**Administrivia**

- Lifecycle Architecture (LCA) group assignment will go out later today.
- Informal feedback meetings with LCO groups
  - EasyShare: Mon, 2:45pm-3:15pm, room 624
  - TeamForge: Mon, 3:15pm-3:45pm, room 624
  - FantasySportsLeague: ??
  - OpEnSpaCe: ??

**Lecture 06: Lifecycle Architecture Review**

Valentin Razmov

**Outline**

- Team Conversations: a few remarks
- Overview of Lifecycle Architecture (LCA) phase

**Resources**

- "Anchoring the Software Process", Barry Boehm
  - Especially pp. 1-10
- "Death March", by Edward Yourdon
  - Ch. 2: pp. 56-57

**The Next Project Stage: Lifecycle Architecture (LCA) Review**

- Culmination of the detailed planning and design
  - An elaboration of the LCO review document
  - Requires that the same five elements be addressed
  - But more details and decisions are expected, and fewer open options
  - LCO materials serve as a starting point
  - But no reason to stay too close to the original idea if your expanded team wants to change/adapt some aspects
  - Changes between LCO and LCA are frequently needed to improve focus and/or scope.
  - Major risks are resolved or a management plan created.
- Stakeholders must approve the decisions and the overall strategy before the project moves forward.
Basic LCA Elements

Operational concept
- Defines user community, environment, what it does and does not do, major benefits for all stakeholders

System requirements
- All features defined at this stage, including performance, reliability, security

System architecture
- Identifies any changes from original architecture, existing packages to be used, areas where changes are (proactively) anticipated in the design

Lifecycle plan
- Defines major milestones, maps tasks to resources, identifies roles and responsibilities

Feasibility rationale
- Remaining risks: likelihood, impact, risk mgmt plans

03 Jul 2006 CSE403, Summer'06, Lecture06

LCA Milestone Deliverables – at a High Level

Detailed requirements specification document
- Shows that you understand well what is being built
- Shows that you know how to build it
- Test plan document
- Schedule and task assignments
- Presentation

Note: These are all evolving documents, not set in stone. The LCA deliverables represent your latest understanding of what the issues are and what solutions you are envisioning.

03 Jul 2006 CSE403, Summer'06, Lecture06

Describing Specifications, Architectures, and Other Beasts

- A specification or design document must be understandable not only to its author.
  - Standard ways are needed to unambiguously express common relationships in a system.

- Standard notations exist, some of which we will discuss in class
  - Diagrams, description languages, etc.
  - You will be expected to use those in your LCA documents.

03 Jul 2006 CSE403, Summer'06, Lecture06

LCA Milestone Deliverables – in a Bit More Detail (1/3)

Detailed requirements specification document
- From the point of view of the customer
- Techniques: use cases, commonality and variability analysis, prototyping

Detailed architecture document
- Technical, reasonably detailed description of:
  - system modules and interfaces between them
  - UIs
  - Identifies assumptions and high-risk areas and proposes realistic solutions and/or alternatives
  - Notations: state/dataflow diagrams, sequence diagrams, class diagrams; UML

03 Jul 2006 CSE403, Summer'06, Lecture06

LCA Milestone Deliverables – in a Bit More Detail (2/3)

Test plan document
- High level strategy: what will be tested, what won’t, and why
- How you plan to test your product in a disciplined way
  - If you can’t test it easily, something is likely wrong with the design.
- A set of actual test cases, derived based on an established methodology
  - Methodology: Structure/Function/Data/Platform/Operations (SFDPO)

Schedule and task assignments
- The actual plan of work
  - How specifically will the project be split into milestones?
  - Who will take on which piece of the puzzle?
- Tools of the trade: Microsoft Project; Excel

03 Jul 2006 CSE403, Summer'06, Lecture06

LCA Milestone Deliverables – in a Bit More Detail (3/3)

Presentation
- Before your (true or surrogate) customers
- Will allow more time to cover important aspects and for audience to dissect weaker points

03 Jul 2006 CSE403, Summer'06, Lecture06
Parallels between LCA Milestone Deliverables and Core LCA Elements

**Milestone Deliverables**
- Overview presentation
- Specification document
- Architecture document
- Team structure, schedule, task assignments, risks
- Test plan

**LCA Elements**
- Operational concept
- System requirements
- System architecture
- Lifecycle plan
- Feasibility rationale
- missing in Boehm’s LCA

Test Plan

- “Failing to plan is planning to fail.”
- Describes what you want to test and how you will do it
  - Must match the tests you will create (are creating)
- Follows a disciplined methodology (e.g., SFDPO)
  - Some types of tests may not fit nicely, so invent your own categories

**Issues to consider:**
- Do you have tests in each of the categories?
- Do your proposed tests cover all use cases?
- Which tests are critical / high priority?

Test Planning using the SFDPO Methodology

- **Structure** (what the product is):
  - What files does it have? Do I know anything about how it was built? Is it one program or many? What physical material comes with it? Can I test it module by module?
- **Function** (what the product does):
  - What are its functions? What kind of error handling does it do? What kind of user interface does it have? Does it do anything that is not visible to the user? How does it interface with the operating system?
- **Data** (what it processes):
  - What kinds of input does it process? What does its output look like? What kinds of modes or states can it be in? Does it come packaged with preset data? Is any of its input sensitive to timing or sequencing?
- **Platform** (what it depends upon):
  - What operating systems does it run on? Does the environment have to be configured in any special way? Does it depend on 3rd party components?
- **Operations** (how it will be used):
  - Who will use it? Where and how will they use it? What will they use it for? Are there certain things that users are more likely to do? Is there user data we could get to help make the tests more realistic?

What Comes Next?

- Requirements gathering techniques
- Architecture: descriptions, etc.
- Design: strategies, principles