2QueueOOL4u

October 31, 2017

Architecture description and System design



Metadata:

Project name: NextText

Team name: 2QueueOOL4U

Team members:

James Mulvenna/100965629

Yue Zhang/100980408

Devon Plouffe/100715712

Architecture Description (4 pages; 2 text/ 2 diagram):

Over the course of our application advancement, our strategy was frequently modified due to our repeatedly changing conditions. We ran into multiple architectural complications. One being Android mobile application research with regards to our proposed product. As it stands, we have used a collection of architectural styles as no one style best suited our system. Diverse non-functional/functional properties apply to our product, and for this reason particular imperative styles were used respectively.

During the D0 phase 2QueueOOL4U was most concerned with the practicability of the system, we assessed the inter-dependent components and conducted research to determine how achievable our proposed system was. Fortunately, we found in the investigation period that designing our proposed application on iOS would be near unattainable by cause of Apples restrictions on message automation. Despite running into arguably our most important issue, we made the decision as a team to change platforms, as Androids development guidelines solved this obstacle.

In totality, at this moment our product consists of numerous architectural styles including but not limited to: object-oriented, client-server, mobile code, and finally event-driven. Although at this moment these architectural styles apply to our system, it would be naive of 2QueueOOL4U to declare these as the only styles. Realizing the remaining term of development, perhaps a new style might be used to account for outstanding system requirements.

Object-oriented

Client-server

Mobile code (Weather)

Event Driven

* Identify, describe, and justify the architecture of your project
* It is expected that a heterogeneous set of styles and patterns may best suit your project
* The outcome of the Architecture documentation is a system architecture that supports the functional goals and non-functional attributes of your project

*What are the functional properties of your system?*

1. The system should have the ability to save user information
2. The system should be able to import users contacts
3. The system should be able to access conditions
4. The system should list queued messages
5. The system should be able to add new queued messages
6. The system should be able to modify existing queued messages
7. The system should respect user privacy and authorization

*What are the non-functional properties your system needs to support?*

1. 1.1 User information will have the ability to be modified
2. 2.1 Native contacts, Facebook contacts and E-mails contacts will be able to be imported
3. 3.1 Time based services will be used, location based services (API) will be used, weather information (API) will be relayed incrementally
4. 4.1 A list of stored messages will be visual and accessible to the user
5. 5.1 User will have the ability to choose their contact and condition in an “add” interface
6. 7.1 The application will respect the permissions the user has set

System Design (4pages; 2 text/ 2 diagram):

* Describe the design of your system, such that a programmer could implement some subset of the system and integrate it
* Clear description of the structure of the components and its externally visible interfaces
* Document why you selected your design
* Compare the applicability of your design compared to alternative designs in this discussion
* Reference descriptions in architecture of important patterns, classes abstractions, and data structures/algorithms that are critical to the successful implementation of your system
* At minimum, inclue a class diagram that shows all classes and public API
* Include a sequence diagram (optional) that captures how your system behaves for each scenario from the initial proposal
* Clarify where classes reside (client, server) as well as any external API your system will use
* Analysis of how your design minimizes coupling and accommodates changing requirements is required
* Discuss how your design would support or inhibit changing requirements
* Identify a way you think your system may need to evolve in the future and describe how your project’s design would support these changes

Task Overview (1 page):

Oral component (Prepared to defend system design, asked how design could adapt to specific constraint, “you must now support X, how would you do that?)

**Page overview:**

1. Page 2; metadata
2. Pages 3-6; architectural description
3. Pages 7-10; system design
4. Page 11; assignment of tasks for each team member

Worth 30% of final grade, written is worth 60% and oral is worth 40%.