Bachelor Project

Presentation II – Process & Results
4. February 2020

Alain Stulz / 2020
Assessing and Improving the Software Quality of an iOS App Framework
Recap
“Festival Buddy”
How To Create Many Apps Efficiently?

• Custom Framework
Bern Welcome
Five up
Analyze

RQ1: How can we assess the quality of our system?
Non-Functional Requirements
Code Level

Maintainability Measures

52d
Debt

2.4k
Code Smells

Coverage Measures

5.6%
Coverage

Duplications Measures

1.7%
Duplications

55
Duplicated Blocks
Module Level
Organization Level

<table>
<thead>
<tr>
<th>Project</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>academy-ios</td>
<td>0</td>
</tr>
<tr>
<td>adelboden-ultimatefan-ios</td>
<td>0</td>
</tr>
<tr>
<td>arud-consumption-diary-ios</td>
<td>1</td>
</tr>
<tr>
<td>BernWelcome_iOS</td>
<td>8</td>
</tr>
<tr>
<td>EDA_DFAE_iOS</td>
<td>8</td>
</tr>
<tr>
<td>gmenweg-ios</td>
<td>0</td>
</tr>
<tr>
<td>gymme-ios</td>
<td>0</td>
</tr>
<tr>
<td>hep-project-journal-ios</td>
<td>1</td>
</tr>
<tr>
<td>herbstmesse-ios</td>
<td>0</td>
</tr>
<tr>
<td>homussen-ios</td>
<td>1</td>
</tr>
<tr>
<td>inlingua-ios</td>
<td>4</td>
</tr>
<tr>
<td>lukb-ios</td>
<td>1</td>
</tr>
<tr>
<td>marketing-expert-ios</td>
<td>1</td>
</tr>
<tr>
<td>marketing-lexicon-ios</td>
<td>1</td>
</tr>
<tr>
<td>mindpills-ios</td>
<td>0</td>
</tr>
<tr>
<td>mplay-massimo-fall2019-ios</td>
<td>8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Modules from Framework</th>
<th>Imports</th>
<th>Usages</th>
<th>Decision</th>
<th>Focus</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABFCoreUI</td>
<td>8</td>
<td>8</td>
<td>Scope</td>
<td>Quality</td>
</tr>
<tr>
<td>ABFDataAccessLayer</td>
<td>8</td>
<td>8</td>
<td>Scope</td>
<td>Quality</td>
</tr>
<tr>
<td>ABFCoreApp</td>
<td>8</td>
<td>8</td>
<td>Scope</td>
<td>Quality</td>
</tr>
<tr>
<td>ABFFileInfoModule</td>
<td>8</td>
<td>6</td>
<td>Scope</td>
<td>Quality</td>
</tr>
<tr>
<td>ABFNetworkEngine</td>
<td>8</td>
<td>6</td>
<td>Scope</td>
<td>Quality</td>
</tr>
<tr>
<td>ABFCoordinator</td>
<td>7</td>
<td>5</td>
<td>Scope</td>
<td>Quality</td>
</tr>
<tr>
<td>ABFDataAccessLayerConstants</td>
<td>8</td>
<td>4</td>
<td>Ignore</td>
<td></td>
</tr>
<tr>
<td>ABFLocationManager</td>
<td>8</td>
<td>3</td>
<td>Ignore</td>
<td></td>
</tr>
<tr>
<td>ABFAttachmentsModule</td>
<td>3</td>
<td>3</td>
<td>Ignore</td>
<td></td>
</tr>
<tr>
<td>ABFPreCacheManager</td>
<td>8</td>
<td>3</td>
<td>Ignore</td>
<td></td>
</tr>
<tr>
<td>MATableManager</td>
<td>9</td>
<td>3</td>
<td>Scope</td>
<td>Dependencies</td>
</tr>
<tr>
<td>ABFAppCategories</td>
<td>8</td>
<td>2</td>
<td>Scope</td>
<td>Dependencies</td>
</tr>
<tr>
<td>ABFImageManager</td>
<td>8</td>
<td>1</td>
<td>Scope</td>
<td>Dependencies</td>
</tr>
<tr>
<td>ABFNewsModule</td>
<td>1</td>
<td>1</td>
<td>Ignore</td>
<td></td>
</tr>
<tr>
<td>ABFPushSettingsModule</td>
<td>8</td>
<td>2</td>
<td>Scope</td>
<td>Dependencies</td>
</tr>
<tr>
<td>ABFSettingsModule</td>
<td>2</td>
<td>2</td>
<td>Ignore</td>
<td></td>
</tr>
<tr>
<td>ABFShareModule</td>
<td>8</td>
<td>2</td>
<td>Scope</td>
<td>Dependencies</td>
</tr>
<tr>
<td>ABFWebModule</td>
<td>8</td>
<td>2</td>
<td>Scope</td>
<td>Dependencies</td>
</tr>
<tr>
<td>ABFAuthenticationModule</td>
<td>1</td>
<td>1</td>
<td>Ignore</td>
<td></td>
</tr>
<tr>
<td>ABFLearningCardsModule</td>
<td>1</td>
<td>1</td>
<td>Ignore</td>
<td></td>
</tr>
<tr>
<td>ABFLocalization</td>
<td>8</td>
<td>1</td>
<td>Scope</td>
<td>Dependencies</td>
</tr>
<tr>
<td>ABFMapModule</td>
<td>0</td>
<td>0</td>
<td>Remove</td>
<td></td>
</tr>
<tr>
<td>ABFPhotoGalleryModule</td>
<td>1</td>
<td>1</td>
<td>Ignore</td>
<td></td>
</tr>
<tr>
<td>ABFPhotoPicker</td>
<td>1</td>
<td>1</td>
<td>Ignore</td>
<td></td>
</tr>
<tr>
<td>ABFProductModule</td>
<td>0</td>
<td>0</td>
<td>Remove</td>
<td></td>
</tr>
<tr>
<td>ABFProgramModule</td>
<td>0</td>
<td>0</td>
<td>Remove</td>
<td></td>
</tr>
</tbody>
</table>
Developer Interviews

+ Easier to start projects
+ Consistency and speed

- Hard to understand and change
- Effects of changes not visible
Lessons Learned

RQ1: How can we assess the quality of our system?

• Focus on Developer’s Perspective
  • NFRs: Maintainability, Evolvability, …
• Use Different Data Sources
• Specific Tools for iOS Projects
Improve

RQ2: How can we improve the system’s quality?
Improve

1. Process
2. Maintenance
3. Refactoring
Improve

1. Process
2. Maintenance
3. Refactoring

• Developer Conventions
• Continuous Integration
  • Automate Client Projects
  • Build Feedback
Improve

1. Process
2. Maintenance
3. Refactoring

• Remove Old Code
• Rearrange Methods
Improve

1. Process
2. Maintenance
3. Refactoring

- Cover and Modify
- Techniques
Lessons Learned

RQ2: How can we improve the system’s quality?

• Set up Systems and Conventions
• Perform Basic Maintenance
• Cover and Modify
• Use Delegates for Tests
Rewrite

RQ3: What would a better software design look like?
Rewrite

1. Architecture
2. Benefits
Rewrite

1. Architecture
2. Benefits

• Clean Swift Architecture
  • Split into Scenes
  • Protocols
• Modularization
Rewrite

1. Architecture
2. Benefits

• Only Relevant Features
• Less Clutter in Framework
  • Define UI in Client Projects
• More Flexibility
Lessons Learned

RQ3: What would a better software design look like?

• Small and Generalized
• Newer Tools
• Well-Defined Architecture
Conclusion
Conclusion

• Recommendation: Rewrite
  • Changed Requirements, New Opportunities
• Refactor vs. Rewrite
  • Only if you need to
  • Refactor if you need to keep system
  • Rewrite if system holds you back
Summary

Analyze
   Focus on NFRs, Different Sources, Specific Tools

Improve
   Systems and Conventions, Reduce Size, Cover and Modify

Rewrite
   Small and Flexible, Define Architecture, Allow Customization
Testability

• Business Logic, not UI
• Move Logic out of Centralized Classes
  • Single Responsibility
  • DB and Network Calls
• Reduce Dependencies
  • Precompiler Flags
  • Leverage Delegates for Tests
Strategies

• Precompiler Flags
  • #define and #ifdef
  • Escape out of a method early

• Mock Delegates
  • Offer a well-defined interface
  • Control and get insight into class

• Optional Protocol Methods
  • Use @optional Keyword
  • Implement methods only for testing (e.g. shouldCallAPI)