* Introduce yourself (for example, your name, where you're from, interests, family, pets)
  + My name is Daniel Loranger, I am currently living in Orlando Florida near the Disney World parks.  I have a masters degree in Electronics and Computer Engineering, and I am working on the AS is CS to backfill some of the computer theory as despite my degree title, I am almost purely a hardware engineer with only a small handful of computer science type courses, which were mostly in the FPGA programming space.  I have worked most of my career in the medical devices electronics industry, with some side quests in the industrial devices markets such as deep water sonar equipment and automotive power distribution systems.
* What has been your favorite course in the computer science program so far? What did you enjoy about it?
  + To be frank, I have not enjoyed any of the courses.  I came to the program looking to improve my ability to work in more advanced programming languages such as doing more advanced C++, C# and similar languages in direct application to interacting with hardware as I have extensive years working with scripting languages and low level C at the bare metal space.  I have found most of the courses so far to be focused more on the theory of application designs, and almost nothing related to actually writing the code.  I would be far more interested in learning how to write drivers for hardware than learning about how to handle 10T items in an array and worrying about 0.1uS of timing delay.  I get those items are important for some developers, but working with hardware directly is sadly 0% of what I have experienced so far, yet I see every day software engineers that cannot program a piece of hardware without extensive struggles and direct involvement of myself or another EE to interpret the datasheets.  This class is just another example of what I was not interested in, but with 3 classes left to complete, my hopes are low, but I will push thru to complete the degree.  I doubt I will try again and just continue to hack my way thru as best I can.
* What are the major phases of the systems development lifecycle (SDLC)? Describe the key features of each phase in your own words.
  + From the text, the key phases are(Figure 1-2) "Planning", "Analysis", "Design", "Implementation" and "Maintenance"
    - Planning - This phase is primarily responsible for understanding the current tools and their strengths and weaknesses, and also identifying what the future needs and expectations are for the systems to be developed.  This includes understand hardware state of the art, programming language development, industry/consumer expectations, etc.  Once the needs are understood, the plan to implement the changes can begin, which may include setting feature priorities, major milestone delivery targets, key customers, etc.
    - Analysis - This is where the concepts and ideations are devolved from ideas to actionable requirements.  If the product must be "lightweight" becomes, the product shall weight less than xx.x lbs, blue color becomes blue pantone color 123456, etc, basically taking concepts to statements of work that are directly actionable and are directly testable for pass/fail criteria.  This may include attaching regulatory or other governing bodies and industry standards into the design which must be complied with to be legally authorized to sell products or meet customer expectations. Once all design inputs are gathered, an alternative solution is proposed and ultimately accepted for implementation.
    - Design – In the design phase, the alternative solution is deconstructed into logical and then physical specifications that encompass the overall look and feel, behaviors, boundary interfaces. In the logical phase, the entire implementation is completely hardware agnostic and generally also software agnostic in nature. Once logical design is completed, the logical elements are then used to define the hardware requirements that will be needed to support the logical implementations. This can include gathering analog data from the real world and converting it to data a machine can handle, understand the computing power required to perform complex computations, defining data flow infrastructure, etc.
    - Implementation – During implementation, the detailed work of putting together the software, hardware, etc. is performed. The software is written, tested, and verified as components and up into integrated sub systems that are running in conjunction with the required hardware platforms previously selected. In addition to actually developing the produced system, additional work will include user testing and training, generating documentation, and gaining market acceptance.
    - Maintenance – maintenance is the phase where the system is deployed and running successfully, but where new features can be added, workflows can be optimized, tools adapted to changing business needs, but always utilizing the original framework and methodologies.
* Requirements can be collected in various ways: reading documents, an in-person conversation, etc. Choose one of these methods and describe a strategy you would use to help you collect requirements with this method.
  + I like to use the interview/conversation approach to gathering requirements. Written formal requirements often abstract away the intent of the user in ways that can lead to a product that fulfills the requirements, but misses the customers needs in terms of workflows. For example, a tool for generating a written document could be implemented in a spreadsheet infrastructure, and in the end produce a written document, but it would be very painful to use. The end user really wanted a product that emulated a typewriter that allows collecting their thoughts in a lightweight free form interface such as a tool like notepad or a blank Microsoft word document that we are all familiar with. By having the conversation, a higher understand of not only the requirements can be generated, but it also allows learning how the user WANTS to use the product, and can also lead to great conversations the discover features or behaviors that are not captured. Many times this can fall under the requirements of “must facilitate user stories collected”.
* How has technology changed how people collect requirements? I think that the formality of requirements has been significantly changed, maybe for the better, but also for the worse. With highly distilled and very granular requirements, its easier to agree when a project is “DONE”, but I also feel that it can really be stifling as a designer as your requirements can be poorly written and force choices that are not necessarily beneficial to the overall product design. In addition, the requirements collection process can really hinder a business from being able to move quickly and capitalize on market opportunities as I typically see system design cycles that last from 5-10 years (prior to entering maintenance phase) from places that use bloated system design processes, where a start-up will do a light weight “what can we get done quickly and get to market” version of the requirements and be on the market in under a year or two.
* What ethical issues (data collection, bias, and so on) do you think need to be considered when collecting requirements? Certainly, data privacy must be a primary consideration, in the modern world, what is the privacy method planned/required for implementation, and related is the methods of storage/transit and encryption both locally and in transit. Consider the need for personally identifiable information, financial information, etc. Bias considerations could certainly include only asking about the needs of local customers that may not represent the overall population, such as assuming everyone has insurance, or there might be a local dialect (such as English only) that is convenient as a native speaker, but may restrict inputs from other cultures for example prohibiting the ability to use extended character text such as kanji for localizations. Another major bias, is only considering the technical team while excluding other business units such as regulatory, manufacturing, support, etc that may need to live with the output for the maintenance life of the system.