CS 428 Creating an Architecture and Design Document

Winter 2021

Bruce F. Webster

What is architecture?

- Fundamental organization of the system to be constructed
  - Focus on connections and interfaces among subsystems
- Grounded in the end-user's needs and requirements
  - Prioritization/selection of guiding principles and concepts in building that system
- Fundamental structure/environment of the solution
  - Choice of 'materials', 'location', and so forth
  - Resulting constraints and opportunities due to those choices
- Requires negotiation/buy-in among team members, management, end-users
  - Remember: "Architecture is a political act." – Tom Affinito

Some definitions of software architecture

- "To be architectural is to be
  - the most abstract depiction of the system
    - that enables reasoning about critical requirements
    - and constrains all subsequent refinements." (Clements et al., p. 23)
- The architecture of a software system:
  - Defines that system in terms of computational components and interactions among those components…
  - Shows the correspondence between the system requirements and elements of the constructed system…
  - Clarifies structural and semantic differences among components and interactions. (Shaw & Garlan, p. 3)

- Top-level design – functional, physical, and operational, the partitioning of which can be very important (the ‘what’)
- Creative, obsessive juggling of requirements, constraints, technology, costs, and standards (the ‘how’)
- Creating an enduring base for growth and change (the ‘why)
  - – cited in Rechtin (1991, p. 22)

An approach to software architecture (Spinrad)

- Conditions of customer delight – that is, your customer will love your solution because the architecture meets or embodies these aspects
- The ‘what’: draw your top-level design, **showing major subsystems and the interactions among them**
- The ‘how’: document your explicit choices and trade-offs in technology, approach, feature set
- The ‘why’: explain how the ‘what’ and the ‘how’ work towards product success; in other words, how your design (what) and choices (how) will delight the customer

What your architecture should include

- **Reliability**: sufficiently free from errors/downtime
- **Performance**: completes tasks in acceptable time
- **Functionality**: implements all critical/desirable features
- **Competitiveness**: fills need and is superior to other systems
- **Compatibility**: interacts effectively with existing IT systems/programs
- **Lifespan**: operates sufficiently long to achieve benefits
- **Deployment**: ships and installs in an acceptable timeframe
- **Support**: allows upgrading, expanding, and repairing over time
- **Cost**: can be developed, deployed, and supported within the budgeted time and cost

Quality goals affecting architecture

- Specific solutions to implementing architecture
  - Can be mandated and/or prohibited (“Thou shalt”, “Thou shalt not”)
  - Opportunity for design reuse (design patterns)
- Goal of ensuring conceptual unity in actual implementation
- Covers a wide variety of areas
  - UX/UI
  - Database design / data structure design
  - Patterns in module interfaces (including ‘deep interfaces’)
  - Coding standards and guidelines
  - Use of specific tools, solutions, languages, libraries
- Deliverables often depend upon methodology being used

What is design?

- Front matter: purpose of product & purpose of document
- Overall view of system architecture (major subsystems, connections)
- Divisions based on approach/team
  - Front end vs back end
  - Data/database design specifics
  - Game design principles
- Fill in details to allow implementation from the design
- Identify the hard problems up front and prioritize them

Suggested approach to architecture & design document

- Should be on your team's wiki in GitHub by Saturday at midnight
- Next Monday (03/01), in class, each team's chief architect will have to explain (briefly) the rational for that team's approach to architecture and design
- Don't forget status report also due by Saturday night
- Readings for next week:
  - The rest of *Peopleware* (chapter 21-36)
  - Webster #5
    - How to retain IT talent with goal alignment (Webster)
    - Anatomy of a runaway IT project (Webster)
    - Septic code (Webster)
    - Negotiations and lovesongs (Webster)

Assignment for this week: create & upload initial arch/design documents