Software Engineering 1

Notes 2/1/2017

**Presentation 1:**

We draw number for time slot of presentation

Presentation is not graded

Purpose: Go through use cases

Little bit of introduction to project

Little bit of functionality

Description of some use cases

**Document Outline:**

Present the use cases that are least likely to be correct in order to receive input in order to better your use case, in other words the presentation is for others to comment on your use cases and help you

Introduction

Describe the need for your system, in terms that are understood by a non technical person. You don't have to say there is DB, or backend, running on the web. Describe what problem you really are solving.

Current System

Describe a manual system or if there is a system already then you can improve it. If you put the current system into the introduction, then skip this park

Proposed System

Overview: The system that you propose, how will it meet the problem we described in the introduction. Tell a story that helps the audience understand what would go into the system.

Functional Requirements

Do a List of all functions that the system will provide. Organize however you want.

Nonfunctional Requirements:

describe in general terms how the system, what requirements the system has to meet but not in terms of functions. Need to have some, but won't cross check whether those are valid or not.

Constraints:

Similarly come up with some constraints.

Example: The system must be completed by March 17, 2017

Use Case Model:

Your use cases. Has two parts

Powerpoint 2: Use Case Diagram, like slide 9...essentially uml, but includes extend relationships and "include" relationships. You must have at least one of each, sommewhere in the use cases. However you may have more if you design a good system. When you turn in the document, you have to be convinced that your document has all "include" and "extend" relationships.

The challenge is: simplicity, clarity vs completeness.

If your use cases has a lot of extend/include relationships then the use case is too complex and break it up into more use cases.

If I read the use case 4 times and I don't understand it, then its a bad use case.

Use case has to be self explenatory.

ColectMoney

identify actors

entry conditions: invoked by "<use\_case>"