

EECE 310 – Software Engineering

Summary

Agenda

- Friday: Guest Speaker
- Evaluations
- Final Student Presentations (15 & 28)
- Course Summary and Q&A

Friday: Guest Speaker

- Title: **Upgradeability of 3d Party Libraries: A Security Perspective**
- Speaker: Prof. Arie van Deursen
- Part of final exam. Make sure you attend and ask questions!

Admin

- Course Eval: <https://eval.olt.ubc.ca/apsc>
- TA Eval: in-class

Instructor/Course Eval

- Go to <https://eval.olt.ubc.ca/apsc>
- Sign in and provide feedback
- It means a lot to me and the department!

TA evaluation



- Saba Alimadadi (F)



- Keheliya Gallaba
- Dawood Al-Masslawi

Notes for final

- Similar to mid-term (multiple-choice, true/false, open questions)
- 40% of the total course grade
- What to study:
 - All material covered in class
 - Lecture notes, **and your own notes!**
 - Reading Material on Connect
- Bring UBC card to exam.

Course Summary

The purpose

- Software **engineering** is not just programming (although that's a huge part)
- It is the process, the techniques/tools, the teamwork and the product.

Example: Facebook

- First written in PHP using MySQL by an undergrad (now a very wealthy undergrad!).
- Scale challenges: data storage and serving pages to billions.
- Now write PHP that gets compiled to C/C++ (HipHop)
- Haystack high-performance photo storage/retrieval system
- Data mirrored on geographically distributed content distribution networks (CDN).
- **SE huge part of this:** feature gathering, testing, release engineering, engineering tools, and writing code.

<http://arstechnica.com/business/news/2012/04/exclusive-a-behind-the-scenes-look-at-facebook-release-engineering.ars>

What we looked at

- Process (Waterfall, especially Agile: Scrum, XP)
- Versioning (especially Git)
- Requirements (especially user stories)
- Architecture and Design
- Cost Estimation

What we looked at

- Testing
- Code quality: smells and refactoring
- Software Evolution and Maintenance
- UI Design
- Crosscutting concerns and AOP

Prospects for an engineering discipline of software

- Software 'engineering' is a young field
- Challenges:
 - Rapidly changing environments
 - Science and engineering must reinforce one another
- Goals
 - Awareness of best practices
 - Measure/Find/Adopt disciplines that work!
 - Specialize and standardize

Software Process

- structured set of activities to develop a software system
- define roles in the process
- facilitates organization and communication

Process models

- **Waterfall**
 - define requirements up-front
 - hierarchical and traditional management approach
 - suitable for large projects (?) and new teams
 - low trust
 - Add iterations with Spiral model
- **Agile**
 - short iterations, focus on communication and change



Waterfall: battleship,
protected against
everything that might
happen...

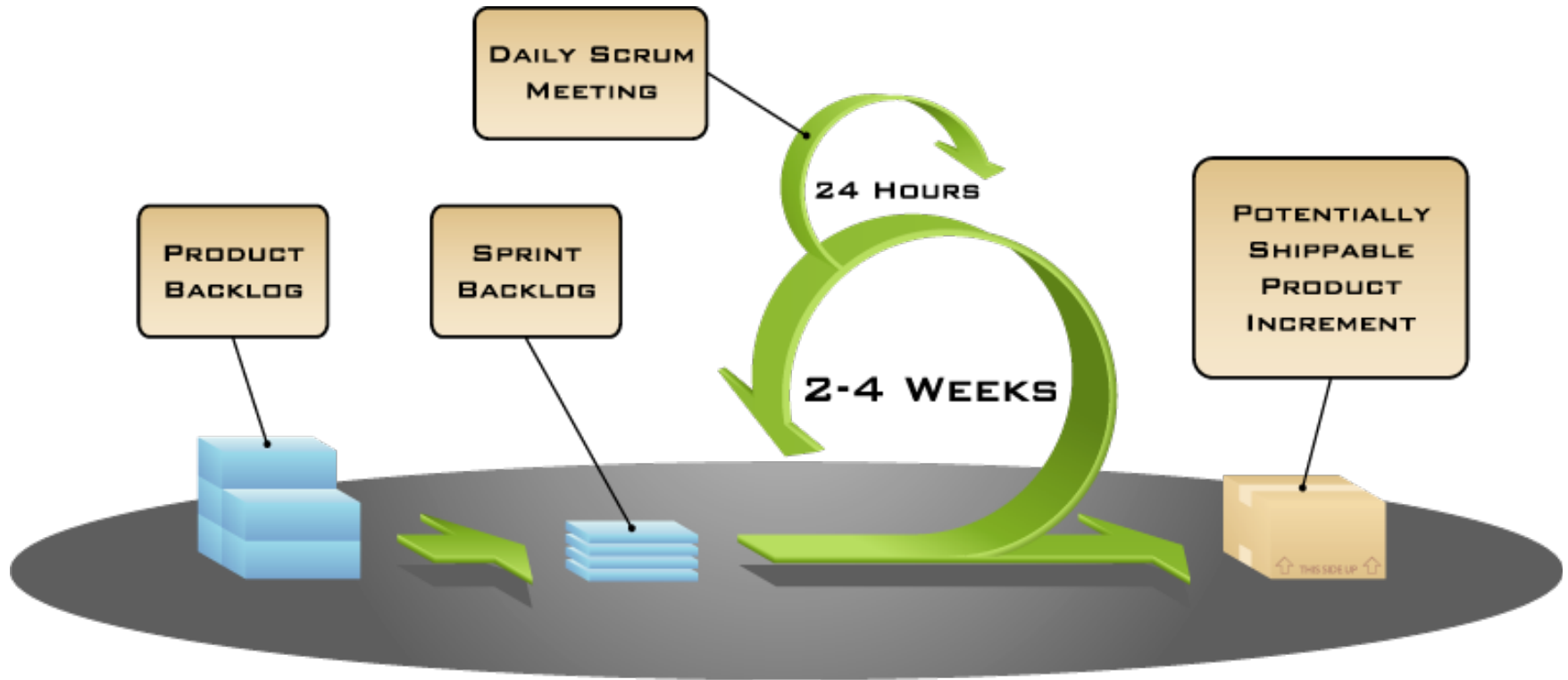
Agile: speedboat,
can change direction very
quickly



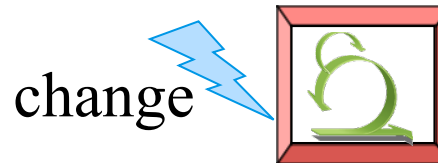
Agile software development

- People over process
- Software over documentation
- Communication over contract
- Adaptable over plan-driven
- But, requires trust and capable team
- But, might require constant refactoring
- Agile methods include Scrum and XP

Scrum



COPYRIGHT © 2005, MOUNTAIN GOAT SOFTWARE



Mountain Goat
Software, LLC

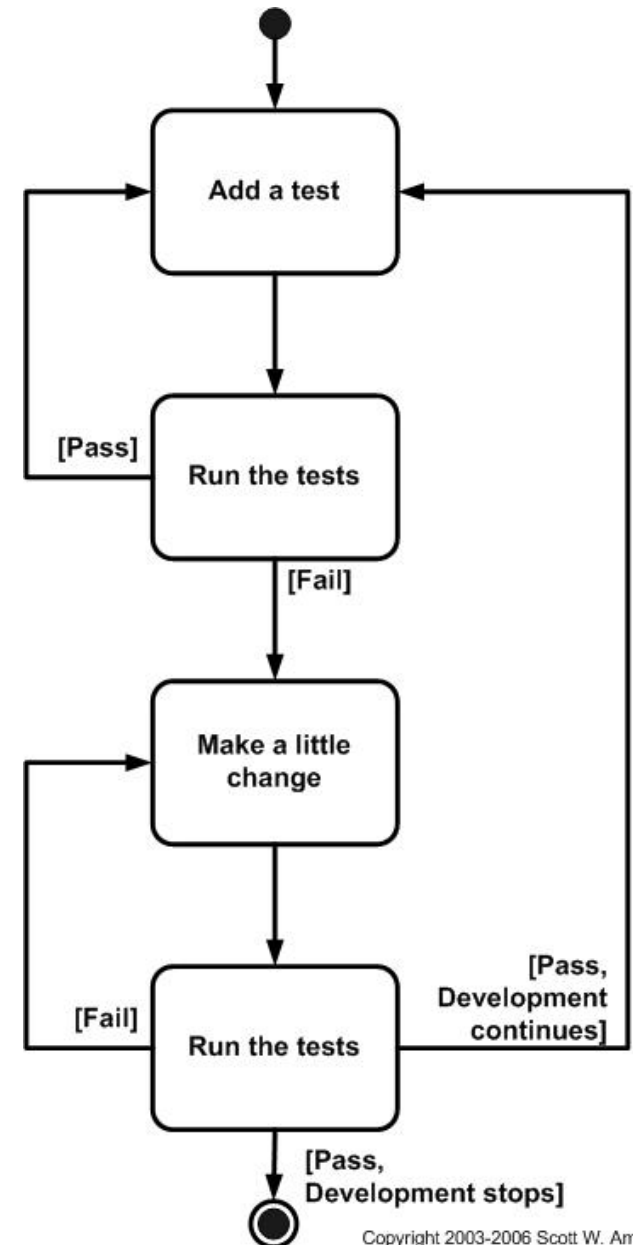
taken from www.mountaingoatsoftware.com (Mike Cohn)

eXtreme Programming

- User stories
- Unit Testing (xUnit)
- Test-driven development
- Continuous integration (eg. Jenkins)
- Pair programming
- Refactoring

Test-driven development

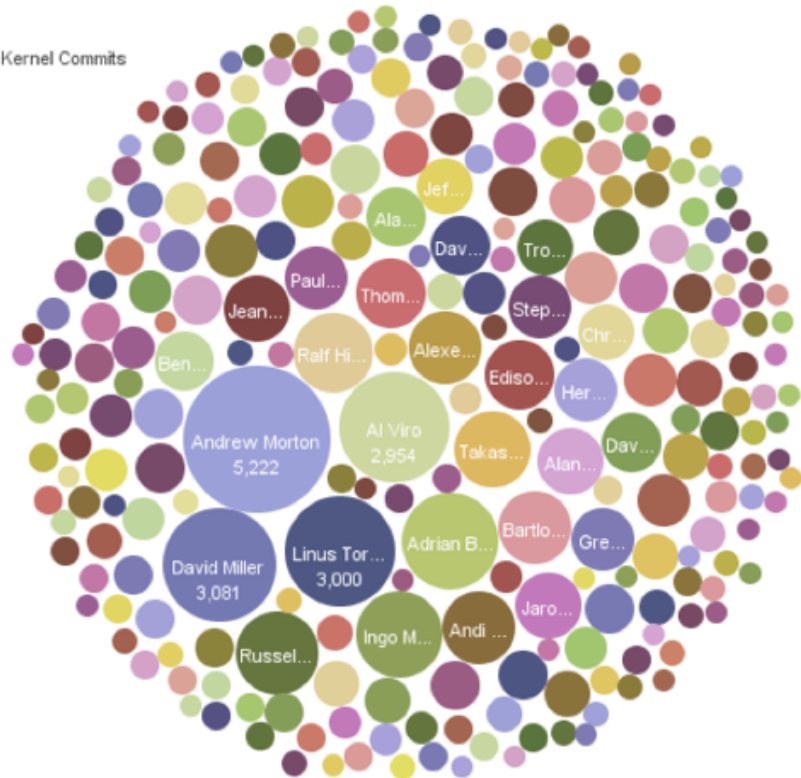
- Tests define units of work
- Tests document expected functionality
- Tests are contracts for methods
- Later additions can be “regression” tested
 - Failure indicates breakage
- Small steps



Version control

- Why is it useful?
- Challenges
 - Central vs. distributed VC
- Merge and conflict
- Workflows
- Branching styles

by Linux 2.6 Kernel Commits



To highlight or find
click or ctrl-click

Created on Many Eyes (<http://many-eyes.com>) ©

Requirements: user stories

- As a [type of user], I want/can/need/etc. [goal or need], so I can [reason]
- has a description of the story providing additional detail
- specifies **acceptance criteria** that defines what is meant for this feature to be DONE
- provides **estimate** of the required effort (story points)

Requirements: motivation

- Define *What* to build
- Define *Why* to build it
- Avoid describing *How* to build it
- Large source of costly errors
- Functional requirements linked to specific features of the system
- Non-functional requirements about wider system-wide properties (attributes, properties, constraints)

Requirements process

1. Elicit requirements by determining stakeholders, existing documents

- Extract explicit and implicit knowledge
- Use techniques like interviews and ethnographic analysis to do so.

2. Analyze and iterate

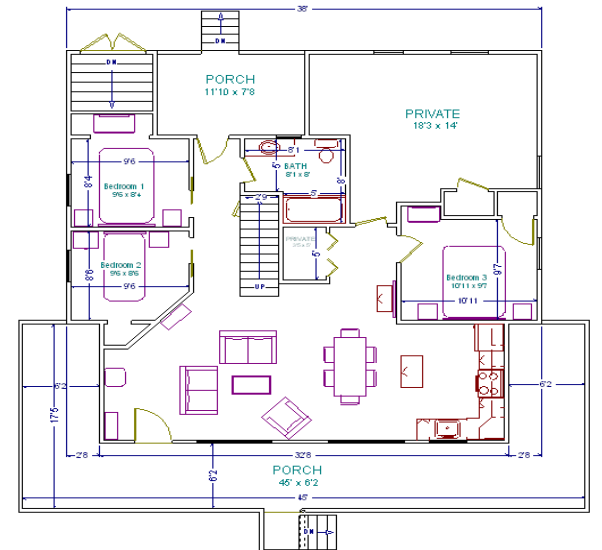
3. Specify for implementation phase

- SRS and use cases

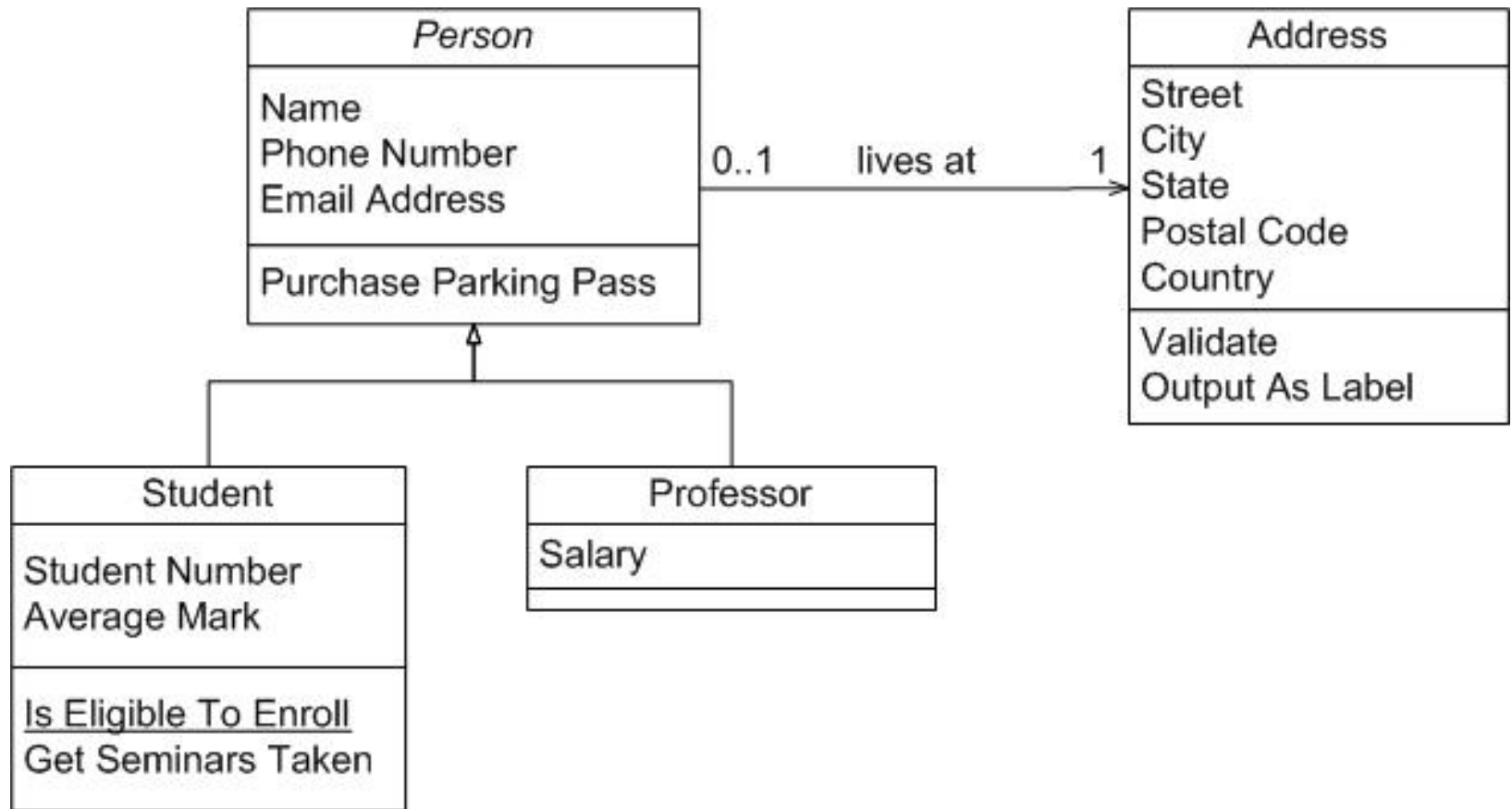
4. Validation: ensure requirements capture customer goals

Design

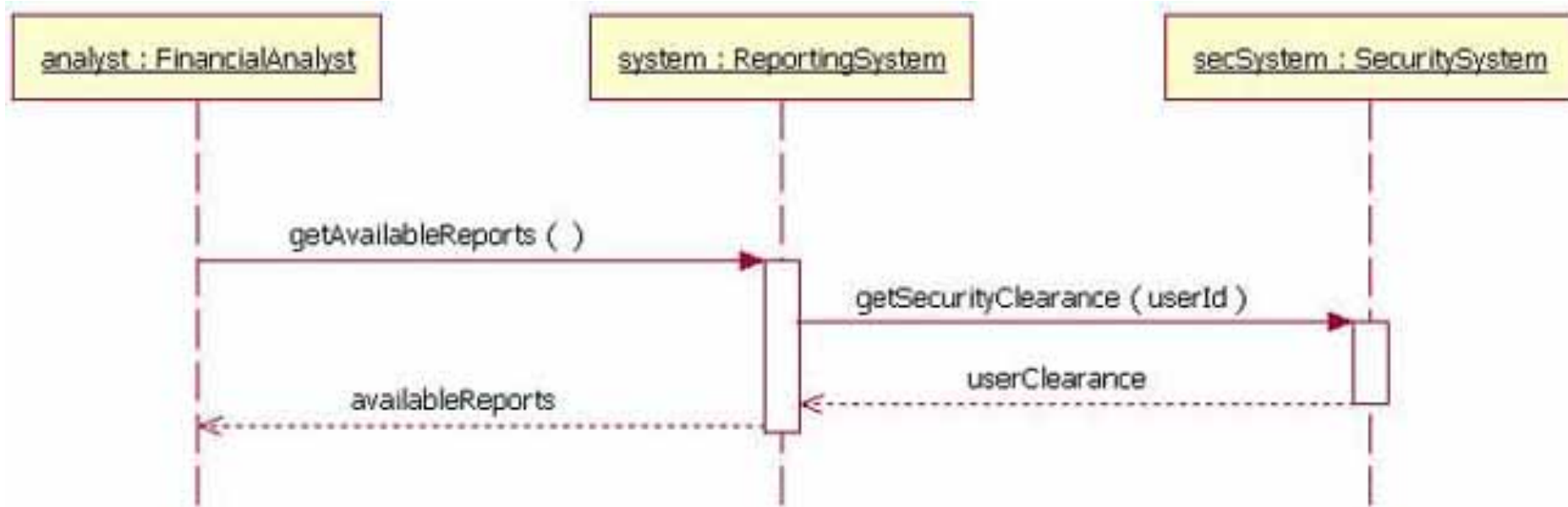
- Design should make programs more maintainable, scalable, etc.
- Design *process* is about communication
- High-level, architectural design
 - Client-server, REST, Peer to peer
- Mid-level, class-level design
 - Class diagrams
- Low-level, object interaction design
 - Sequence diagrams



Class diagrams



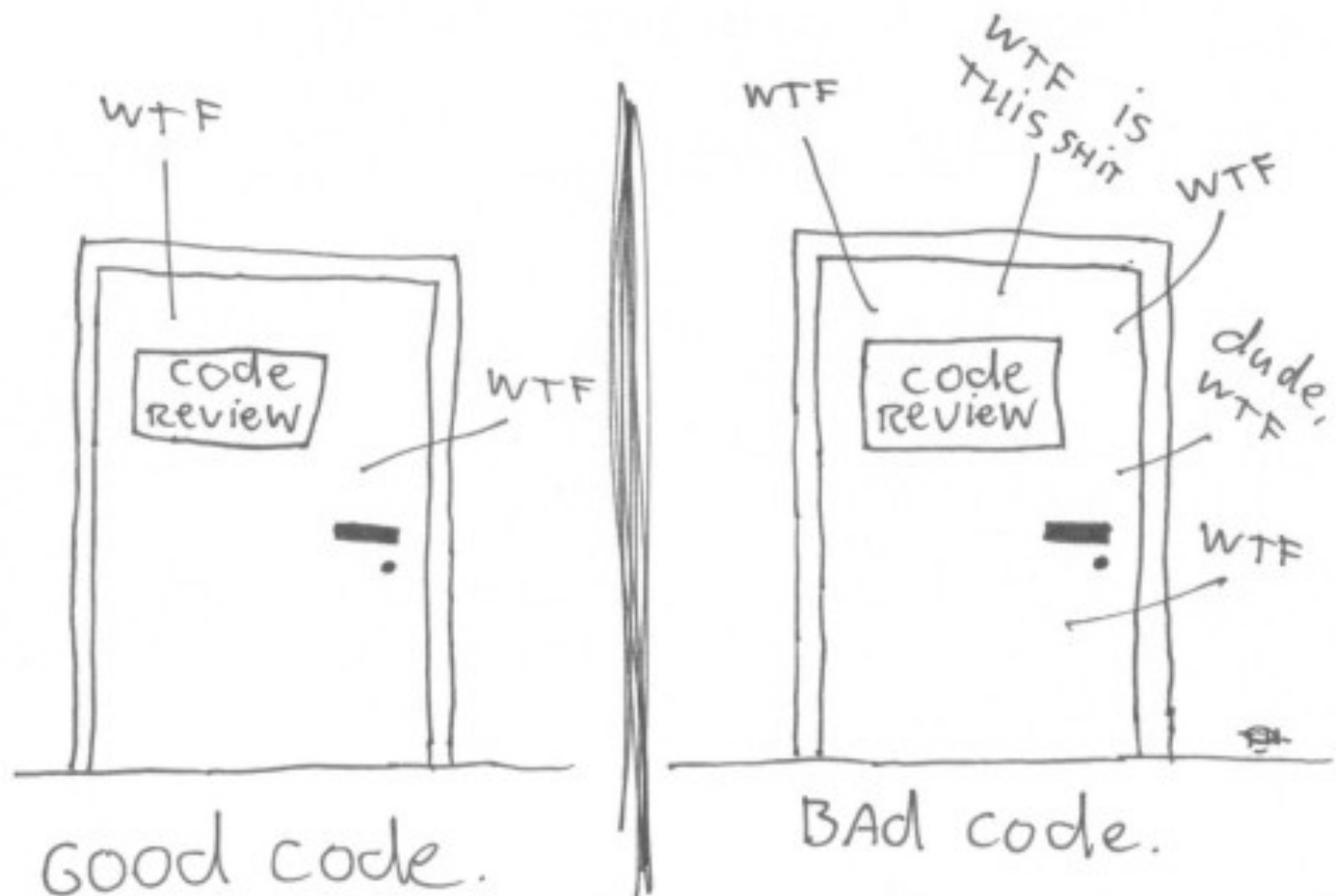
Sequence diagrams



Design principles

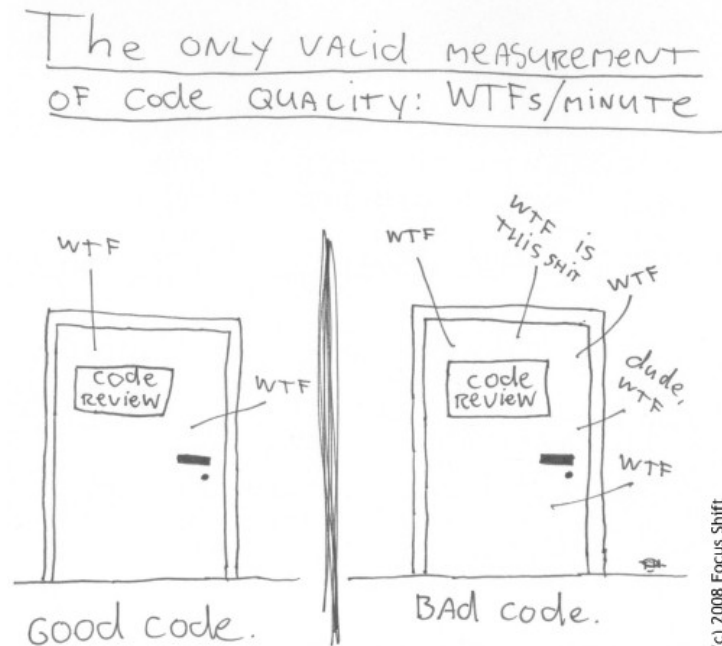
- Designs can be better or worse
- Modularity is a good goal (in general)
 - High Cohesion
 - Loose Coupling
 - Information Hiding
 - Open/Closed Principle
 - Liskov Substitution Principle

The ONLY valid measurement
of code quality: WTFs/minute



Which door is yours? How can we end up at the right door?

- (Automated) Analysis
 - Code Metrics
 - Formatting
 - Static and Dynamic Analysis
 - Measuring and Monitoring
- Craftsmanship
 - Code Reviews
 - Clean code
 - Smells and Refactoring



Refactoring and code smells

- Why refactor: evolving software and short iterations.
- Refactoring cycle: tests, code, tests, refactor, tests
- Code smells.
- Refactorings.

From smell to refactoring.

```
public void seek(float posValue) {
    //if fading, ignore
    if (bFading){
        return;
    }
    //save current position
    float fCurrentPos = fPos;
    //Do not seek to a position too near from the end of the audio file. MAX=98%
    if (posValue>0.98f){
        posValue = 0.98f;
    }
    // leave if already seeking
    if (player != null && getState() == BasicPlayer.SEEKING) {
        Log.warn("Already seeking, leaving"); //$NON-NLS-1$
        return;
    }
    if (mPlayingData.containsKey("audio.type") && player != null) { //$NON-NLS-1$
        Type type = TypeManager.getInstance().getTypeByTechDesc((String) mPlayingData.get("audio.type"));
        // Seek support for MP3. and WAVE
        if (type.getBooleanValue(XML_TYPE_SEEK_SUPPORTED)
            && mPlayingData.containsKey("audio.length.bytes")) { //$NON-NLS-1$ //$NON-NLS-2$
            int iAudioLength = ((Integer) mPlayingData.get("audio.length.bytes")).intValue();
            long skipBytes = (long) Math.round(iAudioLength * posValue); //$NON-NLS-1$
            try {
                player.seek(skipBytes);
                setVolume(fVolume); //need this because a seek reset volume
            } catch (Exception e) {
                Log.error(e);
                return;
            }
        } else {
            Messages.showErrorMessage("126"); //$NON-NLS-1$
            return;
        }
    }
}
```


Tech debt - invisible and negative value

	Visible	Invisible
Positive value	Visible feature	Hidden arch feature
Negative value	Visible defect	Technical debt

Testing

- Proving vs. testing
- Types of tests
 - Unit, regression, system, acceptance
- Challenges of testing: too many combinations!
- Test approaches: functional vs structural
- Stopping criteria (coverage?)

Evolution and maintenance

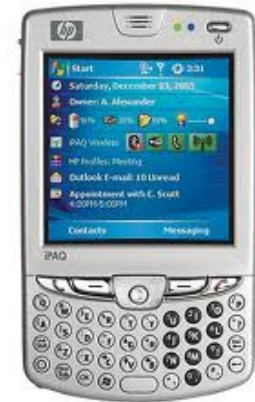
- Why maintenance is a large part of SE
- Types of software evolution
 - Laws of software evolution!
- Maintenance strategies and evolving APIs.



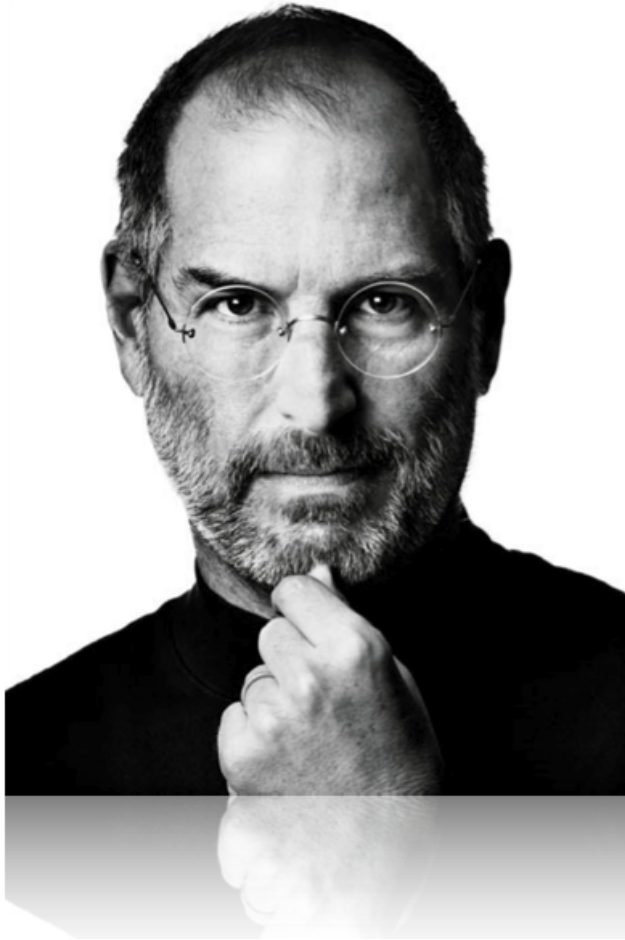
Designing interfaces is hard

How many of you can program or use all aspects of your

- digital watch?
- cell phone?
- DVD player?
- microwave?
- sewing machine?
- washer and dryer?
- stereo system (home or car)?



What is design?



*Design is not just
what it looks like
and feels like.*

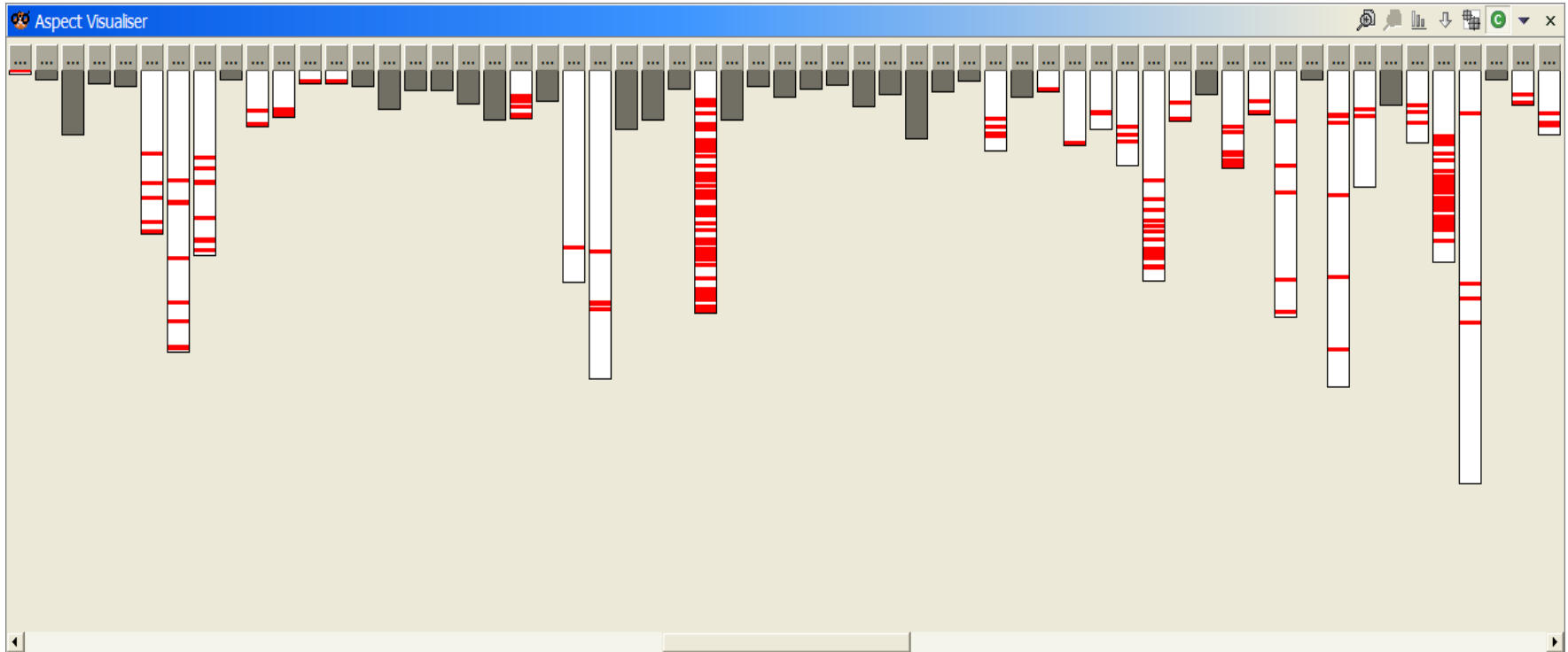
*Design is how it
works.*

Usability Principles



- Nielsen's 10 Principles of UI Design (Jakob Nielsen)

not so good modularity



- logging in Tomcat
 - scattered across the packages and classes
 - error handling, security, business rules, ...

Without AOP

ApplicationSession

[illegible]

StandardSession

[illegible]

SessionInterceptor

[illegible]

StandardManager

1. What is the purpose of the document?
The purpose of the document is to provide a detailed description of the project and its objectives.

2. What are the main objectives of the project?
The main objectives of the project are to develop a new product, improve the existing one, and increase the market share.

3. What are the key milestones of the project?
The key milestones of the project are the completion of the design phase, the start of the production phase, and the launch of the product.

4. What are the risks associated with the project?
The risks associated with the project are the delay in the production phase, the increase in the cost of the product, and the decrease in the market share.

5. What are the resources required for the project?
The resources required for the project are the human resources, the financial resources, and the material resources.

6. What is the timeline of the project?
The timeline of the project is as follows: Design phase (1 month), Production phase (2 months), and Launch phase (1 month).

7. What is the budget of the project?
The budget of the project is \$1,000,000.

8. What is the expected return on investment?
The expected return on investment is 20%.

9. What are the next steps of the project?
The next steps of the project are to start the production phase and to launch the product.

10. What is the conclusion of the document?
The conclusion of the document is that the project is feasible and profitable.

StandardSessionManager

[illegible]

ServerSession

[illegible]

ServerSessionManager

[illegible]

With AOP

ApplicationSession

1. **Introduction**
 2. **Background**
 3. **Methodology**
 4. **Results**
 5. **Discussion**
 6. **Conclusion**
 7. **References**
 8. **Appendix**
 9. **Figure 1**
 10. **Figure 2**
 11. **Figure 3**
 12. **Figure 4**
 13. **Figure 5**
 14. **Figure 6**
 15. **Figure 7**
 16. **Figure 8**
 17. **Figure 9**
 18. **Figure 10**
 19. **Figure 11**
 20. **Figure 12**
 21. **Figure 13**
 22. **Figure 14**
 23. **Figure 15**
 24. **Figure 16**
 25. **Figure 17**
 26. **Figure 18**
 27. **Figure 19**
 28. **Figure 20**
 29. **Figure 21**
 30. **Figure 22**
 31. **Figure 23**
 32. **Figure 24**
 33. **Figure 25**
 34. **Figure 26**
 35. **Figure 27**
 36. **Figure 28**
 37. **Figure 29**
 38. **Figure 30**
 39. **Figure 31**
 40. **Figure 32**
 41. **Figure 33**
 42. **Figure 34**
 43. **Figure 35**
 44. **Figure 36**
 45. **Figure 37**
 46. **Figure 38**
 47. **Figure 39**
 48. **Figure 40**
 49. **Figure 41**
 50. **Figure 42**
 51. **Figure 43**
 52. **Figure 44**
 53. **Figure 45**
 54. **Figure 46**
 55. **Figure 47**
 56. **Figure 48**
 57. **Figure 49**
 58. **Figure 50**
 59. **Figure 51**
 60. **Figure 52**
 61. **Figure 53**
 62. **Figure 54**
 63. **Figure 55**
 64. **Figure 56**
 65. **Figure 57**
 66. **Figure 58**
 67. **Figure 59**
 68. **Figure 60**
 69. **Figure 61**
 70. **Figure 62**
 71. **Figure 63**
 72. **Figure 64**
 73. **Figure 65**
 74. **Figure 66**
 75. **Figure 67**
 76. **Figure 68**
 77. **Figure 69**
 78. **Figure 70**
 79. **Figure 71**
 80. **Figure 72**
 81. **Figure 73**
 82. **Figure 74**
 83. **Figure 75**
 84. **Figure 76**
 85. **Figure 77**
 86. **Figure 78**
 87. **Figure 79**
 88. **Figure 80**
 89. **Figure 81**
 90. **Figure 82**
 91. **Figure 83**
 92. **Figure 84**
 93. **Figure 85**
 94. **Figure 86**
 95. **Figure 87**
 96. **Figure 88**
 97. **Figure 89**
 98. **Figure 90**
 99. **Figure 91**
 100. **Figure 92**
 101. **Figure 93**
 102. **Figure 94**
 103. **Figure 95**
 104. **Figure 96**
 105. **Figure 97**
 106. **Figure 98**
 107. **Figure 99**
 108. **Figure 100**
 109. **Figure 101**
 110. **Figure 102**
 111. **Figure 103**
 112. **Figure 104**
 113. **Figure 105**
 114. **Figure 106**
 115. **Figure 107**
 116. **Figure 108**
 117. **Figure 109**
 118. **Figure 110**
 119. **Figure 111**
 120. **Figure 112**
 121. **Figure 113**
 122. **Figure 114**
 123. **Figure 115**
 124. **Figure 116**
 125. **Figure 117**
 126. **Figure 118**
 127. **Figure 119**
 128. **Figure 120**
 129. **Figure 121**
 130. **Figure 122**
 131. **Figure 123**
 132. **Figure 124**
 133. **Figure 125**
 134. **Figure 126**
 135. **Figure 127**
 136. **Figure 128**
 137. **Figure 129**
 138. **Figure 130**
 139. **Figure 131**
 140. **Figure 132**
 141. **Figure 133**
 142. **Figure 134**
 143. **Figure 135**
 144. **Figure 136**
 145. **Figure 137**
 146. **Figure 138**
 147. **Figure 139**
 148. **Figure 140**
 149. **Figure 141**
 150. **Figure 142**
 151. **Figure 143**
 152. **Figure 144**
 153. **Figure 145**
 154. **Figure 146**
 155. **Figure 147**
 156. **Figure 148**
 157. **Figure 149**
 158. **Figure 150**
 159. **Figure 151**
 160. **Figure 152**
 161. **Figure 153**
 162. **Figure 154**
 163. **Figure 155**
 164. **Figure 156**
 165. **Figure 157**
 166. **Figure 158**
 167. **Figure 159**
 168. **Figure 160**
 169. **Figure 161**
 170. **Figure 162**
 171. **Figure 163**
 172. **Figure 164**
 173. **Figure 165**
 174. **Figure 166**
 175. **Figure 167**
 176. **Figure 168**
 177. **Figure 169**
 178. **Figure 170**
 179. **Figure 171**
 180. **Figure 172**
 181. **Figure 173**
 182. **Figure 174**
 183. **Figure 175**
 184. **Figure 176**
 185. **Figure 177**
 186. **Figure 178**
 187. **Figure 179**
 188. **Figure 180**
 189. **Figure 181**
 190. **Figure 182**
 191. **Figure 183**
 192. **Figure 184**
 193. **Figure 185**
 194. **Figure 186**
 195. **Figure 187**
 196. **Figure 188**
 197. **Figure 189**
 198. **Figure 190**
 199. **Figure 191**
 200. **Figure 192**
 201. **Figure 193**
 202. **Figure 194**
 203. **Figure 195**
 204. **Figure 196**
 205. **Figure 197**
 206. **Figure 198**
 207. **Figure 199**
 208. **Figure 200**
 209. **Figure 201**
 210. **Figure 202**
 211. **Figure 203**
 212. **Figure 204**
 213. **Figure 205**
 214. **Figure 206**
 215. **Figure 207**
 216. **Figure 208**
 217. **Figure 209**

ServerSession

```

1  # Import the pandas module
2  import pandas as pd
3
4  # Create a DataFrame with 5 rows and 3 columns
5  data = {'Category': ['A', 'B', 'A', 'B', 'A'],
6         'Value': [10, 20, 30, 40, 50],
7         'Label': ['Low', 'High', 'Low', 'High', 'Low']}
8
9  # Print the DataFrame
10 print(data)
11
12 # Filter rows where 'Category' is 'A'
13 filtered_data = data[data['Category'] == 'A']
14
15 # Print the filtered DataFrame
16 print(filtered_data)
17
18 # Calculate the mean 'Value' for each 'Category'
19 mean_values = data.groupby('Category')['Value'].mean()
20
21 # Print the mean values
22 print(mean_values)
23
24 # Sort the DataFrame by 'Value' in descending order
25 sorted_data = data.sort_values('Value', ascending=False)
26
27 # Print the sorted DataFrame
28 print(sorted_data)
29
30 # Save the DataFrame to a CSV file
31 data.to_csv('data.csv')
32
33 # Load the DataFrame from a CSV file
34 loaded_data = pd.read_csv('data.csv')
35
36 # Print the loaded DataFrame
37 print(loaded_data)

```

StandardSession

[illegible]

SessionInterceptor

1. **QUESTION** (10 marks)
 The following table shows the number of people who attended a concert in each of the five years from 2000 to 2004.
 Year: 2000, 2001, 2002, 2003, 2004
 Number of people: 120, 150, 180, 200, 220
 (a) Calculate the mean number of people who attended the concert.
 (b) Calculate the standard deviation of the number of people who attended the concert.
 (c) Calculate the variance of the number of people who attended the concert.
 (d) Calculate the coefficient of variation of the number of people who attended the concert.
 (e) Calculate the range of the number of people who attended the concert.
 (f) Calculate the median number of people who attended the concert.
 (g) Calculate the mode number of people who attended the concert.
 (h) Calculate the interquartile range of the number of people who attended the concert.
 (i) Calculate the quartile deviation of the number of people who attended the concert.
 (j) Calculate the mean deviation of the number of people who attended the concert.

StandardManager

[illegible]

StandardSessionManager

Figure 1 illustrates the experimental setup. A participant is seated at a table, looking at a video screen. On the screen, a starting point and a target are visible. The participant's hand is at the starting point. The distance between the starting point and the target is indicated. The setup includes a computer and a video screen.

ServerSessionManager

[illegible]

Be a better software engineer

- Write clean, understandable code not only for yourself, but also for others
- Test your code, measure what you test
- Maintain your design and code.
- Share and be a team player!

Q & A