability, and security.
- Focus on the *interface* between the components (one of the most error-prone aspects of system design)

http://www.sei.cmu.edu/architecture/
ARCHITECTURE – EXAMPLE 1

See Symplified final report (2011) for explanation
ARCHITECTURE – EXAMPLE 2

ARCHITECTURE – EXAMPLE 3

- See ModsDesigns final report (2011)
STATE DIAGRAM*

- See ModsDesigns
ARCHITECTURE – EXAMPLE 4

- See Agilent final report (2011).
ARCHITECTURE – EXAMPLE 5

- Two examples from Newmont 2 Final report
ARCHITECTURE – EXAMPLE 6

- See SMT final report
ARCHITECTURE – EXAMPLE 7

- From SMT
Finite Automata/Activity Diagram

- From SMT

* System flow, supplements architecture
ARCHITECTURE – EXAMPLE 8

- From Recondo 2
MORE EXAMPLES

- More links available on Design Document page
TECHNICAL DESIGN

adding some details
TECHNICAL DESIGN

- Architecture diagrams focus on the interface between components. They are “big picture” drawings.
- It can also be important to focus on details of a particular component.
- These diagrams are likely more familiar to you.
FLOWCHART

- From Circle 77 (illustrates security process)
DATABASE SCHEMA

Database schema from ModsDesigns
ENTITY-RELATIONSHIP DIAGRAM

- ER diagram – Circle 77
For database tables you create, include supporting text that describes the various fields and relationships. That level of detail not needed for tables in an existing customer system.
UML

- Also from ModsDesigns

Remember that DIA has an option to not show attributes/methods.
NOW IT’S YOUR TURN

where the rubber meets the road
BRAINSTORM WITH YOUR TEAM

- Talk about your requirements
- Brainstorm how to represent the *architecture*
- When you have an idea, show it to your advisor.
  - these will be reviewed during advisor meetings... goal here is just to ensure you have some ideas to flesh out

- This is the planning phase. For the final report you will update the architecture diagram and add text descriptions of all the components. For now you just need diagrams.
- You may also add more technical design details in the final report.