Software Architecture. Lab Work 1

Objective

The aim of this lab work is to acquire the skill of writing architectural decision records and performing a back-of-the-envelope estimation with the help of architectural katas.

Task

1. Pick the architectural assignment (kata) according to your variant number (see Appendix 1.)
2. Write **seven or more ADRs** for your kata, each comprised of these sections: *Title*, *Status*, *Context*, *Decision*, *Consequences*, and *Notes*. **At least two** of them have to include the *Compliance* (or *Governance*) section, and **at least one** must have the *Alternatives* section.
3. Do the **back of the envelope analysis**. If necessary, augment the kata task with relevant data and information considering the domain.
4. Write a report; describe all of the steps you went through to achieve the results.

Theory Recap

Architectural Katas

An architectural kata is a structured exercise or practice used by software architects and developers to enhance their skills in designing software systems. It emphasizes learning by making design decisions, even if those decisions might not be perfect at first. Katas often use hypothetical scenarios that simulate real-world challenges.

Architectural Decision Records (ADRs)

Thanks to the Second Law of Software Architecture, we know we need a way to capture not just the decision, but the reason we made it. Architects use *architectural decision records* to record such decisions because it gives us a specific template to work with. An ADR is a document that describes a specific architectural decision. You write one for every architectural decision you make. Over time, they build up into an architectural decision log. ADRs are the documentation that supports the *architectural decisions* dimension.

The basic structure of an ADR consists of five main sections: *Title*, *Status*, *Context*, *Decision*, and *Consequences*. We usually add two additional sections as part of the basic structure: *Compliance* (or *Governance*) and *Notes*. This basic structure can be extended to include any other section deemed needed, providing the template is kept both consistent and concise. A good example of this might be to add an *Alternatives* section if necessary to provide an analysis of all the other possible alternative solutions.

The ***Title*** section describes the decision. It should be meaningful, yet concise, and crafted carefully. A good title makes it easy to figure out what the ADR is about. The title should consist mostly of nouns. The title should start with a number — it is recommended to use three digits, with leading zeros where needed.

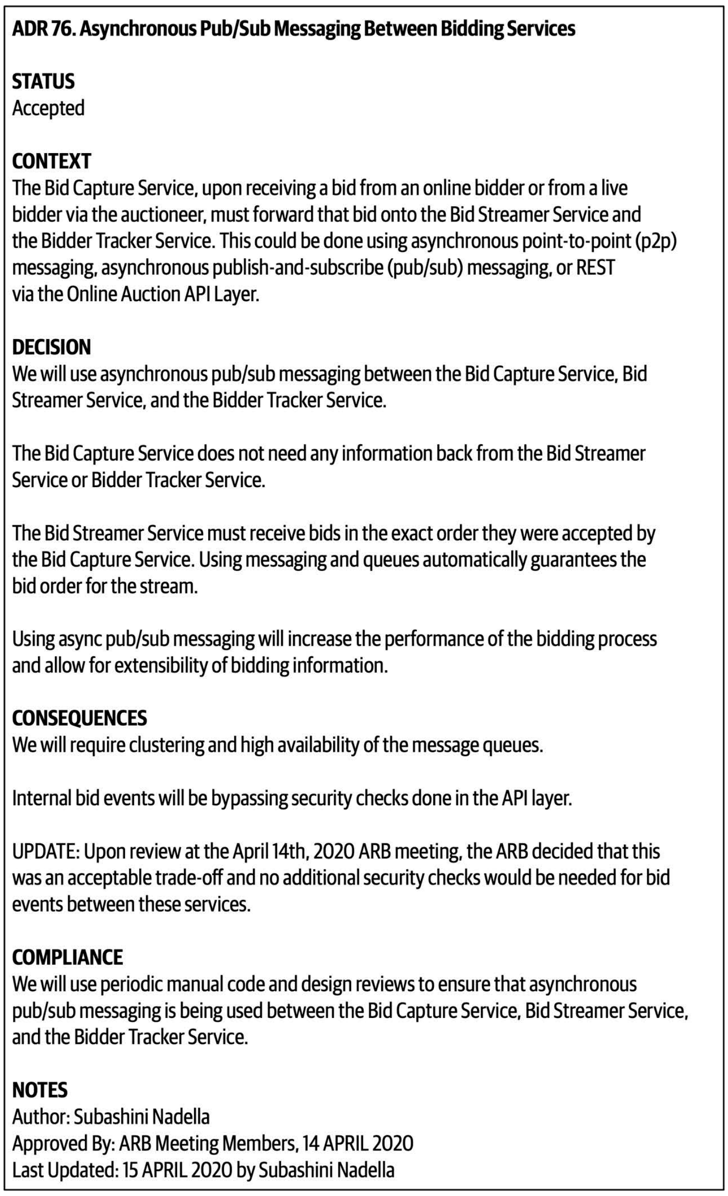
The status section communicates where the team stands on the decision itself. The status of an ADR can be marked as *Proposed*, *Accepted*, *Superseded,* or *Request for Comments* (or *RFC*). *Proposed* status means the decision must be approved by either a higher-level decision maker or some sort of architectural governance body (such as an architecture review board). *Accepted* status means the decision has been approved and is ready for implementation. A status of *Superseded* means the decision has been changed and superseded by another ADR. Superseded status always assumes the prior ADR status was accepted. When the architect wants to validate various assumptions and assertions with a larger audience of stakeholders, they may send out a draft ADR for comments, which then has the *Request for Comments* (or *RFC*) status.

The ***Context*** section of an ADR specifies the forces at play. In other words, "what situation is forcing me to make this decision?" This section of the ADR allows the architect to describe the specific situation or issue and concisely elaborate on the possible alternatives. This section is your place to explain the circumstances that drove you to make the decision the ADR is capturing. It should also capture any and all factors that influenced your decision.

If an architect is required to document the analysis of each alternative in detail, then an additional ***Alternatives*** section can be added to the ADR rather than adding that analysis to the *Context* section. Using a separate section to detail the trade-off analysis delineates it cleanly and avoids cluttering the Context section.

The ***Decision*** section of the ADR contains the architecture decision, along with a full justification for the decision. It should be written in a very affirmative, commanding voice rather than a passive one.

The ***Consequences*** section of an ADR is another very powerful section. This section documents the overall impact of an architecture decision. Every architecture decision an architect makes has some sort of impact, both good and bad. Having to specify the impact of an architecture decision forces the architect to think about whether those impacts outweigh the benefits of the decision.



ADR 76. Asynchronous Pub/Sub Messaging Between Bidding Services

STATUS
Accepted

CONTEXT
The Bid Capture Service, upon receiving a bid from an online bidder or from a live bidder via the auctioneer, must forward that bid onto the Bid Streamer Service and the Bidder Tracker Service. This could be done using asynchronous point-to-point (p2p) messaging, asynchronous publish-and-subscribe (pub/sub) messaging, or REST via the Online Auction API Layer.

DECISION
We will use asynchronous pub/sub messaging between the Bid Capture Service, Bid Streamer Service, and the Bidder Tracker Service.

The Bid Capture Service does not need any information back from the Bid Streamer Service or Bidder Tracker Service.

The Bid Streamer Service must receive bids in the exact order they were accepted by the Bid Capture Service. Using messaging and queues automatically guarantees the bid order for the stream.

Using async pub/sub messaging will increase the performance of the bidding process and allow for extensibility of bidding information.

CONSEQUENCES
We will require clustering and high availability of the message queues.

Internal bid events will be bypassing security checks done in the API layer.

UPDATE: Upon review at the April 14th, 2020 ARB meeting, the ARB decided that this was an acceptable trade-off and no additional security checks would be needed for bid events between these services.

COMPLIANCE
We will use periodic manual code and design reviews to ensure that asynchronous pub/sub messaging is being used between the Bid Capture Service, Bid Streamer Service, and the Bidder Tracker Service.

NOTES
Author: Subashini Nadella
Approved By: ARB Meeting Members, 14 APRIL 2020
Last Updated: 15 APRIL 2020 by Subashini Nadella

Figure 1 — ADR Example

The ***Compliance*** (or ***Governance***) section forces the architect to think about how the architecture decision will be measured and governed from a compliance perspective.

The ***Notes*** section includes various metadata about the ADR (*Original author, Approval date, Approved by, Superseded date, Last modified date, Modified by, Last modification*).

Back-Of-The-Envelope Estimation

A back-of-the-envelope (BOTE) estimation in the context of software architecture refers to a quick, rough calculation or approximation used to evaluate the feasibility, scale, or resources required for a software system or a specific architectural decision. This type of estimation helps software architects make preliminary assessments without getting bogged down in extensive details or formal analysis.

To successfully perform a back-of-the-envelope estimation, the following concepts should be well understood: power of two, latency numbers every programmer should know, and availability numbers.

Power of two. Although data volume can become enormous when dealing with distributed systems, calculation all boils down to the basics. To obtain correct calculations, it is critical to know the data volume unit using the power of 2.

Table 1 — Power of two

Power	Approx. value	Full name	Short name
10	1 Thousand	1 Kilobyte	1 KB
20	1 Million	1 Megabyte	1 MB
30	1 Billion	1 Gigabyte	1 GB
40	1 Trillion	1 Terabyte	1 TB
50	1 Quadrillion	1 Petabyte	1 PB

Latency numbers every programmer should know

Dr. Dean from Google reveals the length of typical computer operations in 2010. Some numbers are outdated as computers become faster and more powerful. However, those numbers should still be able to give us an idea of the fastness and slowness of different computer operations.

Table 2 — Latency numbers

Operation name	Time
L1 cache reference	0.5 ns
Branch mispredict	5 ns
L2 cache reference	7 ns
Mutex lock/unlock	100 ns
Main memory reference	100 ns
Compress 1K bytes with Zippy	10,000 ns = 10 µs
Send 2K bytes over 1 Gbps network	20,000 ns = 20 µs
Read 1 MB sequentially from memory	250,000 ns = 250 µs
Round trip within the same datacenter	500,000 ns = 500 µs
Disk seek	10,000,000 ns = 10 ms
Read 1 MB sequentially from the network	10,000,000 ns = 10 ms
Read 1 MB sequentially from disk	30,000,000 ns = 30 ms
Send packet CA (California) ➤ Netherlands ➤ CA	150,000,000 ns = 150 ms

Availability numbers

High availability is the ability of a system to be continuously operational for a desirably long period of time. High availability is measured as a percentage, with 100% means a service that has 0 downtime. Most services fall between 99% and 100%.

A service level agreement (SLA) is a commonly used term for service providers. This is an agreement between you (the service provider) and your customer, and this agreement formally defines the level of uptime your service will deliver. Cloud providers Amazon, Google, and Microsoft set their SLAs at 99.9% or above. Uptime is traditionally measured in nines. The more the nines, the better. As shown in the table, the number of nines correlate to the expected system downtime.

Table 3 — Availability numbers

Availability %	**Downtime per day**	**Downtime per year**
99%	14.40 minutes	3.65 days
99.9%	1.44 minutes	8.77 hours
99.99%	8.64 seconds	52.60 minutes
99.999%	864.00 milliseconds	5.26 minutes
99.9999%	86.40 milliseconds	31.56 seconds

Example

Estimate Twitter (now “X”) Query per second (QPS) and storage requirements.

Please note the following numbers are for this exercise only as they are not real numbers from Twitter.

Assumptions:

- 300 million monthly active users.
- 50% of users use Twitter daily.
- Users post 2 tweets per day on average.
- 10% of tweets contain media.
- Data is stored for 5 years.

Estimations:

Query per second (QPS) estimate:

- Daily active users (DAU) = 300 million · 50% = 150 million
- Tweets QPS = 150 million · 2 tweets / 24 hour / 3600 seconds = ~3500
- Peek QPS = 2 · QPS = ~7000

(We will only estimate media storage here.)

Average tweet size:

- tweet_id: 64 bytes
- text: 140 bytes
- media :1 MB

Media storage:

- 150 million · 2 · 10% · 1 MB = 30 TB per day
- 5-year media storage: 30 TB · 365 · 5 = ~55 PB

Submission

Push the report to your corporate Git repository and message your professor.

Appendix 1 (Architectural Katas)

There are 20 variants of architectural katas.

01 *Agile Dead Trees*
A publisher wants to unify its authoring Content Management System (CMS) and customer store experience, trying to get books published to customers as quickly as possible.

- Users: dozens of publisher employees, hundreds of authors, thousands/millions of customers
- Requirements:
    - authors publish chapters
    - reviewers see the chapters, make review comments, and notify authors on review
    - authors can reject proposed review changes
    - supports both copy and technical editing
    - customers can buy books (either e- form or dead trees form) online, including those available in 'beta'
    - publisher can push authors' chapters to those customers who bought the 'beta'
- Additional Context:
    - The business is driven to this decision because competitors have a similar offering.
    - Competition for authors is tight.
    - This is part of a long-term strategy to modernize the publishing aspects of the business.
    - Information needed to publish a book (distribution, royalties, marketing) comes from several disparate systems, ranging from emailed Excel spreadsheets to mainframe integration with the printing facility.

02 *All Stuff, No Cruft*
Conference organizer needs a management system for the conferences he runs for both speakers and attendees

- Users: hundreds of speakers, dozens of event staff, thousands of attendees
- Requirements:
    - attendees can access speaking schedule online, including room assignments
    - speakers can manage talks (enter, edit, modify)
    - attendees 'vote up/down' talks
    - organizer can notify attendees of schedule changes up-to-the-minute (if attendees opt in)
    - each conference (being a different subject) can be branded independently
    - speaker slides are accessible online only to attendees
    - evaluation system via web page, email, SMS, or phone
- Additional Context:
    - Conference runs across the US.
    - Very small support staff.
    - 'Bursty' traffic: extremely busy when conference is occurring.
    - Conference organizer wants to easily 'skin' the site for different technology offerings.

03 *Am.I.Sck*

Your company wants to build a software system supporting chat nurses (advice nurse) answering questions from customers about potential health problems.

- Users: 250+ nurses worldwide, hundreds to thousands of customers
- Requirements:
    - access patient medical histories
    - provide an service-level agreement on turnaround time for interaction
    - assist nurses in providing medical diagnosis
    - support nurses geographically divergent from clients
    - enable client customers to reach local medical staff (if necessary), contacting the local medical staff directly ahead of time (if necessary)
    - eventually enable parts of the system for direct client customer use
- Additional Context:
    - company is building software solutions in niche spaces like this one
    - agressive growth with 2nd round VC funding
    - fast time to market is overall company goal to capture market
    - our lawyers have determined that the conversation is not considered a medical record

04 *Check Your Work*

A university has greatly expanded its CS course and wants to be able to automate the grading of simple programming assignments.

- Users: 300+ students per year, plus staff and admin.
- Requirements:
    - students must be able to upload their source code, which will be run and graded
    - grades and runs must be persistent and audit-able
    - required plagiarism detection system involving comparing with other submissions and also submitting to a web-based service (TurnItIn).
    - integration required with the University's learning management system (LMS)
    - professor sets due date and time, after which submissions are rejected
    - students can submit as many attempts as they want to improve their grade
    - professors determine grading criteria, which may include metrics and/or tests
- Additional Context:
    - University's LMS is mainframe based and quite difficult to make changes to
    - grades are audited each year by state-based regulatory body
    - University has very little budget for IT as it is building a spare stadium for SportsBall
    - University has a record for highest-performing CS graduates in the country

05 *E(xperimental) College*

Local college now offers unique non-credit courses in addition to the usual grad/undergrad courses, and they need a registration and payment system

- Users: students, college users, admin, and accounting
- Requirements:
    - existing central student/class registry is NOT integratable--only https web form access allowed
    - accept payments
    - track registrations
    - non-credit registrations must be duplicated in central registry (but not by hand!)
    - students enter the central registry if their payment succeeds
    - updates/deletes from the central registry is okay as a manual process but preferably automated
    - payments can be credit card, bank account withdrawal, check or cash
    - course prereqs
    - students can request invoice/receipts and/or transcripts (which should be emailed)
    - multiple students register for multiple classes with one purchase
    - admins must see course lists, payments, but not student personal info
- Additional Context:
    - very little budget for IT
    - 'How do they register and pay?' was a neglected requirement for many months
    - hard deadline for class registration in 6 months

06 *Gird the Grid*

A company who builds management software for electrical grids needs to update their outdated software solution, and plan to sell their offering as a platform.

- Users: small to medium size market electic market companies, able to accommodate electrical grids from 100,000-1,900,000 electrical customers
- Requirements:
    - configurable for specific grid company characteristics (state, tax rates, etc.)
    - state-of-the-art user experience
    - dashboards with analytics reports with near-real time data from the grid
    - excellent reporting capabilities
    - sophisticated analytics-based engine to determine best throughput/money
    - administer through either desktop or mobile devices
    - security penetration attempt reporting
- Additional Context:
    - Four nines (99.99) reliability
    - turn-key deployment on remote sites
    - security is a first-class concern
    - company wants to shift from managing electrical grids to becoming a software reseller

07 *Going Green*

A large electronics store wants to get into the electronics recycling business and needs a new system to support it. Customers can send in their small personal electronic equipment (or use local kiosks at the mall) and possibly get money for their used equipment if it is in working condition.

- Users: Hundreds, hopefully thousands to millions
- Requirements:
  - Customers can get a quote for used personal electronic equipment (phones, cameras, etc.) either through the web or a kiosk at a mall.
  - Customers will receive a box in the mail, send in their electronic, and if it is in good working order receive a check.
  - Once the equipment is received, it is assessed (inspected) to determine if it can be either recycled (destroyed safely) or sold (eBay, etc.).
  - The company anticipates adding 5-10 new types of electronic that they will accept each month.
  - Each type of electronic has its own set of rules for quoting and assessment.
- Additional Context:
  - This is a highly competitive business and is a new line of business for us
  - If we haven't received a type of electronic equipment in a year we will remove it from our system
  - We need to maintain a list of electronic equipment we are willing to accept as it changes often
  - Each piece of equipment has it's own assessment (inspection) rules
  - We have the right to change the original quote to the customer if the product isn't in the condition they said it was

08 *Going...Going...Gone!*

An auction company wants to take their auctions online to a nationwide scale--customers choose the auction to participate in, wait until the auction begins, then bid as if they were there in the room, with the auctioneer

- Users: scale up to hundreds of participants (per auction), potentially up to thousands of participants, and as many simultaneous auctions as possible
- Requirements:
  - auctions must be categorized and 'discoverable'
  - auctions must be as real-time as possible
  - auctions must be mixed live and online
  - video stream of the action after the fact
  - handle money exchange
  - participants must be tracked via a reputation index
- Additional Context:
  - auction company is expanding aggressively by merging with smaller competitors
  - if nationwide auction is a success, replicate the model overseas
  - budget is not constrained--this is a strategic direction
  - company just exited a lawsuit where they settled a suit alleging fraud

09 *Hot Diggety Dog!*

Local hot dog stand merchant wants a point-of-sale system for his hot dog stand operators

- Users: fifty or so hot dog stand operators, thousands of customers in the local area (via social-media)
- Requirements:
  - must be lightweight in size--laptop is too unwieldy to use efficiently when making hot dogs on the street
  - allow for discounts
  - track sales by time and location
  - send inventory updates to mobile inventory-management staff (who drive to the location with supplies)
  - provide a social-media integration so customers can be notified when a hot dog stand is nearby
  - export information in format importable by accounting tools
- Additional Context:
  - forced into undertaking this development because the current hodgepodge ways of tracking sales requires too much manual effort
  - time to completion is important
  - building a solution that won't need replacement in 3 years is more important
  - no serious budget constraints

10 *I Can Haz Cheezborger?*

Client wants to create websites to follow Internet trends--as a new trend is identified, create a website highlighting and following it, and allowing users to interact with it

- Users: millions+ readers, thousands+ posters, dozens admins
- Requirements:
  - high SEO
  - easy for users to add content
  - easily mashable/clickable
  - reject inappropriate content
  - easy trend analysis
  - user forums
  - user moderators
  - ubiquitous accessibility
  - easy admin 'reach'/accessibility
- Additional Context:
  - VC funded startup
  - wants to harvest data across trends for deeper market analysis
  - in 10 years, wants to be a powerhouse site for Internet trends

11 *I'll Have the BLT*

A national sandwich shop wants to enable 'fax in your order' but over the Internet instead (in addition to their current fax-in service)

- Users: thousands, perhaps one day millions
- Requirements:
    - users will place their order, then be given a time to pick up their sandwich and directions to the shop (which must integrate with several external mapping services that include traffic information)
    - if the shop offers a delivery service, dispatch the driver with the sandwich to the user
    - mobile-device accessibility
    - offer national daily promotionals/specials
    - offer local daily promotionals/specials
    - accept payment online or in person/on delivery
- Additional Context:
    - Sandwich shops are franchised, each with a different owner.
    - Parent company has near-future plans to expand overseas.
    - Corporate goal is to hire inexpensive labor to maximize profit.

12 *Lights, Please*

A home electronics giant wants to build a system for home automation: turning lights on and off, locking and unlocking doors, remote camera observation, and future unspecified behavior.

- Users: each system will be sold to consumers (small families), but the company expects to sell thousands of these units in the first three years.
- Requirements:
    - the system must be as turnkey as possible, but be sold in modular units (camera, lock, thermostat, etc) for easy purchase
    - the units must be accessible over the Internet (for remote monitoring and access), and it is assumed the user will have an existing WiFi setup (router and connection) to tap into
    - customers can program the system to control the various modules according to their own needs.
    - the electrical engineering for the units will be taken care of by other groups, and the software protocols for controlling the modules is flexible, according to the needs/designs of your architecture. (They will handle implementing the module side of the protocol, once you have specified it to them.)
- Additional Context:
    - willing to invest a large sum to get this new line of business off the ground
    - collects data from customers who opt in to gather broader statistics
    - international company

13 *Make the Grade*

A very large and populous state would like a new system to support standardized testing across all public school systems grades 3-12.

- Users: 40,000+ students, 2000 graders, 50 administrators.
- Requirements:
    - Students will only be able to use the application within testing centers around the state, most of these will be in the schools, but not all of them
    - Students should be able to take a test, and the results eventually consolidated to a single location representing all of the test scores across the state (by school, teacher, and student).
    - Tests will be multiple choice, short answer, and essay.
    - The system should have a reporting system to know which students have taken the tests and what score they received.
    - Short answer and essay questions will be manually graded by teachers, who will then add the essay grades to the system.
- Additional Context:
    - A change approval processes involving three different government agencies is required for changes to the way student grades are kept to ensure security.
    - The state does not own its hosting center, but outsources it to a third party.
    - Project must defend its budget each fiscal year.

14 *Room with a View*

A large hotel reservation company wants to build the next generation hotel reservation and management system specifically tailored to high-end resorts and spas where guests can view and reserve specific rooms.

- Users: Guests (hundreds), hotel staff (less than 20)
- Requirements:
    - Registration can be made via web, mobile, phone call, or walk-in.
    - Guests have the ability to either book a type of room (standard, deluxe, or suite) or choose a specific room to stay in by viewing pictures of each room and its location in the hotel.
    - The system must be able to maintain room status (booked, available, ready to clean, etc.) as well as when the room will be needed next.
    - It must also have state-of-the-art housekeeping management functionality so that cleaning and maintenance staff can be directed to various rooms based on priority and reservation need using proprietary devices supplied by the reservation company attached to the cleaning carts.
    - Standard reservation functionality (e.g., payments, registration info, etc.) will be done by leveraging the existing reservations system.
    - The system will be web-based and hosted by the reservation company.
- Additional Context:
    - 'Peak season is quickly approaching, so the system must be ready quickly or we have to wait until next year!'
    - Company is also investing heavily in cutting edge technology like smart room locks that open via a cell phone
    - Only interested in the high-end market
    - Sales people have tremendous clout in the organization; people often scramble to make their promises true

15 *Sysop Squad*

An electronics giant needs a new trouble-ticket system for their customer-facing IT consultants (the Sysop Squad) in their stores nationwide

- Users: thousands of customers, hundreds of consultants, hundreds of store staff
- Requirements:
    - trouble tickets can be entered by either call-center receptionists, store staff, or customers online
    - tickets route to the appropriate consultant based on location, availability and skill
    - consultants should only need a mobile device
    - customers enter consultant evaluation after service
    - consultant tracks work performed in customer record(s) for future reference
- Additional Context:
    - uptime is critical to the company's reputation
    - the site's performance must degrade gracefully under heavy load
    - good routing of requests is critical to making a profit

16 *Tales Of A Fourth Grade*

Company wants to offer school districts comprehensive student management system as a service

- Users: faculty, staff and student parents
- Requirements:
    - track absences, tardies and excuses (entered by parents, faculty, or staff)
    - manage from 1000 to 1 million students
    - generate reports on student activities
    - be accessible from the playground
    - track student grades and assignments (completed and due)
    - parent-teacher forums
    - run as an SaaS system from a hosting center
    - must adhere to FERPA (Family Educational Rights and Privacy Act) security guidelines, found at [https://www2.ed.gov/policy/gen/guid/fpco/ferpa/index.html]
- Additional Context:
    - company plans to undertake an aggressive national sales campaign
    - current competitor damaged by data breach
    - new CIO
    - main marketing pitch is around is flexibility, configurability, and (recently added) security

17 *The Road Warrior*

A major travel agency wants to build the next generation online trip management dashboard to allow travelers to see all of their existing reservations organized by trip either online of through their mobile device.

- Users: 10,000+ registered users worldwide
- Requirements:
  - The system must interface with the agency's existing airline, hotel, and car rental interface system to automatically load reservations via frequent flier accounts, hotel point accounts, and car rental rewards accounts.
  - Customers should be able to add existing reservations manually as well.
  - Items in the dashboard should be able to be grouped by trip, and once the trip is complete, the items should automatically be removed from the dashboard.
  - Users should also be able to share their trip information by interfacing with standard social media sites.
  - Richest user interface possible across all deployment platforms
- Additional Context:
  - must integrate seamlessly with existing travel systems
  - partnership deals are being negotiated to create 'favored' vendors
  - must work internationally

18 *Where's Fluffymon?*

A service describing missing pets, pet rewards (brokered/managed by the service), and location data points (GPS) of pet sightings using augmented reality to overlay last-seen pet locations

- Users: dozens of missing pet owners, hundreds of 'spotters' (initially), broader depending on rollout success
- Requirements:
  - users interested in finding pets register on the site
  - anyone can see a list of pets missing near to their location
  - pet finders can post 'pet found' messages (with mandatory photo proof) and collect rewards on confirmation from pet owners
  - users can comment on pet missing entries, offering data points (sighted, area checked with no results, etc)
  - mobile device accessibility
- Additional Context:
  - one of a host of AR services being launched by parent company
  - local scalability (per-city), but possibly scaling out to other cities
  - company wants to create a larger social community around pets
  - potential ad revenue from partners like pet stores have the potential to make millions

19 *World of Webcraft*

An entertainment company wants a first-person shooter MMORPG game delivered through the browser

- Users: millions+ (they hope)
- Requirements:
    - users choose the map they want to play in, then duke it out with randomly-selected players of roughly equal skill
    - players can 'trick out' their characters with skills/equipment by sending the company money
    - usable from any 'modern' Web browser
    - full-immersion experience (sound, graphics, etc)
    - players can 'chat' (trash talk) to other players, but only those 'within range'
    - players can create invitation-only tournaments
    - guests can observe all the players without being seen/interact (ghosts)
    - players can create new maps, weapons, and rules
- Additional Context:
    - gaming performance is high priority, but scalability even more so
    - if this game is successful, the company plans to extract the good bits as a framework they can sell
    - biggest expense is the highly skilled artists creating the visual look and feel

20 *You Look Good In Print*

A local copy shop chain wants to offer its customers an 'all-in-one' computing experience: document creation, editing, storage, and printing

- Users: initially, thousands in the local city, but potentially millions if the demand grows
- Requirements:
    - browser-based or delivered documents
    - word processing
    - presentations
    - document templates (as start points)
    - versioning
    - print scheduling
    - automatic and manual payment for storage, printing, etc.
- Additional Context:
    - Main reason for this initiative is better customer engagement and loyalty
    - for historical reasons, operations is handled by another company and isn't very responsive