Worksheet 1: Software Architecture

Updated: 24th July, 2019

For this exercise, it may be a good idea to work in groups if you can.

1. Architectural Mistakes

Suggest what might be problematic about the following architectural decisions:

(a) An airline in-flight entertainment system that allows passengers to extend the system's functionality by downloading custom plugins/apps/etc.

(b) A video-editing application that connects to a remote database in order to run all video encoding/editing operations in SQL.

(c) A social media app written in C.

(d) A government tax calculation app written using the Windows Presentation Foundation GUI library.

2. Architectural Anti-Patterns

Briefly research the following terms: "stovepipe system" and "vendor lock-in". For each case, what is the actual technical problem that it poses?

3. Driverless Taxis

Say you want to start up a driverless-taxi service. The system must be fully automated, with no humans involved other than the passengers, who must request and pay for a taxi beforehand. The system handles the payment, and allocates a car to be sent to pick up each group of paying passengers.

Your plan so far is to purchase fifty existing driverless cars, and potentially modify them as needed. As they are currently designed, each of these cars can navigate on its own from point A to point B, with the passenger entering the desired destination. Each car contains a computer to handle all driving and navigation. However, since the cars were not originally designed to be networked, no real attention has gone into making this system secure.

Nonetheless, security must now be a crucial focus of this project. The cars will now need to be networked in some way, and it would be catastrophic if an attacker were to gain remote access to them.

Propose a suitable architecture for this new system, and explain your choices.

Note: While there is no formal diagrammatic notation for representing architecture, diagrams do help. You can represent different parts of the system as nodes, and show the information exchanged between them as arrows.

4. Survey System

For planning purposes, government agencies often conduct surveys of residents and organisations. (A census is the most notable example of this, but there are many other more frequent and more narrowly-targetted surveys too.)

There are several aspects to this:

- The system must send out notifications to people to complete the survey. It must store responses securely. It must also present the survey data to agency employees upon request (with nice graphical charts).
- We would like to run a single instance of the survey software, as this is generally more efficient.

  However, there are many different kinds of surveys, targetting different groups of people and organisations. Different agencies may want to provide some code that changes the way that the survey is presented to users, to accommodate their specific needs.
- Electronic surveys are cheaper than paper forms, but the government cannot require everyone to have a computer, so it must provide a paper-based option as a backup. The paper forms will have to be mailed to the agency and the data entered into the system by an agency employee.

  The electronic and paper surveys have to be entered in different ways. When a user completes a survey electronically, they log in with their government username and password, and are then guided through the process with lots of explanation.

  However, when an employee enters details from a paper-based form, they (a) won't know the person's password, and (b) will work more efficiently with spreadsheet-style interface rather than a questionnaire. Say that the government agencies also use a standardised Windows environment.

5. E-Voting (Optional!)

Say you wanted to conduct the next election electronically, while maintaining the same standards and protections currently in place. (There is actual real-life interest in doing so, because the results would be known much more quickly.)

What kind of architecture would you use, and why? Also give an example of an architecture that you *would not* use.

This is a complex and open-ended question, and there is a real possibility that there is no perfect answer. Here are some key issues:

- How will a person with no computing device and no ID vote?

  (Currently there are no ID requirements while voting. Voters must be enrolled to vote in a given electorate, but they can have "no fixed address".)
- How can other people audit an election? That is, how could they find out, *independently of the software itself*, whether all the votes were correctly recorded and counted?

(There's no requirement for speed in such an audit.)

- How can you guarantee a "secret ballot" (i.e. making it impossible to trace a vote back to a voter) while at the same time guaranteeing that the election result cannot be corrupted by people voting multiple times?

End of Worksheet