Section 01: Life Cycle Objectives Review

Valentin Razmov

Outline

- Life Cycle Objectives Review milestone – group assignment #1
- The five constituent elements of a Life Cycle Objectives Review

Life Cycle Objectives (LCO) Review milestone

- Group assignment #1: Project Proposals
  - You need to work in pairs, so look for a partner
  - Assignment is available on the course web
  - Due next Wednesday, June 28 @ 10pm
  - Project Proposal presentations in-class on Thursday, June 29
  - ~10 minutes per presentation, so we can hear all
  - Today we cover the necessary elements of a proposal (a.k.a. Life Cycle Objectives Review).

Life Cycle Objectives Elements

- Operational Concepts
  - What is it? (High-level scope and objectives)
- System Requirements
  - What does it do for us? (Lower-level actual planned deliverables)
- System and Software Architecture
  - How? (Technically)
- Lifecycle Plan
  - Who wants it? Who’ll support it? (Resources needed)
- Feasibility Rationale
  - Given the constraints, is this realistic (can it be built)?

1: Operational Concepts

- Top-level system objectives and scope
  - What problem are you trying to solve? Why? For whom?
  - User community, environment, major benefits?
  - Goals and non-goals
    - To set realistic expectations in the audience

  **Tip:** This is what you should be able to explain in a 1-minute pitch (if you didn’t have more time) – a.k.a. “an elevator pitch.”

  **Tip:** It takes practice to refine this, so start early.
2: System Requirements

Essential features of the system

- What does the customer want from this system?
  - Look from the user’s perspective
  - Tip: Avoid details at the start; there’s time to evolve.
- Discuss main capabilities, outcomes, reliability and performance needs, appearance
- Customer involvement is important and beneficial
  - They know best what their interests and needs are, including what fits in their daily work and life patterns
  - ... even if they may not always express it very well
  - They understand the domain better than developers do.
  - Working jointly and openly with customers helps build trust, so any necessary changes are more acceptable.

2: System Requirements (cont.)

Essential features of the system

- This will be your initial written specification
  - Customers can review and sign off quickly or complain early.
  - Putting it in writing makes it less ambiguous than saying it.
  - Forces you to think of major functional areas and seek architectural defects early
  - "Failing to write a spec is the single biggest unnecessary risk you take in a software project" -- Joel Spolsky
  - Be concise yet complete
  - People get attached to their work even if it is no longer of value.
  - Tip: A picture / diagram is (often) worth 1000 words.
  - Tip: Scenarios and stories help, but avoid being verbose.

3: System and Software Architecture

High-level technical description but with enough detail to allow feasibility analysis

- Unlike the previous two elements, this is technical.
- Architectural flaws will only deepen as you go forward, so look for alternatives while it’s still early.
- Tip: Try to come up with several (at least 3) alternative architectural designs.
- Tip: Identify clients, servers, major software components, external 3rd party software, and the interactions between them.
- Tip: Pictures say 1000 words.

4: Life Cycle Plan

Identify stakeholders and their roles

- Users, architects, developers, testers, managers, etc.
- WWWWWHH:
  - Why / What / When / Who / Where / How / How
  - Objectives: Why is the system being developed?
  - Schedules: What will be done, When?
  - Responsibilities: Who will do it? Where are they?
  - Approach: How will the job be done?
  - Resources: How much of each resource?
- Tip: Make your best (educated) guess. Some of this will necessarily change. This is not a contract.

5: Feasibility Rationale

Conceptual integrity and compatibility

- Can this really be built with the available resources?
- Identify project risks
- What are the assumptions? Any unwarranted ones?
  - "If you make one or two ridiculous assumptions, you’ll find everything I say or do totally justified.”
  -- Ashleigh Brilliant, 1671
- Tip: Keep asking “why” until the assumptions emerge.
  - E.g., “Why do we need this?”, “Why is this good?”