CS 446: Software Design and Architecture

Michael Noukhovitch

Spring 2016, University of Waterloo

Notes written from Victoria Sakhini's lectures.

Contents

1 Mobile Application **3**

1.1 Overview . . . 3

1.2 Design Considerations . . . 3

1.2.1 Client Type . . . 3

1.2.2 Devices to Support . . . 3

1.2.3 Connectivity . . . 3

1.2.4 Device Constraints . . . 3

1.2.5 Architecture . . . 4

1.3 Design Issues . . . 4

1.3.1 Authentication/Authorization . . . 4

1.3.2 Caching . . . 4

1.3.3 Communicaion . . . 4

1.3.4 Configuration Management . . . 5

1.3.5 Data Access . . . 5

1.3.6 Device Specifics . . . 5

1.3.7 Exception Management . . . 5

1.3.8 Logging . . . 5

1.3.9 Power Management . . . 6

1.3.10 Synchronization . . . 6

1.3.11 Testing . . . 6

1.3.12 UI . . . 6

1.3.13 Validation . . . 6

2 Software Architecture **7**

2.1 Definition . . . 7

1 Mobile Application

1.1 Overview

A mobile application is structured of mulitple layers: **presentation**, **business**, and **data**.

1.2 Design Considerations

1.2.1 Client Type

Rich local processing required, must work in ocassionally connected scenario

Thin can depend on server processing and will always be fully connected

Rich Internet Application requires a rich UI and only limited access to local resources (+ maybe portably to other platforms)

1.2.2 Devices to Support

Consider

- screen size and resolution
- cpu power
- memory and storage space
- dev tool availability
- user requirements, org constraints
- specific hardware requirements

1.2.3 Connectivity

If internet access is required, plan for intermittent or unavailable network connection

- caching
- state management
- batch communications

1.2.4 Device Constraints

Think of platform constraints, mainly:

- memory
- battery life
  - processing requirements
  - backlighting
  - memory I/O

  - wireless connections
- responsiveness of design
- security
- network bandwidth

1.2.5 Architecture

- layered architecture (multiple layers can be on device)
- reuse and maintainability
- smallest footprint possible

1.3 Design Issues

1.3.1 Authentication/Authorization

- security and reliability
- think about more than single user

1.3.2 Caching

- improve performance
- support offline work
- decide on what to cache based on limited resources

lazy acquisition defer acquiring resources as long as possible

1.3.3 Communicaion

- wifi, wired, bluetooth
- secure communication
- wireless is unreliable

active object support async processing by encapsulating service request and completion response

communicator encapsulate internal details of communication

entity translator transforms message data types into business types for requests and reverses for responses

reliable sessions end to end reliable message transfer

1.3.4 Configuration Management

- how to handle device resets
- how to allow configuration (OTA, from some host?)

1.3.5 Data Access

- low bandwidth
- high latency
- intermittent connectivity

active record include data access object within domain entry

data transfer object object storing data transported between processes, reducing method calls

domain model business objects that represent entities in a domain and relationships between them

transaction script organize logic for each transaction in a single procedure, making calls directory to DB (or through wrapper)

1.3.6 Device Specifics

- screen size
- orientation
- memory, storage space
- network bandwidth
- connectiviy
- OS
- hardware constraints

1.3.7 Exception Management

- prevent sensitive exception details from being revealed to the user
- improve application robustness
- keep application in consistent state after an error

1.3.8 Logging

- log only essentials because of size constraints
- may need to synchronize logs with server

1.3.9 Power Management

- power is limiting design factor
- research communication protocols and their effect on battery life

1.3.10 Synchronization

- secure communications OTA
- handle connection interruptions

sync design pattern component installed on device tracks changes to data and tells server when connected

1.3.11 Testing

Mobile debugging is costly, so make sure to invest heavily in testing beforehandas emulators may not be adequate to simulate a device in debugging.

1.3.12 UI

- build mobile first
- design for simplicity
- design around blocking operations (since user can only see once screen at a time)

application controller object that contains all the flow logic

MVC separate the data, presentation, and actions into three separate classes

- model manages behaviour and data (logic)
- view manages information display
- controller manages user input

MVP same as MVC but presenter manages presentation logic and interaction between view and model

pagination separate content into individual pages

1.3.13 Validation

- protect device and application
- improve usability
- validate client-side and server-side

2 Software Architecture

2.1 Definition

No perfect definition but AINSI says

> recommended practice as the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution.