Domains: ride share, course registration, ecommerce,

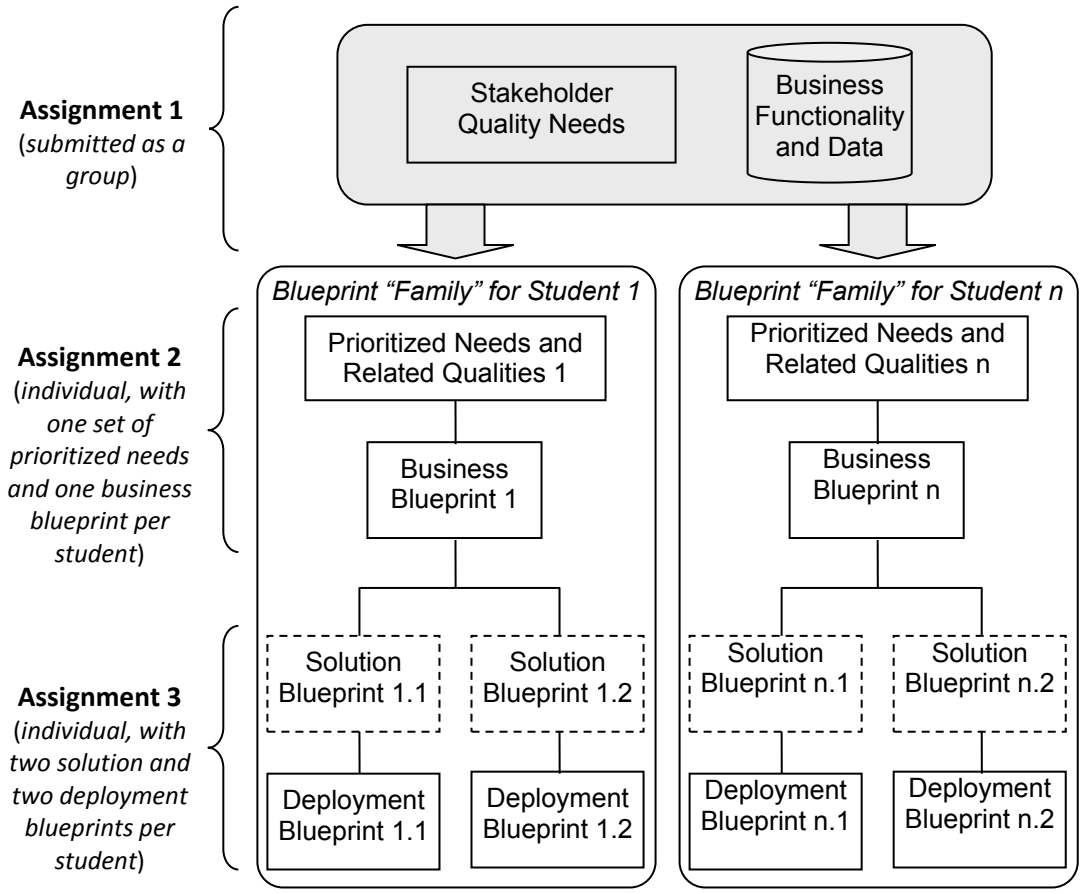
Couse Project Milestone 1
Domain Problem and Stakeholders
Software Architecture
Barber – Fall 2019

The objective of this course project is to define a software architecture for the problem domain of your choice. You will work in groups and a portion of your project report will be due at milestones throughout the semester. The course deliverables and respective grade percentages are as follows:

- **Course Project**
  - **Section 1: Domain Problem and Stakeholders**
    - What problem is being solved and who influences and will be impacted by the system?
  - **Section 2: Derivation and Evaluation of Business Blueprints**
    - Description of architectural components and connectors along with rationale for design decisions and evaluations to assess whether the blueprints meet expectations.
  - **Section 3: Derivation and Evaluation of Solution and Deployment Blueprints**
    - Description of architectural components and connectors along with rationale for design decisions and evaluations to assess whether the blueprints meet expectations.

In assignment milestone 1 of your course project, you will describe the domain problem and the stakeholders that influence design and development and will be impacted by the resulting system. Essentially, you will deliver Section 1 of the Course Project. Choose a domain with which your team is familiar and is sufficiently complex that you can *elaborate on at least 30 functions*.

IMPORTANT NOTE: Milestone 1 is the only team deliverable. Milestone 2 (Section 2) and Milestone 3 (Section 3) will be delivered on an individual basis. If at any point in the semester you decide changes to section 1 are required (i.e. requirements change), you must obtain agreement from all team members.



As with many software engineering artifacts, an architecture's primary uses are as

> (i) *a communication medium* -- to get everyone involved on the same page regarding system expectations and the vision about how the implementation will be mapped to the problem domain and
>
> (ii) *an analysis tool* -- to provide a model for conducting evaluations intended to predict how the system will perform compared with expectations.

As a basis for both uses, Assignment 1 requires that you understand the domain and categorize the stakeholders that will influence, impact, and/or contribute content to the architecture.

The following pages outline the information required for Milestone 1.

1. The Domain Problem and Stakeholders

1.1. Domain Description

Overall description for someone who is unfamiliar with the domain (no more than 1/2 page).

1.2. Envisioned System

Highlights of the envisioned system (no more than 1/2 page).

1.3. Stakeholders for the Architecture

1.3.1. Consumers

- For each consumer category…
  - What are their visions/expectations of the system (in the language of the stakeholder)?
  - What are their roles w.r.t. the development project (business expert, end-user, customer,…)?
  - What are their roles w.r.t. the domain?
  - In what part of the organization do they work?
  - What is the impact if their visions/expectations are not met?

1.3.2. Producers

- For each producer category…
  - What are their visions/expectations and/or what constraints do they place on the system (in the language of the stakeholder)?
  - What contribution will they make in designing, developing, or testing the system?
  - What is the impact if their visions, expectations, and constraints are not met?

1.4. Functional Requirements

1.4.1. Function Specifications

Identify at least 30 individual functions (e.g., “Perform degree check” and “Add a course” in the course registration domain)? For each function, provide the following:

- Name (start with a verb).
- Brief description.
- What data are input, output, or transformed by these functions?
  - From what function is an input received *or* is the input received from an external source.
  - To what function is an output sent *or* is the output sent to an external consumer.
- Who are the performers (and others involved) and how do they fit in the list of stakeholders?

- What resources are required to perform the functions aside from the data?
- Where is the function performed (organization and/or physical location)?
- What are the function's preconditions and post-conditions?
    - What triggers the execution of this function?
        - These may include events (e.g., "Perform degree check" cannot be performed until the advising office is opened) or the availability of data listed above (e.g., "Perform degree check" cannot be performed without the availability of the students record, which is input data to the "Perform degree check" function).
    - What determines when it has completed?

1.4.2. Scenario Specifications

Identify at least 10 scenarios in which the functions are executed:

- What sequence of functions (from Section 1.4) compose the scenario (e.g., scenario “Register for semester courses” is composed of “Perform degree check” and “Add a course” in that order)?
- What environment is needed to execute the scenario? Environment may be actual location or a context.
- What triggers the execution of this scenario and what determines when it has completed (i.e., preconditions and post-conditions)?

1.4.3. Essential Scenario

If you wanted to give someone a feel for the domain problem by selecting a *single* scenario, which one would you choose and why?

1.5. Qualities and Constraints

Given input from all known stakeholders (as described in Section 1.3 describe at least 10 system qualities and constraints derived from stakeholder visions and expectations. These qualities are often generalized as "ilities" or non-functional requirements including but not limited to reliability, usability, reusability, maintainability, flexibility, performance, and security. Examples of constraints include compatibility with legacy systems and schedule or budget constraints.

For each quality or constraint, provide the following:

- Description: Describe the quality/constraint in the context of the system being developed.
- Category: e.g., Does the quality fall under usability or performance?
- Stakeholder source: Which stakeholders motivated the quality/constraint?
- Scope: Does the quality apply to selected functions, selected scenarios, or the entire system?
- How should the quality be evaluated (as described by the stakeholder)?

1.6. Deployment Environment

Describe the environment(s) where the software will reside. These are also drawn from stakeholder expectations and requirements and may be specific to this project or enterprise-wide standards. Considerations include the following:

- Available computing platforms and resources on those platforms (e.g., CPU, memory, disk).
- Required connectivity.
- Operating system and runtime environment.
- Database.
- Implementation standards.
- Physical location.
- Third-party software (libraries, COTS, GOTS).

For each item, provide the following:

- Description.
- Stakeholder source.

Comments

- General comments:
  - When questioning what is expected in the assignment, follow the outline.
  - Everything you specify for Assignment 1 should be used at Assignment 2 and 3 as either blueprint content or rationale support.
  - Formatting matters!
- Regarding *Envisioned System*:
  - When describing the envisioned system, emphasize notable "features" that highlight how the system will be implemented, what is unique about the offering, why an existing system is being rewritten, etc.
  - If your envisioned system mentions specific design/architectural features, that's okay because those features may be notable from a marketing perspective and/or may be drivers for the development effort.
- Regarding *Stakeholders for the Architecture*:
  - Don't leave out any stakeholders. At a minimum, you'll have some set of end-users and business experts for *consumers* and project team members (e.g., developers, testers, system administrators) for *producers*.
    - Consumers receive, use or purchase the system.
    - Producers construct the system. You would not have producers executing domain functions (unless they were categorized as consumers as well).
  - You need the "risks of not satisfying" for prioritizing later.
- Regarding *Function Specifications*:
  - Thirty functions sound like a lot but will give you options when you allocate to architectural components.
  - Express functions as verb phrases.
  - Be careful about defining function specifications that suggest you already have a design in mind (e.g., function performed on "server"). In general, keep in mind the dividing line between domain and implementation – we don't want to reduce design options unless it's absolutely necessary and we want to be able to reuse across implementations. (Of course, your specific "dividing line" depends on your domain.)
  - "System" is typically not a good performer. The system is helping someone with a role in the domain execute a task.
  - The performer should never be a producer. Function performers should reference specific roles, thereby tying functional requirements back to specific end-users.
  - Dependency is an important consideration when identifying architectural components, and the dependency between the functions allocated to those functions suggest dependency between the respective components. If a function requires input data generated by another function, that implies a dependency.
- Regarding *Scenario Specifications*:
  - Scenarios are often named in terms of the objective.
  - There are many definitions of "scenario," but as described in this class, scenarios are *sequences of functions*. All your scenarios should reference only the functions you defined.
    - A function can be referenced in more than one scenario.
    - All functions should be referenced in at least one scenario.

    - People interpret "scenario environment" differently -- could be a specific location, a general description of the surroundings, a characterization of location (e.g., accident on freeway during rush hour), or a description of the conditions under which the scenario makes sense. If there is nothing notable about the environment, say so.

- Regarding *Essential Scenario*:
    - In addition to being a high-level functional description, the “essential scenario” is also a scoping mechanism (i.e., functional context for the development effort).

- Regarding *Qualities and Constraints*:
    - Be specific about the meaning of a given quality in the context of the system being developed (what does “reliability” mean in this context?).
    - Make a clear connection between stakeholder and qualities. The “impact if their visions/expectations are not met” is important for prioritizing during architecture derivation.
    - Similarly, the source of qualities/constraints should be specific stakeholders (both consumers and producers) -- you want to know whom to negotiate with when it comes time for tradeoffs.
    - Be explicit for scope. As with every piece of information in this specification, it can be used as justification in your derivation plan. It may also affect your evaluation. (Although, if it "applies to entire system," so be it, i.e., a cross-cutting "aspect.")
    - Be as specific as you can when describing how a quality should be evaluated.

- Regarding *Deployment Environment*:
    - My experience is that students have no problem describing the deployment environment, so feel free to provide details.