Computer Science Capstone Design

Assignment: Capstone Presentation

Dry Run: 20pts; Capstone Presentation: 100pts

Overview

This is it! You've figured out what to build, done the design, and sweated through the hard implementation work. Plus you've managed that difficult client, and you've (hopefully!) turned a rag-tag group of CS seniors into a high-performing software development team. It's time to tell the real world all about your project!

The overall nature of your Final Capstone Presentation is fairly similar to the Design Reviews, only this time you'll have a more mixed audience ranging from CS faculty, to general faculty from around NAU, to potential future employers, to your parents and your grandma. Thus, you want to design a talk that has enough to satisfy each of these groups. Your communication goal is also subtly different: In contrast to a DR, where you are specifically interested in comments and helpful suggestions from your peers in your company, your Capstone presentation is focused more outward, aimed at providing a coherent, retrospective "story" about your project. Consequently, your primary goals for this presentation are:

- Present the problem in a way that everyone in the audience can connect to. This is no different than the DR, except the diverse and generally less technical nature of the audience makes it even more important for you to do good job here. You've been working on how to explain your client's problem and motivate the need for your project for three DRs now. Hopefully you have this down pat!
- Review the whole project. Same basic list of topics as all the DRs, except that now you're not focusing on any particular stage, as you were for the DRs, but are reviewing all of them. Thus you'll intro the problem, then outline the project phases (nice diagram) as an outline for you talk. Then you'll walk through each: Requirements development, design, implementation, and testing. Save some extra time/focus for showing off the final result, since that's the most important part! You should certainly have a demo (live or not) of some sort to show your working solution.
- You are trying to convince the audience not only that the project was worthwhile and that you produced something, but also that it was *effective*. That it will have **an impact**. Thus, your concluding slide or two should emphasize what this impact (or expected impact). After you review your solution's coolest highlights, review how your product will impact the client. Be specific: time saved, accuracy…estimate some realistic numbers. Another nice thing is to show the value to your client: estimate how many person-hours you, as a team, invested in this project. Then multiply this by a realistic consulting rate charged by a software consulting company, say, $100/hr. This is the monetary value that the client has just received.

You should absolutely get your client to come to Capstone. This is a big day for you…and for the happy client as well. In fact, in order to become Capstone clients, they all had to commit to doing everything possible to attend Capstone. If you client says they can't make it, be sure to emphasize the importance of the event and see if you can't change his/her mind. Be sure to help them be there: send a campus map with the right building marked, highlight convenient parking lots, tell them how to get a day pass for parking. **Do this several weeks in advance**, so that the client can plan ahead!

The Assignment

In this assignment, you will prepare and present a final presentation for your project. The overall topic flow for this is similar to the DRs, as noted above:

- Intro: Intro the overall project area. Sell it as a vital/valuable market. Intro client's business and his/her needs.
- Problem and Solution Statement: Remind us of what's broken/inefficient, and what your vision for a solution is.
- Present an overview of the project stages you traversed in tackling the project: requirements, design, implementation, and testing/refinement. This provides the outline for the rest of the talk. Then walk through those phases, briefly summarizing what happened in each. Should feel like an engaging story!
- Review that Challenges/Resolutions. After you talk about what happened, people want to know the highlights…what was hard/challenging/frustrating on the project…and how did the heros overcome/solve these issues.
- Review the project plan and schedule: Basically the same Gantt you've had, except it shows the whole project period from fall to now. You've introduced/discussed the phases earlier, now show us how they lay out on the schedule, showing what happened when. Walk us through it briefly.
- Conclusion: summarize and wrap it all up nicely. Emphasis on impacts!

Final Capstone Presentation: Detailed Content Outline

Building on the overview just presented, the following outline gives a bit more detail on what you may want to discuss at each step of the talk. Remember that this is only a general outline! In the end, it's up to you to tell a compelling and coherent story of your project, so feel free to change the topic order as needed, or to spend more/less time in any given section.

Introduction (< 30 secs)

The usual. Begin by introducing yourselves briefly: Go through each team member's name and role(s) on the project, as well as your team name, client, faculty mentor.

Problem Statement (about 2-5 minutes, depending on domain complexity)

As we've said from the start, this is an absolutely key section. If you don't explain and motivate your project *very very clearly* here, you'll be in grave danger of having lost your audience. Lacking a clear idea of what you're doing, they literally won't be able to grasp the rest of your talk. And lacking a strong motivation of why this project matters, what its impact for the client and/or society might be, they won't care enough to listen. Hopefully you have learned from DR feedback and have successively refined this critical part of your talk to the point that you can really deliver this well.

As a review, one more time: Begin by introducing the BIG picture: talk about the general area that the client's problem falls in. If possible, give some stats on how big/important/popular/vital this area is. If possible, connect it to the audience by describing an example scenario/problem that the audience can easily relate to. Maybe they've run into it themselves or, if not, you've explained the dynamics of the domain so well that they can easily imagine that the situation would be problematic. If it's a complex area, graphics that help you explain processes, entities or process flows relevant to understanding how business in this sector works can help support you. THEN introduce your introducing your sponsor and the organization they're attached to, and say how they and their organization contribute within the larger picture of the sector you intro'd. What do they produce, how does it fit into that larger sector, and what is the volume/importance/user base of their part? What it the process by which your client produces whatever data/product that they are producing?

Ok, now the audience hopefully fully understands what your client does and how it matters. Now (and only now!) are you ready to go on to describe the problem: what's broken, why you were hired. This should be easy if you've already described the workflow/dynamics of how your client's production/business process works: you then just have to explain what's bad/inefficient about it. Describe the problem in overall terms briefly, then get down to **bulleting out a few specific things** that are not satisfactory. By the end of this, your audience should be really clear on why a software solution is needed.

Solution Overview (about 1-8 minutes, depending on how you tackle this)

Now you need to outline your plan for fixing the problems you just outlined. Again, this is easy if you've done a good job of describing the business workflow, and then pointing out specific problems with it: you just connect your solution right into this discussion. Begin with a broad statement of your overall solution, e.g., "The solution that we envision to address the client needs just outlined is to transition the entire workflow to a secure, highly reactive web application that …". This is a great place to have process- or data-flow graphics you used earlier to describe the clients business processes re-appear, with the elements you are fixing/adding clearly highlighted. After you've intro'd what you envision overall, **present a list of bulleted specific features of your solution**; choose them so that it's pretty clear to listeners that your solution features will certainly address the problem issues bulleted out earlier. Explain as much as needed (e.g. walk us through figures, whatever) to make it clear.

POSSIBLE DEMO LOCATION HERE: Depending on how you think the flow is best for your talk, this might be one place to include a demo/walk-through of your solution…and then follow that with your detailed project process description. Or you could just give a screen short or two now to get across the idea, and save the full demo for closer to the end (See later section). Up to you.

In sum, the point of your solution statement is introduce and convince how your solution vision (potentially embodied in real software now) solves the clients specific problems. If you've done your work in the Problem Statement and Solution Vision, your audience will:

a) understand the problem domain and what problems your client has
b) will have a solid overall idea of what you have in mind to fix it,
c) and will (!!) be strongly convinced that your solution vision will actually fix the clients problems completely and elegantly.

OK, this is the end of the critical introductory info that establishes the solid foundation for your talk. You're about 3-6 minutes into the talk and everyone is hopefully absolutely clear on what you're up to, and is hooked on your project. Time to get to describing how you did the project and how it went. As always, smoothly lead in with a nice segue: "Now that we've established what we're doing, let's look in more detail at some project details and status". Then you can present an diagram showing the major project phases…which you then walk through in upcoming minutes:

Requirements/Specs review (about 1 minutes)

The first stage was requirements acquisition and analysis. Review what you did in the fall term: how did you elicit requirements, how many did you collect overall in the various (functional, performance, constraints) category? Then distill these out into 4-7 key higher level requirements you pursued. When you are done here, the audience should have a clear idea of the process you used to gather requirements, and of the outcome of that process, i.e., the key requirements you discovered..

Architecture and Implementation (about 2-4 minutes)

Now that everyone understands what you're building and why, it's time to show off your geek chops a little. As always, you must do this logically, with a good story flow going from high-level overview to flashing a few low-level details.

Architecture Overview. This establishes the overall picture of how your solution tackles the problem. Start with , just as in DR2 and DR3: Graphics are highly useful here: give an sketch of high-level modules/elements and how they are interconnected. So for a web app might have a front end (using some frameworks), a back end (using some langs/frameworks/DB), and a hosting solution. Intro each and then walk through how they interact. To drive this walk-through, it's useful to have a concrete domain-level example (use case)...then you just walk through that use case, explaining what functionalities each module contributes and/or how data flows through them as the user interacts with the system.

Implementation glimpses. Now that everyone understands the overall approach and rough structure of your system architecture, it's time to give the geeks in the audience some detail to feed on. Don't go crazy here, diving into random screens of low-level code! Again, start at the higher level: what frameworks/languages/product used for each part of architecture, and given a brief justification for why chosen. Now you just want to give a few select "glimpses" of the gory detail...exactly what does "implementation" in your system look like? So you lead in with "Given these frameworks, implementing our system meant...<whatever the process was for your project>". So for a web app project you might say "setting up a backend DB to drive the web app front end, and then creating that front end using our chosen <whatever> framework". Then give one slide that talks about DB backend design, e.g., gives the ER diagram (more as a graphic than something readable) and some stats: how many DB tables, performance expectations, how much data it is expected to hold, and any other design highlights. Then you'd do something similar for the front end: a graphic that maybe shows an editor that gives a glimpse of what "coding in framework X" looks like, plus bullets to walk through highlights: different kinds of files used by framework, how you proceeded to program them (strategy, iterations), etc.

You aim here is to hint at what it took to get it done, i.e., wow people into thinking "this took a bunch of complex work, and this team was clearly competent in doing it". As you finish your overview, you can remind the audience that your final as-build report will have all of the low level detail, and invite them to check that out if they want to know more.

Demo: Prototype Review (2-3 minutes)

If you didn't already place the demo up in the solution overview, this is where it should go. See what gives the best flow for your project...

You've worked hard all semester to get a nice beta prototype up…the audience is surely dying to see what you've done. If you haven't chosen to insert a walk-through of your prototype earlier on, e.g., as part of an extended solution vision, then this is the time to show it off. **Warning: Live demos are dangerous!** If something breaks or the network goes down, you're sunk. On the other hand, for some products, it's the cool dynamic aspects that are the most impressive. Ask yourself whether the live visual flow is critically important/impressive for your product. If not, a series of well-chosen screenshots to present and discuss are probably a better bet.

SIDE NOTE: HOW TO DESIGN A GOOD DEMO: The goal of this demo is to show off/convince the audience that you have created a solution that will address the client's problem. This means that, for certain, your demo should be driven by a clear, concrete usage scenario that exactly matches the client's use cases. This means that you start by setting up the scenario: "The key end users of our product are the <whatever> working for <client>, who are trying to <whatever the domain task is>. As we've shown [in your prob intro part], this used to be difficult/impossible. Using our product, it becomes easy. For instance, imagine that <describe your specific scenario>. Using our product, you'd start by...<now walk us through accomplishing this task with the product>. If you have *multiple* stakeholders (e.g. an end-user and an admin user), then you do two distinct scenarios and show each one briefly. Remember: you don't have to show EVERY feature of your product...just show off a clean flow of an imaginary user getting the job done.

Challenges and Resolutions (about 1-2 minutes)

This section is about the same as this term's DRs, except you are highlighting challenges that occurred anywhere in the project period. Look back at previous DRs and distill out the key challenges faced on this project….and how you solved them. We hope that, at this final stage, there are no longer any open/unsolved problems…but if this does happen for your project, be open and honest about it: "we wanted to do this, we tried it…but it didn't work account because X".

In any case, your goal here is to review the tricky challenges you faced and give some defense of your choice of solution. If you do this well, the audience will feel that you have conscientiously identified and tackled some hard issues in a reasonable way.

Schedule and Testing (about 2 minute)

You have now described what you did; now put it on a timeline. Give a Gantt chart showing major development phases (only. Skip detailed tasks) and how those laid out over the whole year. Briefly walk up through it, highlighting where major milestones were achieved.

Then convince us that you've tested the product you'll deliver: Overview your testing plan, then how it worked out: How many units, how many tests/unit average, integration testing. For user testing, talk about your testing/refinement strategy first, then say how it worked out: how many "tests" you did, and an example or two of the insights that came back ...plus, hopefully, the fix you did to repair that shortcoming. Your discussion of testing should make us think "wow, conscientious testing! We can be confident that it's a high quality product". Close with some summary statement of where you are: something like "schedule is right on time, hope to deliver early", "somewhat behind, but we think we can catch up", whatever.

Future Work (<1 minute)

Having talked in detail about the product you've developed and how, show your deep understanding of the problem domain by commenting on functions that were not in the clients requirements...but that emerged as something that would probably be useful to add. In essence, answer the questions "If the client were going to continue developing, what features should be prioritized to improve the product". This is NOT a listing of features you had on your plate and just didn't get done (though the list could include one or two of these, if that's how it worked out). Rather, you want to broadly paint the future direction of your product: what would you be including in Version2.0 and why.

Conclusion (about 1 minute)

Finish your talk by providing a solid summary of your presentation.

1. Remind folks of the big picture and how solving your problem is important.
2. Say something about the solution you've produced, and highlight a few features that make it especially cool/powerful.
3. Say something about the design and implementation phases, e.g., working closely with our client, we were able to get this project done on time, implementing x% of the requirements set for this project.
4. End by focusing on project impact: how it has affected your client (be specific: time saved, more accuracy, new capabilities for client, whatever…give us estimated numbers to make it real. If possible, give an estimate of project value to the client…and, of course, a statement regarding your plans to deliver the product and how the client is feeling about it (happy, satisfied, etc.).

When you finish, the audience should have a satisfied feeling, like they've just finished an interesting story and have arrived at the happy end. End with thank-yous, plus invite audience to come see your poster!

Closing comments and Logistics

The times given for each of the topical areas to discuss are, of course, only nominal. It will be up to each team to tailor the amount of discussion in each area to the project…and, of course, to the overall time you have for the presentation. The point is that, if you go overtime, you'll bore your audience and drown them with detail --- and you'll probably be stopped in mid-sentence and yanked from the stage by the session moderator. Embarrassing! Yet, if you use less than your allotted time, you've left some valuable time on the table that you could have used to explain some key points in your project more clearly. So you'll want to map out your time, try it, and adjust until you get it right.

Some more logistical details:

- The room assignments for the presentation "sessions" will be posted on the Capstone website as soon as they have been finalized by the Capstone organizers.
- **All members** of your team must participate in the presentation.
- The time slot allotted to each talk can vary slightly each year, as the organizers work to accommodate all teams within the times available. Check the UGRADS schedule (will be sent out by me around late March) to see how much time is scheduled for each team. Then subtract off about five minutes for questions. For instance, teams typically have about a **total of 25 minutes** for their presentation, including some time for questions and switching to next team. So you should **plan to talk for about 19-20 minutes,** to leave a few minutes for questions from the audience and the switch to the next presenters. Whatever happens, the next talk after your must start on time!
- The timing of the various talk elements in the outline above is just a suggestion, to give a rough idea of what to shoot for. If you used the *upper* bounds on each part, you'd be over time. The explanation needs (e.g. for problem intro) will vary widely between projects; it is up to each team to optimize the time available for their presentation.
- Dress up appropriately, meaning looking professional and credible…and eminently hire-able!

Deliverables

Final Capstone Presentation: Refine and practice your talk in front of teammates, classmates, and CS mentors.

The deliverables for this assignment are:

- A "dry-run" walk-through of your presentation with your CS mentor. Arrange a time with your mentor to do this!
- Your final public presentation, given at the designated time at UGRADS.
- Your "Team Facesheet", in hardcopy again, presented to your mentor at the public UGRADS session, so that your mentor can comment on each of your individual performances during the talk.