DESIGN AND IMPLEMENTATION OF MULTI-HEAD PRESENTATION SOFTWARE FOR THE iOS PLATFORM

by

Michael S. Babienco

A PROJECT DEFENSE

Presented to the Faculty of

The School of Computing at Southern Adventist University

In Partial Fulfilment of Requirements

For the Degree of Master of Science

Major: Computer Science

Under the Supervision of Professor Hall

Collegedale, Tennessee

February, 2015

Design and Implementation of Multi-head Presentation Software for the iOS Platform

Approved by:

Tyson S. Hall

Professor Tyson Hall, Adviser

[illegible]

Professor Scot Anderson

[illegible]

Professor Richard Halterman

Date Approved 2/24/15

DESIGN AND IMPLEMENTATION OF MULTI-HEAD PRESENTATION SOFTWARE FOR THE iOS PLATFORM

Michael S. Babienco

Southern Adventist University, 2015

Adviser: Tyson Hall, Ph.D.

ShareSynch, currently available for both Windows and Mac OS X, is a presentation software application tailored towards evangelistic speakers with limited experience. The software has several essential features including the use of speaker notes during a presentation, support for independent slide and speaker note language, speaker note pagination with dynamic font scaling, editing of presentations and rich text speaker notes from within the application, and dynamic appeal video configuration. No known iOS application contains all of the PC software's essential features. In this paper, we discuss the design and implementation of ShareSynch on the iOS platform. The iOS version of ShareSynch, developed in conjunction with ShareHim, mirrors functionality currently available in the PC software and also adds new features such as dynamically generated PDF documents, downloadable sermon series, and a new file format that provides for reduced storage consumption. ShareSynch was published in the iOS App Store on March 2, 2015.

DEDICATION

For all those who imagined that I had an actual office in the computing department

Contents

Contents ix

List of Figures xiii

List of Tables xv

1 Introduction 1

2 Background 5

2.1 Quality Assurance 5

2.2 Design Patterns 7

2.3 Mobile Platforms 8

2.4 User Expectations 9

2.5 Current Mobile Presentation Applications 10

2.5.1 Full-Featured Applications 10

2.5.2 Template-based Applications 12

2.5.3 Other Presentation Applications 12

3 ShareSynch iOS Development and Implementation 15

3.1 Application Summary 16

3.2 Development Approach 18

3.3 Task Delineation 20
3.3.1 Presentation Mode 24
3.3.2 File and Database Implementation 24
3.3.3 Sermon Editing System & Settings 24
3.3.4 Sermon Slideshow Installation 25
3.3.5 UI Design and Implementation 25
3.3.6 Beta and User Testing 26
3.3.7 Final Changes, Bug Fixes, Testing, and Documentation 26
3.4 Final Deliverables 26

4 Testing & Evaluation Plan 29

5 Testing and Evaluation Results 33

6 Conclusion 39

A Requirements Specification 41
A.1 Task Delineation 41
A.2 Hardware Requirements 42
A.3 Application Requirements 42
A.3.1 General 42
A.3.2 Download & Installation 44
A.3.3 Opening and Managing a Sermon Series 46
A.3.4 Presentation Mode 48
A.3.5 Edit Mode 51
A.3.6 Settings 53
A.4 Change Orders 54

B Application Screenshots **57**

Bibliography **63**

List of Figures

3.1 The database schema for the ShareSynch iOS application allows for the storage of all sermon series information and user data. 22

B.1 ShareSynch's main menu screen lets users directly access available sermon series and settings without having to open another page. The sections for available sermon series and settings can be expanded or contracted as necessary. 58

B.2 The select presentation screen shows which variation is currently loaded and which sermons are currently available. The user can also change the slide language, change the speaker note language, or generate a PDF of a selected sermon from this screen. The currently selected sermon cell is highlighted in light blue, and its title text is set to a bold font. 59

B.3 Presentation mode prominently displays speaker notes for the user. Arrows indicate that advancing the speaker notes will move the presentation to the next slide. When a sermon has an appeal, appeal options are shown in the toolbar along with the change font and presentation mode settings. 60

B.4 ShareSynch lets users edit a sermon presentation in a variety of ways. Notes are editable in rich text. Slides can be inserted from the user's device or from another sermon within the same series. The user can make a variety of other changes to the presentation from this screen, such as changing sermon order or slide visibility. 61

B.5 ShareSynch lets users preview dynamically generated PDF documents before sharing them with others. The user's customized speaker notes and sermon order are reflected in these documents. The page-breaking algorithm attempts to minimize the amount of whitespace at the end of each page by placing as much of a slide's speaker notes on the current page as possible before inserting the rest of the notes on subsequent pages. 62

List of Tables

3.1 Libraries and Open-Source Projects Used in ShareSynch iOS 21
3.2 Task Delineation for the Development of the ShareSynch iOS Application . . . 23

4.1 ShareSynch Performance Testing 31

5.1 ShareSynch Performance Testing Results 34
5.2 ShareSynch Battery Test Results 36

Chapter 1

Introduction

ShareHim's ShareSynch presentation software has been designed to help speakers learn to easily practice and participate in evangelism, both locally and internationally [1]. The software has many advantages and unique features when compared to similar presentation software; however, speakers of ShareSynch sermons can only use ShareSynch on a Windows or Mac OS X computer and cannot present from any other operating system or device due to ShareSynch's proprietary sermon file format. The call for a mobile version of the software has grown in recent years due to the increased popularity and convenience of presenting sermons from mobile devices such as tablets or phones. Creation of a mobile version of the ShareSynch software gives speakers the ability to use ShareSynch sermon materials and take advantage of all of ShareSynch's unique features while presenting from a mobile device.

ShareSynch includes several unique features that help novice speakers actively participate in the evangelism process. A set of sermon presentations make up a sermon series, and each series contains several different languages for both sermon slides and speaker notes. The language used for speaker notes does not need to match the language used for sermon slides, and the languages used for a presentation can be easily changed from within the application. Speakers can adjust the font size of speaker notes during a presentation, and notes are dynamically paginated based on this font size. When a speaker advances the presentation, the next slide is not shown if there are more pages of notes to show for the current slide. Instead, when there is another page of notes to show, the speaker notes are scrolled smoothly downward to the next page of notes. Videos shown for the sermon appeal can change automatically dependent upon the speaker's choice for the appeal type and song without the need for the speaker to edit the overall sermon.

Android and iOS are the two largest mobile application markets. Although Android has a bigger market share when compared to iOS, many Android devices do not support external displays at all, and more than 30% of currently active Android devices cannot utilize the external display APIs [2, 3, 4, 5]. All iOS products since the release of the iPhone 4 support external displays, and at least 95% of iOS devices can use the external display APIs [6, 7, 8]. We discuss the development of an iOS version of ShareSynch that mirrors functionality currently available in the PC version of ShareSynch. The iOS version also extends the software's capabilities by allowing for the download and installation of new sermon series via the internet, creating dynamically generated PDF documents of user-edited sermons, and incorporating a new file format and storage mechanism that has the potential to greatly reduce consumed storage space for a sermon series.

The iOS version of ShareSynch was developed in conjunction with the Center for Innovation and Research in Computing (CIRC) at Southern Adventist University.

The rest of this paper discusses background material in Chapter 2, discusses the development and implementation for the project in Chapter 3, and describes test methodologies for the application in Chapter 4. Chapter 5 gives the results of application testing, and Chapter 6 presents a summary of project accomplishments and deliverables. Appendix A gives the application requirements, and Appendix B shows some screenshots of the finished product.

Chapter 2

Background

Emerging research in mobile development is quite disparate, including mobile application quality assurance, design patterns, the choice of application deployment type and platform, and user expectations. In this chapter, we discuss research pertinent to the development of the iOS version of ShareSynch and perform a competitive analysis of the available features in currently available mobile presentation applications.

2.1 Quality Assurance

Mobile application markets are driven by customer reviews, and any faults within an application can drive users away from utilizing the application [9]. In order to achieve a high level of quality, applications should be thoroughly tested, which can be difficult to achieve because mobile applications have unique types of challenges when compared to traditional software programs. Several challenging areas include the invoking of remote services, creating one application on a wide variety of hardware devices, power consumption, and unique security concerns [10]. Muccini et al. [11] discuss challenges in the realm of mobile application testing, which include mobile connectivity, limited resources, and differing versions of operating systems. Testing all areas of an application can take a considerable amount of time, so automating as many tests as possible is essential to reducing the amount of time testing the application. One particularly difficult testing area is graphical user interface (GUI) testing, because tests must check to see if all device types show the application in a clean and user-friendly way in all available device orientations. An application's GUI must adhere to user interface (UI) guidelines and may include universal user interfaces, which increases the difficulty of GUI testing [12].

GUI testing is defined by Nguyen et al. [13] as "the process of testing a software application through its GUI," and it is a form of system-level testing for a GUI application. Unfortunately, the number of GUI test cases is extremely large, and it grows quickly whenever any user functionality is added to the application. Usually, GUI testing takes two forms. The first uses a scripting language to create a type of unit tests that can perform manipulations on GUI items as well as invoke user events. These tests are manually coded, and testers can assert GUI item properties in the same manner as normal unit test assertions. The second type of GUI testing involves capturing user events and then replaying these events at a later time to see if they still function as required for the application. Many GUI testing frameworks exist to help developers create and execute GUI tests on the iOS platform, including GUITAR [13], Calabash [14], Kif [15], Xamarin Test Cloud [16], UIAutomation [17], and UI AutoMonkey [18]. None of these frameworks, however, allow a developer to create GUI tests using both Objective-C code and official, documented Apple APIs. Only Kif allows a developer to create tests in Objective-C, but it uses undocumented APIs and relies on the accessibility properties of GUI elements rather than pointers to GUI objects, which can make test creation difficult and time consuming.

2.2 Design Patterns

Biel and Gruhn [19] propose two new design patterns for mobile development: the Client-side Multi-Screen Support (CMSS) pattern and the Mobile Application Usability Test Suite (MAUTS) pattern. The CMSS pattern helps developers support a wide variety of hardware screen sizes and resolutions, and the MAUTS pattern suggests that developers utilize users and their data in automated and manual tests during all stages of application development in order to reduce application crashes. Although the patterns were designed for use on Android devices, the CMSS pattern is still applicable to iOS and the ShareSynch project due to the multitude of screen resolutions and sizes for iOS [20]. In the mobile market, users have a variety of device types, each with their own screen size and resolution. Developers must create a UI that allows users to perform all functions in a given application, remains easy to utilize by all types of users, and adheres to UI guidelines. They must also make decisions regarding how UI differences will be handled between different platforms or devices [12]. Devices can be categorized into three different pixel density classes: high, medium, or low density classes. An application needs to function properly and look attractive on each class of device as well as on every device within each class. The Client-side Multi-Screen Support pattern [19] advises developers to design their UI for a specific screen size and resolution in each class, then use a minimal amount of runtime code to scale this general UI to a UI that functions properly and looks acceptable on each type of device within that class. This technique improves user experience across many devices, but can increase media design efforts and programming effort when compared to designing for a single screen size and resolution.

2.3 Mobile Platforms

Developers of mobile applications often must decide between building a fully-native application for each mobile operating system, building a mobile web application that can run on multiple platforms, or using a third-party abstraction tool to transform a single set of code into native code for each device [10]. Although cross platform tools seem promising, a market analysis performed by Smith's Point Analytics [21] states that limitations in these tools, specifically in the realms of performance and device-specific functionality, are causing developers to struggle and reducing efficiency gains normally obtained by using cross platform tools. The market analysis also suggests that although the market for cross platform tools is growing, the rate at which the market is growing is rapidly decreasing. When a single code base is used for a large variety of devices, irregularities often exist due to device-specific conditions [22].

Charland and Leroux [23] investigated whether native applications can really be replaced by web applications. The main benefit of web applications is their ability to run across different platforms, thus saving development costs and reducing development time. Although web application capabilities are increasing, they are still slower than native applications, and it can be difficult to replicate the native user interface in a web application. A user expects a speedy application that has an interface design similar to other native applications, and a web application cannot always meet those needs.

2.4 User Expectations

Customer expectations for mobile applications are reviewed by Haller [24] in an examination of more than 1,000 application reviews in three app stores. Complaints fell into four different categories: functional problems, such as bugs or crashes; technical issues, such as performance and battery consumption; GUI interaction problems, including poor design or missing languages; and business decision complaints or suggestions. Improvement ideas, bugs, and crashes ranked among the top five reasons for poor application reviews. Haller believed that testing procedures were not carried out by many projects, as shown by the multitude of crashes and bugs. Khalid et al. [25] also investigated the reasons for user complaints regarding mobile applications in a study of 6,390 one and two-star reviews. Khalid et al. found that functional errors, feature requests, and application crashes are the reason for more than 50% of complaints, thus supporting the work by Haller [24].

The work on privacy expectations by Jung et al. [26] showed that users do not want mobile applications to gather and share unnecessary or private information. Jung et al. enlisted twenty Android smartphone users and monitored the data that their smartphone applications collected and transferred for three weeks. Participants were interviewed after the three week period to learn about their privacy expectations and reactions to the actual data collected and transferred. Several users expressed surprise at the amount of data that several apps collected, and over 50% of users were concerned by applications that shared location information with third parties. Users expected applications to collect data in a way that coincided with the way the application was used.

2.5 Current Mobile Presentation Applications

A large number of presentation applications are currently available in the iOS App Store. Each of these applications have different features and capabilities. We performed a search on the keyword "presentation" in the iPad iOS App Store on August 7, 2014 and January 26, 2015 and located the top five presentation applications by popularity and by relevance as determined by the App Store. Three applications, Google Slides [27], Keynote [28], and Prezi [29] were listed in both groups. This section organizes most of these applications according to their features; the section first reviews full-featured applications, then discusses template-based applications, and ends with an examination of the remaining presentation applications. CloudOn [30], which appeared in App Store results during the August 7 search, is not discussed in this section because it was purchased by Dropbox [31] on January 19, 2015, and it will be shutting down on March 15, 2015 [32].

2.5.1 Full-Featured Applications

Keynote [28], created by Apple Inc., is available for free for new iOS owners who have activated their device after August 2013 [33]. Keynote lets users create, edit, and give presentations via an external device or screen. All device orientations are supported in presentation mode via several different layout options, and the application lets the user create duplicates of presentations. Unfortunately, presenter notes are only available in presentation mode when the iOS device is connected to an external display; when the device is not connected to an external display, only the presentation's slides are shown on the speaker's device in presentation mode. Furthermore, a speaker cannot view speaker notes, the current slide, and a preview of the upcoming slide all at one time while in presentation mode.

Microsoft PowerPoint for iPad [34] brings the popular PC program to iPad devices. The application is free to download, but in order to view speaker notes during a presentation as well as have full editing capabilities, the user must have an Office 365 subscription, which requires a monthly or yearly payment. Presentations can be created, imported, and presented without an Office 365 subscription; speaker notes can be created and edited in rich text, even though they are not seen while presenting in the free version of the software. When an Office 365 subscription is active, the application has many of the same capabilities as the desktop version of PowerPoint.

Google Slides [27] offers many of the same capabilities as Microsoft PowerPoint for iPad. Although the user must have a Google account to use the application, once the user logs in to their Google account, they can create, edit, and present Google Slides or PowerPoint documents. Speaker notes are editable in rich text, but they are not viewable while giving presentations. The application offers real-time, online editing capabilities with multiple users working on the same document, and it offers Google Drive functionality for loading and saving documents.

WPS Office [35] is a completely-free application that allows users to create and edit Microsoft Word, Excel, and PowerPoint documents. Pictures, music, and video can be inserted into the presentation from the user's device. A small number of rich text capabilities are available when editing slides, but speaker notes can only be written in plain text. Presentation mode shows the current presentation progress, a presentation timer, the current slide as well as previews of other slides, and optionally shows speaker notes and slide preview thumbnails.

2.5.2 Template-based Applications

FlowVella [36], previously known as Flowboard, utilizes templates for quick presentation creation. Individual slides are made from a pre-defined or blank layout, and users can easily insert text, pictures, or videos into the slide layouts. Each slide element can link to another presentation slide, allowing users to easily create presentations that quickly jump between slides. Users can pay for extra features, such as the ability to export slides to a PDF document. FlowVella does not offer speaker note functionality.

Haiku Deck [37] and Prezi [29] offer more templates for users to quickly create appealing presentations. Speaker notes are visible in Haiku Deck while in vertical device orientations, and the application lets speakers use iPhones as presentation remotes. Haiku Deck also helps you easily share your presentations with others through a multitude of sharing options. Prezi creates presentations that consist of one giant slide. As a presentation continues, Prezi zooms in and out of the slide in order to give the appearance of having individual slides. Within the provided templates, the Prezi slides are easily configurable in theme and structure. Like FlowVella, Prezi has no speaker note capabilities.

2.5.3 Other Presentation Applications

SlideShark [38] lets users store their presentations in the cloud for easy access across several iOS devices. SlideShark only lets users give presentations; no editing capabilities are available within the application. When presenting from SlideShark, animations, fonts, and other media are shown in the same manner as in the desktop version of PowerPoint. Speakers can add annotations to slides while they are being presented. Presentation mode can show speaker notes, a thumbnail list of slides, timers, and the current slide on the iPad version of the application; however, speaker notes can only be seen when presenting from an iPad.

Nearpod [39] is a unique presentation application that is built for the education market. A speaker creates an interactive presentation filled with items such as quizzes or polls. Audience members join the presentation via the internet and their own electronic device and can interact with the presentation when required, such as by answering poll questions. As audience members submit answers to poll questions or otherwise interact with the presentation, the speaker views these results in real-time. Unfortunately, the Nearpod application pushes users to an external web browser for several important tasks, such as presentation creation, and it does not offer speaker note functionality.

GoToMeeting [40], by Citrix, lets users schedule, share, and join presentations over the internet. Users can talk with one another or see each other's camera feeds in order to make collaboration more simple. The presenter can share web pages, presentations, or other document types with other users. Joining other presentations is free, but giving presentations requires a monetary subscription after a free thirty-day trial. No speaker notes are available for the user while presenting, and the user cannot create any type of document in the application.

None of the applications discussed in this section allow for pagination of speaker notes, changing the speaker note font size while presenting, changing the presentation or speaker note language independently of one another, or hot-swapping of certain slides.

Chapter 3

ShareSynch iOS Development and Implementation

ShareSynch is an extensive presentation software system that is currently available for Windows and Mac OS X systems. The iOS version of ShareSynch took the core features of the PC software and implemented them on the iOS platform. The application allows for the presentation of ShareSynch materials on an external display via an Apple TV or a VGA/HDMI cable. Users can edit sermon slides and speaker notes from within the application. New features that are not currently available in the PC software have been implemented for this project, including the ability to download sermon series from the internet, a new file format, and dynamic PDF generation of sermon slides and notes. The new file format and its corresponding database are used to store all sermon series and sermon information, including pictures, videos, and notes. Over fifteen external libraries were utilized in the development of the application, and the application contains about 13,000 lines of non-library code. In this chapter, we give a summary of the application and discuss the development approach, task delineation, and final deliverables for the iOS version of ShareSynch.

3.1 Application Summary

The ShareSynch iOS application has direct support for the iPad Air, iPad Mini, and iPod Touch 5th generation devices. Tangential support is provided for the iPhone 5, iPad 2, iPad with Retina Display, and a small number of other iOS devices. Each iOS device must run iOS 7.1 or 8.x, and the device needs to be compatible with an adapter that allows for VGA or HDMI output. The application supports external displays via these adapters or via an AirPlay device, such as an Apple TV.

Upon opening the application for the first time, users have access to a demo sermon series that contains a single sermon. Users can gain access to more sermon series by subscribing with ShareHim to sermon series; once subscribed, users can download new sermon series via resumable downloads over the internet or install individual sermons via iTunes application management. A user can have multiple series installed at one time. Before entering presentation mode, users can adjust the slide and speaker note language or create variations of a sermon series.[1] A user can also dynamically generate sermon PDF files, which show the user's customized speaker notes and slides.

A speaker can begin presenting a ShareSynch sermon after loading that sermon in presentation mode. Both images and videos are shown on the external display. The presentation interface shows a scrolling list of slide preview images, current presentation progress in terms of the number of visible slides, speaker notes, and a toolbar that lets a speaker make quick changes to presentation settings. The

[1] A sermon series variation is a copy of a given sermon series that contains altered versions of sermon slides or speaker notes.

presentation can be advanced via a Bluetooth remote or keyboard or by tapping in or swiping over the speaker notes. Speakers can quickly jump to any slide in the presentation by tapping the preview image for that slide. Speaker notes take up the majority of the presentation mode screen; note font size can be changed from the toolbar with a simple tap of the increase or decrease font buttons. Notes for each slide are paginated, and there is a visual indicator on the screen when the user has reached the last page of speaker notes. When a speaker attempts to advance the presentation, the application first checks to see whether the last page of notes are being displayed for the current slide. If the last page is being shown, the sermon is moved to the next slide. If there are more notes to show, the speaker notes are smoothly scrolled to the next page, and the next slide is not shown. The external display smoothly animates from one image or video to the next when appropriate.

An appeal may be given at the end of a sermon. Three appeal types exist: vocal, instrument only, or text only. Vocal and instrumental appeals project videos, and a text-only appeal projects a single image. As the presenter advances the speaker notes for an appeal, text is displayed on top of the projected video or image in order to communicate with the congregation. The appeal song selection and type can be changed from the presentation mode toolbar. When a user changes the appeal song selection or type, the application automatically swaps the appropriate slides into and out of the presentation; the user does not need to enter the editing interface to make changes to the appeal selection and type. Appeal videos are downloaded and installed via the application's sermon series download interface.

The user can access the sermon editing capabilities from the presentation interface for that sermon. Speaker notes can be edited in rich text with the following options: bold, italic, underline, paragraph indentation, and bulleted lists.

Slides can be hidden or rearranged, and new slides can be created from pictures or videos on the user's device. Slides can also be duplicated or imported into the current sermon from other sermons within the same sermon series. Changes to the sermon are saved automatically, and undo/redo buttons are available for the user should they need them.

One unique feature of the iOS version of ShareSynch is the ability to dynamically draw text on top of images. The same image can be used for the background of a large number of slides, and each of these slides can have text drawn on top of its image. This feature will allow ShareSynch series creators to design one slide image to be used across a large number of slides and then designate text and text attributes for that slide separately. Text in a specific language is drawn onto the image at runtime. Text attributes include position, width, height, font name, font size, and font color. When this capability is utilized, the application can use one image for many different slides and languages, which drastically reduces the sermon storage cost when compared to storing a unique image for every slide in every language.

Details on all ShareSynch features and requirements appear in the ShareSynch Requirements Specification located in Appendix A.

3.2 Development Approach

The ShareSynch iOS application was implemented as a native application in Objective-C in order to take advantage of all available device functionality [21, 22]; the application was built for both the iPod Touch and iPad platforms. We made substantial use of the Model-View-Controller design pattern [41] when organizing code, and we also used the Delegation design pattern [42] for communication between different objects. We utilized the recommendations set forth in the iOS App Programming Guide [43] for general guidelines and principles to use while developing the application. The application does not give any user information to third parties; does not gather personal user information outside of a user-supplied name and email address, which are used for subscription purposes; and only performs data gathering on the length of time that sermons are presented for sermon copyright purposes. ShareSynch uses the same web protocols as the PC software when requesting information from ShareHim servers. The HTTP protocol is used when uploading usage statistics, verifying a user's subscription, and requesting sermon download information. The sermon download information returned by ShareHim servers defines whether files should be downloaded via HTTP or HTTPS.

All pictures, videos, and other pertinent non-user data files are stored in the Application Support directory. This directory is not directly accessible to users and thus prevents users from easily destroying or obtaining any application data files. Keeping data files in their own directory also allows us to add the com.apple.MobileBackup attribute to the Application Support folder, which prevents the backup of large data files through iCloud or iTunes [44]. Pictures, videos, and other sermon series content are housed in large archive files; archive files allow for file compression, which helps to save valuable storage space. Archive files are encrypted, which safeguards copyrighted sermon images and videos should the user somehow gain access to the Application Support directory. The SQLite database that houses series and sermon information is also stored in the application's Application Support folder. iTunes application management lets the user copy data files into the application's Documents directory, which is the application's only publicly accessible directory. Once sermon install files are copied into this directory via iTunes, the application can detect the presence of these files and move them into the private Application Support directory. Once install files have been moved to the Application Support directory, they are available for installation from the application's sermon installation interface.

The ShareSynch application utilizes an SQLite [45] database, and the database schema can be seen in Figure 3.1. The database stores much of the application data, including sermon series information, activation data and their corresponding expiration dates, speaker notes, and user variations.

We have utilized over fifteen external libraries and other open-source projects in order to speed up development. Most libraries are licensed under the MIT license [46], and three libraries are licensed under either the BSD 2-Clause or BSD 3-Clause license [47, 48]. The ShareSynchIO library is proprietary and is not freely available online. Several of the libraries were tweaked and edited in order to fix bugs, finish incomplete features, and otherwise improve the libraries for use in ShareSynch. Table 3.1 lists each of these libraries, their purpose in the application, and whether or not ShareSynch uses a modified version of these libraries.

3.3 Task Delineation

The ShareSynch project was broken into seven major task areas for development and testing. A summary of major tasks, including estimated work hours and date of completion for each task, can be seen in Table 3.2. Most requirements in the requirements document were assigned a task number corresponding to the tasks listed in this table, as seen in Appendix A. Between August 25, 2014, and February 6, 2015, work progressed at the rate of approximately 20 hours per week. Due to delays in receiving official ShareSynch install files from the client as well

Table 3.1: Libraries and Open-Source Projects Used in ShareSynch iOS

Name	Purpose in ShareSynch iOS	Manually edited?
AFNetworking [49]	Allow for resumable downloads and download statistics	
CTAssetsPickerController [50]	Let users import pictures or videos from their device into the application	✓
DraggableCollectionView [51]	Drag and drop UICollectionView cells	
DTCoreText [52]	Converts HTML text to NSAttributed-String objects	
FileHash [53]	Generates an SHA-384 hash for file integrity checks	✓
iCarousel [54]	Slide preview images and carousel for the present screen	
IQKeyboardManager [55]	Fixes errors with keyboards and text input on small screens	
JNKeychain [56]	Easy-to-use API for loading and storing keys into a device's keychain	
KSLabel [57]	Generates a black border around text so that it can be easily seen against any background	✓
MGBoxKit [58]	Trims characters from the front and back ends of an attributed string	
MRProgress [59]	Displays a graphical progress view during potentially long operations	✓
MSCellAccessory [60]	Allows UITableViewCells to have a black disclosure indicator	
Objective-Zip [61]	Allows for the reading and writing of encrypted or unencrypted zip files	
Reachability [62]	Checks to see if the user has an active internet connection	
RichTextEditor [63]	Provides the user with tools to edit large blocks of text in rich text	✓
ShareSynchIO	ShareHim API that reads ShareSynch iOS install files that have been exported from the PC version of ShareSynch	✓

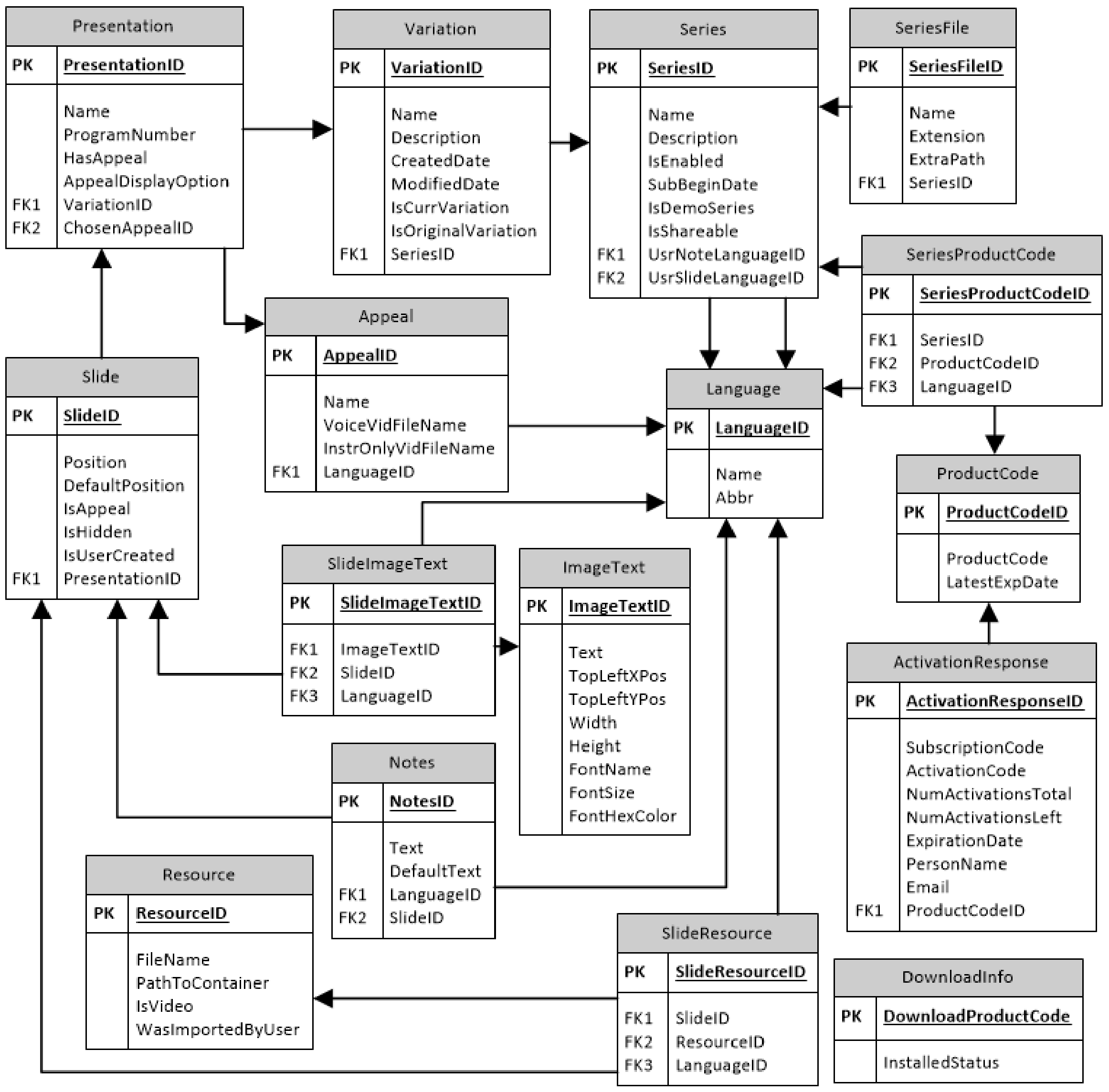


Figure 3.1: The database schema for the ShareSynch iOS application allows for the storage of all sermon series information and user data.

as new requirements added during the course of the project, the order of task implementation was not completed in the proposed order. After the third task was completed, the rest of the tasks were worked on simultaneously, which caused some tasks to be finished after their estimated date of completion. This section

Table 3.2: Task Delineation for the Development of the ShareSynch iOS Application

Task	**Estimated Work Hours**	**Estimated Date of Completion**	**Actual Work Hours**	**Actual Date of Completion**
Presentation Mode	100	August 22, 2014	104	August 11, 2014
File and Database Implementation	70	September 15, 2014	64	September 4, 2014
Sermon Editing System & Settings	160	November 10, 2014	99	October 9, 2014
Sermon Slideshow Installation	80	December 08, 2014	313[2]	December 29, 2014
UI Design and Implementation	60	December 29, 2014		December 4, 2014
Beta and User Testing	60	January 19, 2015		January 30, 2015
Final Changes, Bug Fixes, Testing, and Documentation	40	February 2, 2015		February 2, 2015
Project Completion	**570**	**February 2, 2015**	**580**	**February 2, 2015**

[2]The final four tasks were worked on simultaneously.

details items completed in each task along with the features available for users when a given task was completed.

3.3.1 Presentation Mode

The first task of development was the creation of the speaker's presentation view. The application's presenter view lets the speaker project slides and video onto an external monitor while viewing notes and slide thumbnails on the iOS device. This task included initial project and source control setup, displaying and scrolling speaker notes and slides, overlaying text on an image, and playing an appeal video with text on top of the video.

3.3.2 File and Database Implementation

The second task included the creation of the application database and the implementation of file formats, file storage, and file manipulation. Sermons, slides, variations, and all other data were loaded as appropriate from data files and the database once this task was completed. The user did not see many obvious changes after this task was completed because most changes occurred in the application backend. Because the final file and database implementation details were not finalized before project coding began, this task was placed after presentation mode implementation.

3.3.3 Sermon Editing System & Settings

The sermon editing system was the most intricate part of developing the ShareSynch application. Once the application's editing capabilities were available, the user was able to edit slide notes in rich text; move slides from one position to another; and add, hide, or remove slides from a given sermon. The speaker could also undo or redo their changes to a sermon. Dynamically generated PDF files of the user's edited sermons could be generated; these PDFs contain the speaker's own notes along with their corresponding slides. The PDF files have intelligent pagination, which attempts to minimize the amount of white space on each page of the PDF. The settings page for the application was also created during the course of this task.

3.3.4 Sermon Slideshow Installation

When this task was completed, users were able to download a sermon series from ShareHim servers or install sermons via iTunes after subscribing to a sermon series. When downloading a sermon series over the internet, the user cannot utilize any individual sermon from a series until the whole series has been downloaded and installed. A demo sermon was created for users who have not yet subscribed to any sermon series so that they could still become familiar with the application's features.

3.3.5 UI Design and Implementation

Because the application was created for both iPod Touch and iPad devices, a task was specifically devoted to the creation and implementation of graphical user interfaces that functioned properly on widely varying screen sizes. User interfaces were designed according to the interface guidelines given by Apple for iOS applications [64]. Two main interfaces were created: one for the larger iPad screen size and density and a second for the significantly smaller iPod Touch. The application manipulates these interfaces at runtime to adjust for differing screen sizes and densities. Once the UI design and implementation task was completed, users were able to move through the full ShareSynch application via an intuitive and clean interface on all directly-supported device types and in all device orientations. Several screenshots of the finished application can be seen in Appendix B.

3.3.6 Beta and User Testing

The application underwent thorough testing and evaluation. All features were tested; many optimizations and bug fixes were made. Along with GUI and feature testing, this testing phase also included device testing, battery-consumption testing, unit testing, user testing, and acceptance testing.

3.3.7 Final Changes, Bug Fixes, Testing, and Documentation

This final task enveloped the final details and work of the project. Documentation was created as necessary, any last-minute bugs were fixed, and more testing was completed. Documentation included comments above every function that was not part of an external library, file format documentation, and general project documentation. Over seven thousand lines of comments were written in and above ShareSynch functions. When this task was completed, the application became ready for use by the public.

3.4 Final Deliverables

Once the project was completed, we delivered the following materials to ShareHim:

- Application source code
- Administrative access to project source control
- Documentation on the project's setup and installation as well as its organization

- File format documentation
- Final project report

Chapter 4

Testing & Evaluation Plan

A large variety of test types were executed during the course of ShareSynch's development. Xcode's XCTest framework, the iOS simulator, and physical devices were used to perform unit tests, battery usage analysis, and device testing. Other test types were executed in Instruments, a program included in Xcode for performance and graphical testing [65].

Unit tests were created within the native Xcode XCTest framework. These black-box tests focused on data model functionality, and they ensured that application data was properly loaded, modified, and saved. Along with unit tests for data model classes, additional unit tests were created for a small number of custom data structures, such as the structure that controlled undo and redo actions on the edit page. The XCTest framework was also used to create performance tests. Performance tests in the XCTest framework measure how long, in seconds, a block of code takes to execute. The block of code is executed ten times, and data on the average time taken as well as each individual test time is provided to the tester. ShareSynch has several areas where the application must perform tasks within a given time frame so that the application does not feel sluggish to the user. Table 4.1

shows these areas and their maximum time for completion on the iPad Air. If the application could not perform one of these tasks in the given time frame, the functionality was adjusted so that the user could continue to utilize the application while lengthy tasks were running. For example, if the entire sermon cannot be loaded within ten seconds on average, the application should show the available sermon content after ten seconds and continue to load the rest of the sermon in the background. Even if a task can be performed in the required time, however, a user is shown a progress meter of some type so that they know the application is not frozen.

The Instruments application [65] was used to perform memory leak and random input tests as well as a small amount of automated GUI tests. During an application's execution, Instruments can inform a tester of memory leaks that occur throughout the normal course of an application's usage. Once a memory leak has been detected, Instruments tells the developer which function the memory leak occurred in or where the creation of the leaked object took place. We aimed to have no memory leaks in our own application code. Memory leaks could occur within Apple's framework code, but because this code was out of our control, we could not prevent these leaks. ShareSynch must also not crash. In order to facilitate the random inputs and events that may lead to ShareSynch crashing, we used the UI AutoMonkey framework [18], which executes inside the Instruments software. The UI AutoMonkey framework gives an iOS application rapid, random events and inputs in order to test for application stability.

One of ShareSynch's requirements states that it must be able to present multiple hour-long sermons on a single battery charge. A script was created to simulate the user presenting a slide show, and this script was run for two hours in order to test for battery usage. The slide show used for testing contained images, videos, and

Table 4.1: ShareSynch Performance Testing

Item	**Maximum Time for Completion on iPad Air**
Saving Changes to Sermon Order	250 milliseconds
Saving Changes to Slide Visibility	250 milliseconds
Saving Note Edits	250 milliseconds
Application Boot	2 seconds
Loading Sermon Series Details	2 seconds
Saving New Slides	2 seconds
Creating a Variation	5 seconds
Restoring a Sermon	7 seconds
Loading Sermon	10 seconds

an appeal, which allowed each slide type to be used during the course of the test. Before testing began, the device was connected to an external display via AirPlay so as to simulate a real presentation as much as possible. Battery tests passed if there was at least 15% battery left after two hours of the presentation simulation. Devices were first tested at full screen brightness, and if a device did not meet the minimum battery charge requirement, the device was subsequently tested at half brightness and minimum brightness.

Along with the aforementioned test types, a mobile application must be tested in its usage environment and on all directly-supported device types [24]. Device testing ensured that the application ran efficiently on each device type and also ensured that the GUI scaled appropriately on each directly supported screen resolution. Device testing did not take place on tangentially supported devices. User and acceptance testing were performed after all features have been fully implemented in the application. These final two types of testing, along with some device testing, were performed by ShareHim employees. Tests run by ShareHim employees verified that the application met not only its listed requirements but also its functional use by speakers.

Chapter 5

Testing and Evaluation Results

Over the course of unit testing, 85 test functions were created. Each of these test functions ensured that one or more functions of a data model or data structure class executed correctly or performed as required. Several bugs were discovered and fixed because of these tests, including a problem with deleting the current variation, an issue with duplicating a slide available in multiple languages, and a bug with loading a series by its name. After the tests were created, they served as helpful regression tests when more functionality was added; the regression tests caught several errors that were created in the extended functionality.

Performance test results can be seen in Table 5.1. As shown in the table, every single performance metric was well beneath the maximum time for completion. Unfortunately, performance metrics for boot time are impossible to measure with the XCTest framework and cannot accurately be measured by Xcode log timestamps [66], so we tested boot time using a handheld stopwatch. Boot time started when we tapped on the home screen icon for ShareSynch, and time ended when the main menu became visible to the user. Ten boot time tests were executed, and the average time is reported in the table. The test for saving changes to a

Table 5.1: ShareSynch Performance Testing Results

Item	Maximum Time for Completion on iPad Air	Actual Time for Completion on iPad Air
Saving Changes to Sermon Order	250 milliseconds	188 milliseconds
Saving Changes to Slide Visibility	250 milliseconds	142 milliseconds
Saving Note Edits	250 milliseconds	2 milliseconds
Application Boot	2 seconds	0.99 seconds
Loading Sermon Series Details	2 seconds	0.84 seconds
Saving New Slides	2 seconds	0.53 seconds
Creating a Variation	5 seconds	1.77 seconds
Restoring a Sermon	7 seconds	2.17 seconds
Loading Sermon	10 seconds	5.02 seconds

sermon order was made after moving a slide ten positions, and the slide visibility test was executed with a change of ten different slides' visibility. Twenty-six sermons were installed for both the "Loading Sermon Series Details" and the "Creating a Variation" tests. Times for restoring a sermon were gathered manually using the timestamps provided in Xcode log statements, and the sermon restoration process had to account for two user-created image slides, one user-created video slide, three rearranged slides, and two hidden slides. With the exception of the boot time and sermon restoration tests, all unit and performance tests were fully automated.

Memory leak tests were performed several times throughout the course of development, and a small number of memory leak errors were fixed in a third-party library. No known memory leaks exist in our application code; however, several small memory leaks still exist in the application. These leaks occur because of Apple-provided code, so the remaining leaks are unfixable at this point in time. These leaks are usually between sixteen and forty-eight bytes and are never above 240 bytes. During a short fifteen minute test of various portions of ShareSynch, the application had fifty-one small memory leaks in Apple-provided code. The UI AutoMonkey library was used to test a great deal of quick, random input throughout the application. This library helped to discover several race conditions that caused application crashes in the download functionality of the application.

Most GUI testing was primarily performed manually rather than via the Instruments application. Creating effective GUI tests in Instruments, even with the help of the Tuneup JS framework [67], is difficult and extremely time-consuming. We used the Instruments GUI testing framework by manually creating a small GUI test that performed several tests on the main menu, presentation, and edit screens. The record and play back functionality of Instruments was used to assist in the development of this GUI test. We found that Instruments often refused to run any tests after a single code syntax error occurred until Instruments was restarted, even if the cause of the error was removed. Instruments does not restart the application between tests or after tests have completed, so a tester must manually reset the application by themselves or create code to manually perform resets in between each test. The lack of code completion and syntax highlighting within Instruments makes development of tests tedious. Because of these difficulties, GUI tests were performed manually on all directly-supported devices and the iOS simulator. These manual tests analyzed each screen on both iOS 7 and iOS 8 as well as in both the landscape and portrait device orientations. Two ShareHim employees assisted us with this task, and we discovered several faulty graphical areas of the application during the course of testing, including incorrectly sized labels on small devices and text truncation errors.

GUI testing also brought about new ideas regarding the layout of the main menu and select presentation screens of the application. The main menu was substantially changed from its original version, and the select series screen and the settings screens were both merged into the main menu. The select presentation screen was tweaked to have a smaller amount of whitespace so that more content could be displayed at one time. Screenshots of the new and improved GUI are shown in Appendix B.

Battery tests were performed on ShareSynch's three directly-supported devices: the iPad Air, the iPad Mini, and the iPod Touch 5th Generation. Each of the devices was fully charged before test execution. Table 5.2 shows the results of battery tests when each device's screen brightness was set on the maximum level. Each device passed its battery test at maximum brightness, so subsequent battery tests with a less-intense brightness did not have to be performed. The iPad devices are able to handle several additional hours of presentation time after a two-hour test, but the iPod Touch should be used for a maximum of two hours before having its battery recharged.

Table 5.2: ShareSynch Battery Test Results

iOS Device	**Starting Battery Percentage**	**End Battery Percentage**
iPad Air	100%	64%
iPad Mini	100%	69%
iPod Touch 5th Generation	100%	19%

User and acceptance tests were performed by several ShareHim employees. In combination with our own device tests, these tests helped to unearth several new bugs in the iOS application. Once employees had used ShareSynch on their own devices, the ShareHim employees made several feature changes and requests, including the ability to delete activations and the ability to overwrite previously installed sermons. A full list of change orders can be seen in Appendix A.4. After the desired changes were made to ShareSynch iOS, ShareHim employees accepted the application for submission to the iTunes App Store.

Chapter 6

Conclusion

ShareSynch is a proprietary presentation application currently available for Windows and Mac OS X systems. Unique features of ShareSynch include the ability to have different languages for sermon slides and speaker notes, pagination of speaker notes in various font sizes, and the easy adjustment of appeal selections. No currently known iOS application mirrors these unique and essential features. The iOS version of ShareSynch was developed in Objective-C and is compatible with the iPad Air, iPad Mini, and the iPod Touch 5th Generation with tangential support for several related devices; the iOS application also introduces several new features, including a new file format and storage scheme that allows for the ability to draw text onto an image so that one image can be used for many different slides, downloadable sermon series, and dynamically generated PDF documents that contain user-edited sermons and speaker notes. We finished developing and testing the application on February 2, 2015; the final project contains over 13,000 lines of code and over 7,000 lines of code documentation. Source code, documentation, and the final project report have been delivered to ShareHim. ShareSynch became publicly available in the iOS App Store on March 2, 2015. The application will see further enhancements in future versions of the product including the ability to install variations and playback high resolution videos.

Appendix A

Requirements Specification

A.1 Task Delineation

Most of the requirements in Section A.3 have been assigned to one of the following task areas for organizational purposes. The applicable task area is found in brackets before the requirement is given.

1. Presentation Mode
2. File and Database Implementation
3. Sermon Editing System & Settings
4. Sermon Slideshow Installation
5. UI Design and Implementation
6. Beta and User Testing
7. Final Changes, Bug Fixes, Testing, and Documentation

A.2 Hardware Requirements

- The application shall have direct support for iPod Touch 5th Generation, iPad 2, and iPad Air with tangential support for iPhone 5, iPhone 5S, iPhone 5C, iPad Mini, iPad Mini with Retina Display, iPad with Retina Display.
- The iOS device shall run iOS 7.1 or 8.x.
- The iOS device shall be compatible with an adapter that allows for VGA or HDMI output.

A.3 Application Requirements

A.3.1 General

- [1] Upon opening a sermon in presentation mode, the application shall ask users to turn off battery-consuming services when the network can be reached.
- [1] Upon opening a sermon in presentation mode, the application shall ask users to turn off battery-consuming services when the network can be reached.
- [1] The application shall automatically remove the warning to turn off battery-consuming services after 5 seconds.
- [5] The application shall support both landscape and portrait layouts in all modes.
- [2] The application shall support separate different languages for application text, sermon notes, and presentation slide content.

- [2] The application shall support JPEG or PNG formats for presentation slide images.
- [1] The application shall support presentation slides that are text overlays on images. The application shall support overlaying text of a given font, font size, and font color onto an image at a given starting point, width, and height for demoing purposes.
- [2] If an image or video uses the overlay text feature, the image or video shall not be tied to a specific language so that multiple languages can use the same image or video.
- [4] Any sermon series downloaded or installed shall be verified for integrity via an accompanying checksum or hash.
- [4] The application shall check for newly downloaded (via iTunes or WiFi) sermon series upon launch and install all new series upon launch.
- [6] The application shall support the presentation of multiple hour-long sermons on a single battery charge when battery-consuming services (such as WiFi and cellular data) are turned off.
- [1] The application shall be compatible with a Bluetooth remote for advancing slides.
- [2] The application shall be compatible with multiple sermon series.
- [1] The application shall allow for presentations via an external video adapter and through AirPlay.
- [4] The application shall verify that the users subscriptions are valid upon opening the application.

- [1] When transitioning between slides, the application shall smoothly fade from one slide to the next during the transition.
- [1] By default, the application shall mirror the users screen when connected to an external screen.
- [1] The application shall start a timer when the user enters presentation mode.
- [1] When the user leaves presentation mode, the application shall end the timer. If the timer length is 20 minutes or longer, the presentation time length and name of the presented sermon shall be saved in a JSON format.
- [4] The presentation time length along with the presented sermon name shall be uploaded to ShareHim servers once every seven days.

A.3.2 Download & Installation

- The user shall be able to download the ShareSynch software for free from the iTunes app store.
- [4] When opening the application for the first time, the application shall prompt the user for their activation code or allow the user to use the demo sermon series.
- [4] A successful activation code entry shall result in access to a specific set of sermon series that the user can now download or install.
- [4] When downloaded from the app store, the app shall contain a single sermon series entitled Demo that contains a single sermon for demo purposes.

- [4] The demo sermon series shall not be available after the user has downloaded a complete sermon series from ShareHim.
- [4] The application shall verify the users information with the ShareHim website before granting access to non-demo sermon series downloads when an internet connection is available.
- [4] If an internet connection is not available, the application shall prompt the user for an activation code, which can be manually obtained from ShareHim via phone or the internet. Any information that the user will need to give ShareHim for activation shall be displayed along with this prompt.
- [4] The user shall have access to the Download Sermon Series page, which contains a list of ShareHim sermon series available to subscribe to. Series not currently available for the current user shall be grayed out.
- [4] The user shall be able to sort the list on the Download Sermon Series page by name or by available series.
- [2] Slides, pictures, audio, and video shall not be accessible from outside the iOS application. These resources shall be encapsulated in large archive files that are password protected so that they are not easily accessible from outside the application.
- [4] The user shall be able to install new sermon series through iTunes application management or wirelessly over WiFi and cellular data.
- [4] The user shall see the size of a sermon series before downloading it over WiFi and cellular data.

- [4] The user shall be able to view the amount of available space left in the application for sermon series from the Download Sermon Series page.
- [4] The application shall give a warning if there is not enough space on the device or in the application to download a given sermon series.
- [4] The user shall be able to start, pause, and resume the download of a sermon series at any time.
- [4] The user shall see the progress, speed, and estimated download completion time of the sermon series download on the Download Sermon Series page.
- [4] The user shall not be able to access any sermon of a series until the entire series has been downloaded and installed.

A.3.3 Opening and Managing a Sermon Series

- [2] Upon opening the app, the user shall be presented with the following options on the main menu: Load Last Used Sermon, Load Sermon, Download Sermon Series, and Settings.
- [2] The user shall be returned to slide 1 of their last-used sermon upon choosing Load Last Used Sermon on the main menu.
- [4] Selecting the Download Sermon option shall take the user to the Download Sermon Series page.
- [2] Upon selecting Load Sermon from the main menu, the user shall be provided with a list of available sermon series to choose from.

- [2] When the user chooses a sermon series, the user shall be taken to a new view that contains thumbnail previews of all sermons within that series. This view is the sermon selection screen.
- [2] From the sermon selection screen, the user shall see data on the selected sermon series (slide language, note language, etc.), picture thumbnails of each sermon in the series, and a toolbar at the bottom of the screen.
- [2] From the sermon selection screen toolbar, the user shall have the option to change the series language for slides.
- [2] From the sermon selection screen toolbar, the user shall have the option to change the series language for notes.
- [2] From the sermon selection screen toolbar, the user shall be able to enter variation management mode
- [2] The variation management mode shall show a list of all sermon variations for a given sermon series.
- [2] In variation management mode, the user shall be able to create a new variation based on the default, unedited variation.
- [2] In variation management mode, the user shall be able to create a new variation based on another variation.
- [2] In variation management mode, the user shall be able to delete variations of a sermon series, but shall not be able to delete the original, unedited variation.
- [2] In variation management mode, the user shall be able to switch the currently loaded sermon series to a different variation of that series.

- [3] From the sermon selection screen toolbar, the user shall be able to generate a PDF of a sermon. This PDF shall contain all user notes (as edited by the user) along with a small picture of each slide.
- [3] The user shall be able to print or email the generated PDF.
- [2] From the sermon selection screen, the user shall be able to choose a sermon in order to enter presentation mode.

A.3.4 Presentation Mode

- [2] The user shall see a loading progress indicator dialog when loading a sermon will take longer than half a second.
- [2] The user shall be able to cancel loading a sermon via the progress indicator dialog.
- [1] The application shall alert the user if there is no secondary display available or if a secondary display is disconnected while in presentation mode.
- [1] The speaker shall have button controls to control whether the external screen mirrors the current display or projects the sermon slides and videos.
- [1] The user shall see a thumbnail preview of the previous, current, and next slides along with the current slides notes.
- [1] The thumbnail previews of slides shall show that slide's position in the overall sermon.
- [1] The user shall be able to move forward (advance) and backward (go back) by swiping left or right, by touching the right or left sides of the screen, by

swiping right or left, or by tapping the next/previous buttons, as defined by a user-specified setting for advancing a sermon.

- [1] When sermon notes for a single slide do not fit on a single page during a presentation, the user shall see a visual indicator when there are more notes to display for the current slide.
- [1] When sermon notes for a single slide do not fit on a single page during a presentation, advancing will scroll the sermon notes for a single slide smoothly without moving the presentation to the next slide. If there are no more notes to show, advancing shall move the presentation to the next slide.
- [1] The user shall be able to skip to any slide by scrolling the thumbnail previews to the right (forward) or to the left (backward) and selecting a slide by tapping on its thumbnail.
- [1] When scrolling the thumbnail previews, the previews shall not wrap from end to beginning or vice versa.
- [1] The user shall be able to tap the thumbnail of the next or previous slide in order to instantly jump to the next/previous slide without smooth scrolling of notes.
- [1] The user shall be taken to the first slide after advancing on the final slide.
- [1] The user shall be taken to the last slide after going back on the first slide.
- [1] If the user is using an external keyboard or has a Bluetooth remote with number capabilities, the user shall be taken to a specific slide number after typing in the slide number and hitting Enter ($\hookleftarrow$).

- [1] If the user is using an external keyboard or has a Bluetooth remote with number capabilities, hitting the left and right arrow keys will move the sermon backward or forward, respectively.
- [1] The user shall see the number of the currently active slide in the current sermon along with the total number of slides in the current sermon in the following format: [# of current slide]/[total # of slides].
- [1] The user shall see a toolbar at the bottom of the screen.
- [1] The user shall be able to increase or decrease the font size of sermon notes from the toolbar.
- [3] The application shall remember any indentation as it increases font, decreases font, and performs line wrapping.
- [1] The user shall be able to change the appeal song for a given sermon from the toolbar.
- [1] The user shall be able to change the appeal options for a given sermon from the toolbar to the following options: vocal, instrument only, text only.
- [1] Changes to the appeal song or appeal song options shall not come into effect while the sermon is displaying the appeal. Changes shall come into effect the next time the appeal slide is shown.
- [1] In the preview of a video slide, the user shall see the word Video along with the length of the video in text on top of the preview for the video.
- [1] When a video is playing, the user shall see the length of time remaining for the video playback only on the users device.

- [1] Some slides may be appeal slides. An appeal slide shall have text dynamically written onto the slide that mirrors portions of the speakers notes. When the user advances the appeal notes, if there are more notes to display, both the notes on the users device and the text on the external display changes.

A.3.5 Edit Mode

- [3] The user shall be able to open sermons in edit mode from presentation mode.
- [3] The user shall be able to change slide notes, slide order, and which slides to display during a presentation in edit mode.
- [3] When editing a slides notes, the user shall have the following formatting options: bold, italic, underline, and bullets.
- [3] The user shall be able to use the rearrange slides mode while in edit mode.
- [3] In rearrange slides mode, the user shall be able to rearrange the slides of a sermon by dragging and dropping slides from one location to another while viewing thumbnail previews of all slides in a sermon.
- [3] The user shall see the numerical location of each slide in the lower right-hand corner of each slide preview.
- [3] The user shall be able to select one or more slides at one time to set them as hidden or shown when rearranging slides.
- [3] The user shall be able to insert new slides containing pictures or video from their local iOS device into the current sermon.

- [3] The user shall be able to insert new slides containing pictures, video, or slides from another sermon into the current sermon.
- [3] The user shall be able to duplicate a slide, including notes, at the current slide position.
- [3] The user shall be able to delete slides, including their accompanying resource, if the slide was created by the user and the resource was imported into the application by the user. If the resource was not imported by the user, only the slide shall be deleted.
- [3] When inserting new slides, the user shall be able to insert new slides before or after the current slide.
- [3] Changes to sermon order or contents shall be saved automatically upon the change being made.
- [3] Changes to sermon notes shall be saved automatically upon changing to a different slide in edit mode, when the user exits edit mode, or when the user leaves the app for any reason (such as via the home button).
- [3] The user shall see both the current slide and its notes while editing a slides notes.
- [3] The user shall have undo and redo buttons for the following change types: change of note text, hiding or showing a slide, changing a slides position, and adding a slide.
- [3] The maximum amount of times the user shall be able to undo or redo shall be 10 times.

- [3] The undo and redo information shall be lost when the user leaves edit mode.
- [3] The user shall be able to restore a slides notes to its initial contents.
- [3] The user shall be able to reset an entire sermon to its initial state. This operation shall be confirmed with the user before being performed. Resetting a sermon restores all slides to their default position, notes, and visibility. Resetting a sermon also deletes any custom slides created by the user.
- [3] The user shall be able to edit the text shown on top of appeal videos. This is done via the normal editing interface for notes. The notes on top of an appeal video may advance along with the speakers notes. The format for appeal text shall mirror the PC version of ShareSynch. Notes to be shown on top of the appeal video shall be inside a <C><c> tag. Advancing the notes on an appeal slide shall advance the text displayed on the external screen to the text within the next <C><c> tag, if available. A single <R> tag shall designate that the text on the external screen shall be removed, and any text after the <R> tag will be displayed as the final notes for the speaker.

A.3.6 Settings

- [3] The user shall be able to view and edit applications settings after selecting Settings on the main menu.
- [3] The user shall be able to view their series subscription details and activation code.
- [3] The user shall be able to view information about the mobile application and its creators.

- [3] The user shall be able to view information on how much storage space the application is using.
- [3] The user shall be able to permanently delete sermon series from their device.
- [3] The user shall be able to change their default method of advancing sermon notes (swipe, previous/next buttons, or touching the left/right side of the screen).

A.4 Change Orders

- The application shall directly support the iPad Mini.
- The iPad 2 shall be tangentially supported.
- The user shall be able to open a PDF for previewing without emailing or printing the PDF.
- In presentation mode, the user shall see each slide's position as well as the total number of slides in the presentation in the preview thumbnail for each slide. The format for slide positions shall be the following: [# of current slide] / [total # of slides].
- The user shall not have buttons available in presentation mode for advancing or going back through the sermon.
- In presentation mode, the user shall see a visual indicator when moving the slide show forward will move the presentation to the next slide. The user shall not see a visual indicator when there are more notes to display for the current slide.

- A playing video's audio shall fade in and out when the presentation transitions to or from that video.
- Available appeal videos shall be displayed as available in every sermon language regardless of the appeal video language.
- Available appeal videos shall be displayed in a table and sorted by language.
- The list of available appeal videos shall indicate to the user whether the listed appeal videos are available as vocal videos, instrumental videos, or both.
- The application shall not prompt the user for their subscription or activation code until they attempt to install a sermon for which they have no active subscription.
- The user shall be able to overwrite previously installed sermons via new and updated sermon install files.
- The user shall be able to have multiple activations for the same series.
- The user shall be able to access a list of current activations and their expiration dates from the activation screen.
- The user shall be able to quickly fill in the activation screen's text fields with data from a previous activation.
- The user shall be able to delete activations.
- The application shall have a help screen that gives the user information about utilizing the application and allows for a sound test.
- The user shall be able to download vocal appeal videos separately from instrumental appeal videos.

- Any downloads for the demo sermon series shall be designated as belonging to the demo by appending the string “(Demo)” to the title of a demo series download.
- The user shall be able to access the available series list from the main menu.
- The user shall be able to access the settings list from the main menu.
- The user shall be able to show or hide the lists of available series and settings from the main menu.
- From the about screen, the user shall see the amount of device space remaining, not the amount of space that is being used by the application.
- The user shall be able to access license information for open-source code that is utilized in the application from the about screen.
- The user shall see the expiration dates for all languages for a series from the Manage Sermon Series screen.
- The user shall be able to delete appeal videos from the settings portion of the application.
- When managing appeal video downloads, the user shall be able to delete installed appeal videos on a language-by-language basis and shall be able to delete vocal videos separately from instrumental videos.

Appendix B

Application Screenshots

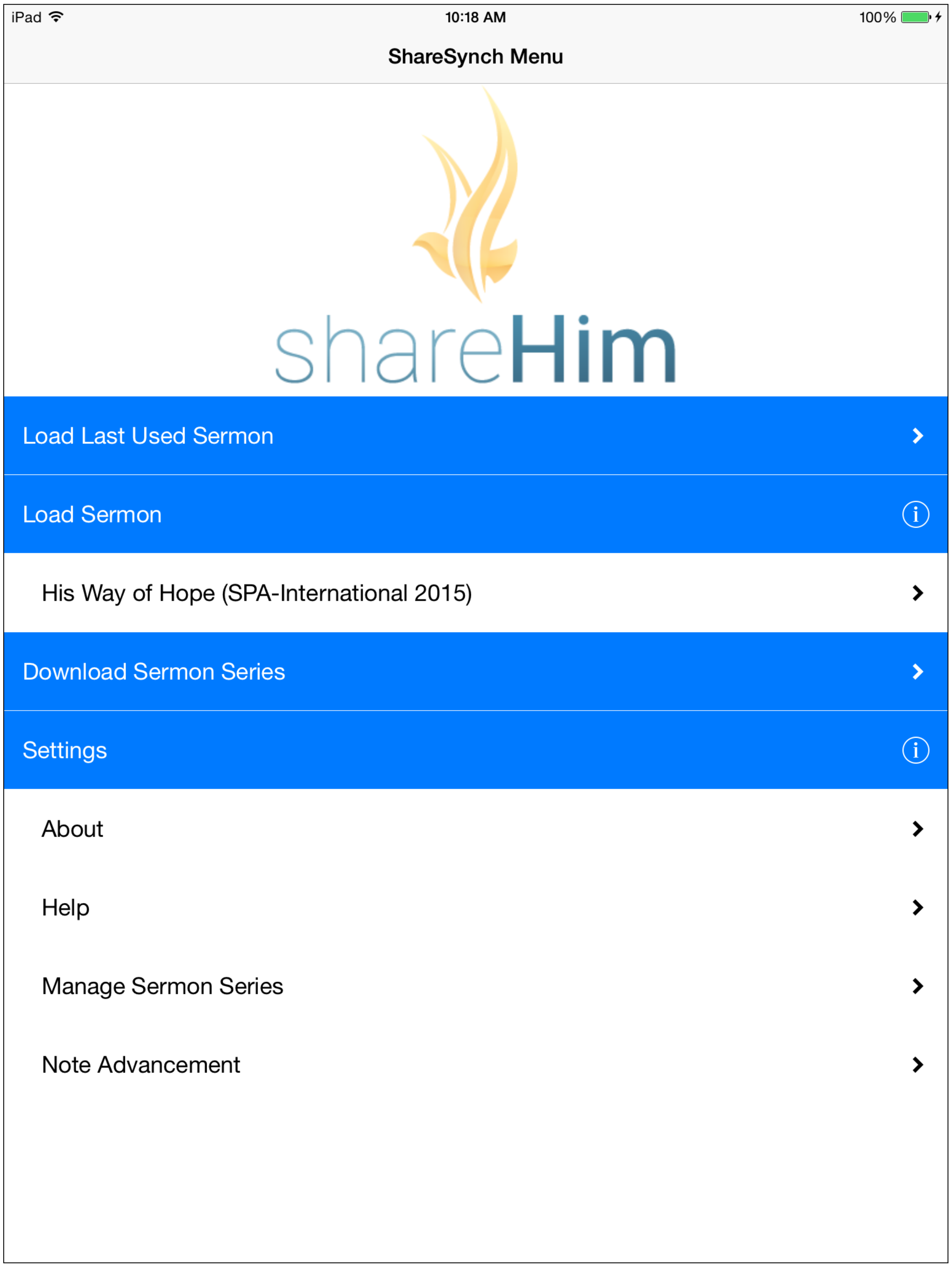
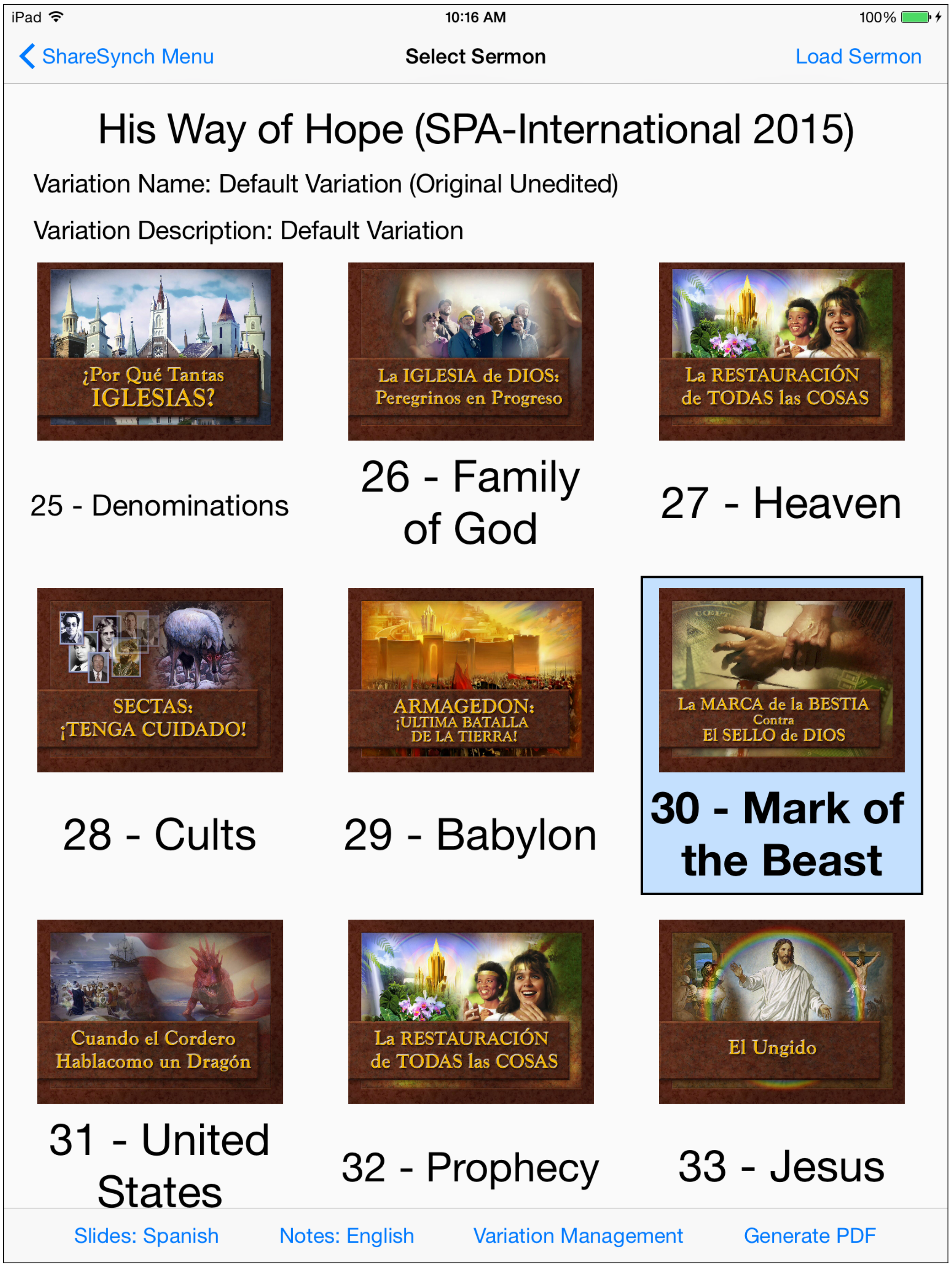


Figure B.1: ShareSynch's main menu screen lets users directly access available sermon series and settings without having to open another page. The sections for available sermon series and settings can be expanded or contracted as necessary.



iPad 10:16 AM 100%

ShareSynch Menu

Select Sermon

Load Sermon

His Way of Hope (SPA-International 2015)

Variation Name: Default Variation (Original Unedited)

Variation Description: Default Variation

¿Por Qué Tantas IGLESIAS?
La IGLESIA de DIOS: Peregrinos en Progreso
La RESTAURACIÓN de TODAS las COSAS
SECTAS: ¡TENGA CUIDADO!
ARMAGEDON: ¡ULTIMA BATALLA DE LA TIERRA!
La MARCA de la BESTIA Contra El SELLO de DIOS
Cuando el Cordero Hablacomo un Dragón
La RESTAURACIÓN de TODAS las COSAS
El Ungido

25 - Denominations

26 - Family of God

27 - Heaven

28 - Cults

29 - Babylon

30 - Mark of the Beast

31 - United States

32 - Prophecy

33 - Jesus

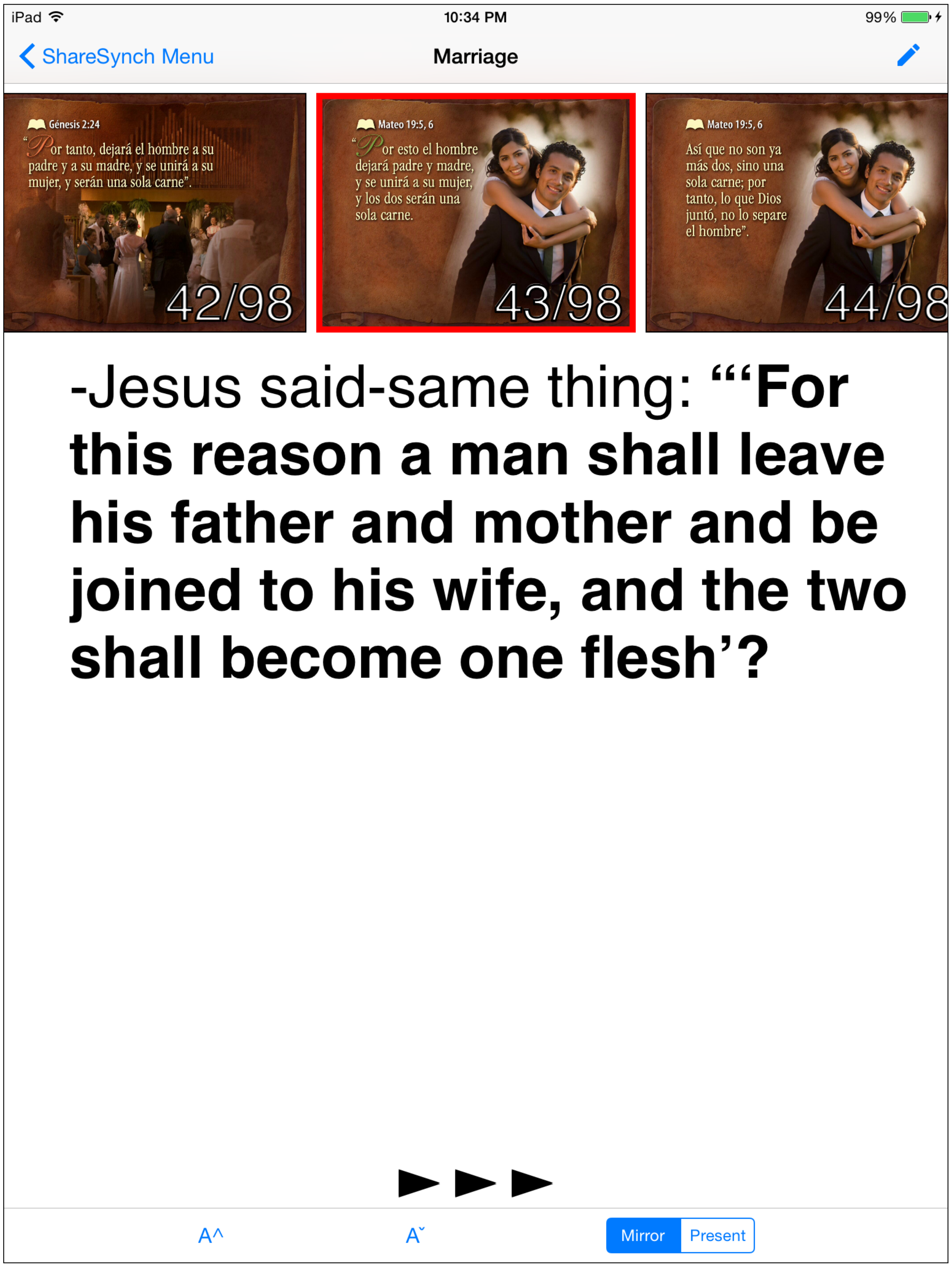
Slides: Spanish

Notes: English

Variation Management

Generate PDF

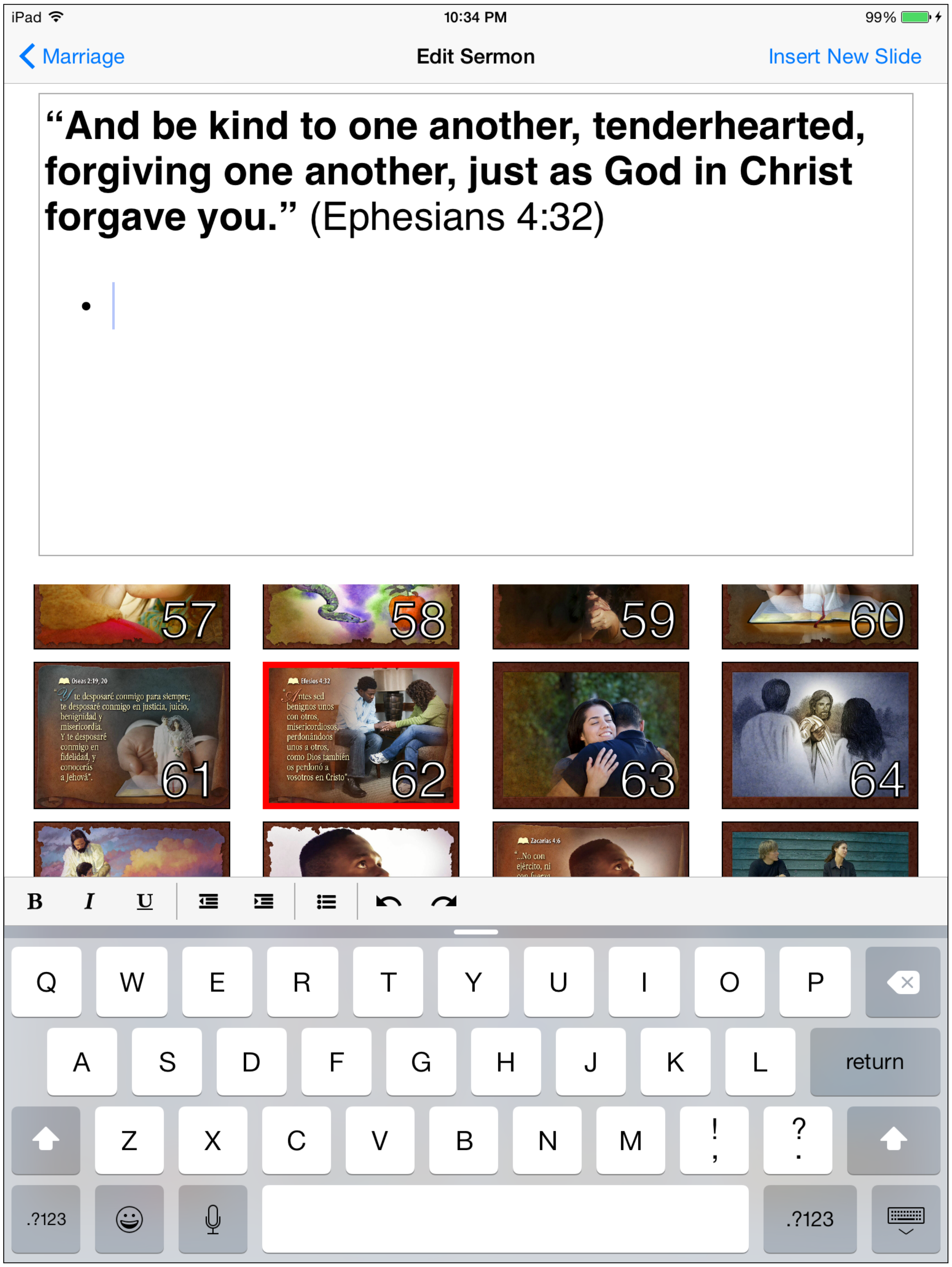
Figure B.2: The select presentation screen shows which variation is currently loaded and which sermons are currently available. The user can also change the slide language, change the speaker note language, or generate a PDF of a selected sermon from this screen. The currently selected sermon cell is highlighted in light blue, and its title text is set to a bold font.


iPad
10:34 PM
99%
ShareSynch Menu
Marriage
Génesis 2:24
"Por tanto, dejará el hombre a su padre y a su madre, y se unirá a su mujer, y serán una sola carne".
42/98
Mateo 19:5, 6
"Por esto el hombre dejará padre y madre, y se unirá a su mujer, y los dos serán una sola carne.
43/98
Mateo 19:5, 6
Así que no son ya más dos, sino una sola carne; por tanto, lo que Dios juntó, no lo separe el hombre".
44/98
A^
Aˇ
Mirror
Present

-Jesus said-same thing: "'**For this reason a man shall leave his father and mother and be joined to his wife, and the two shall become one flesh'?**

►►►

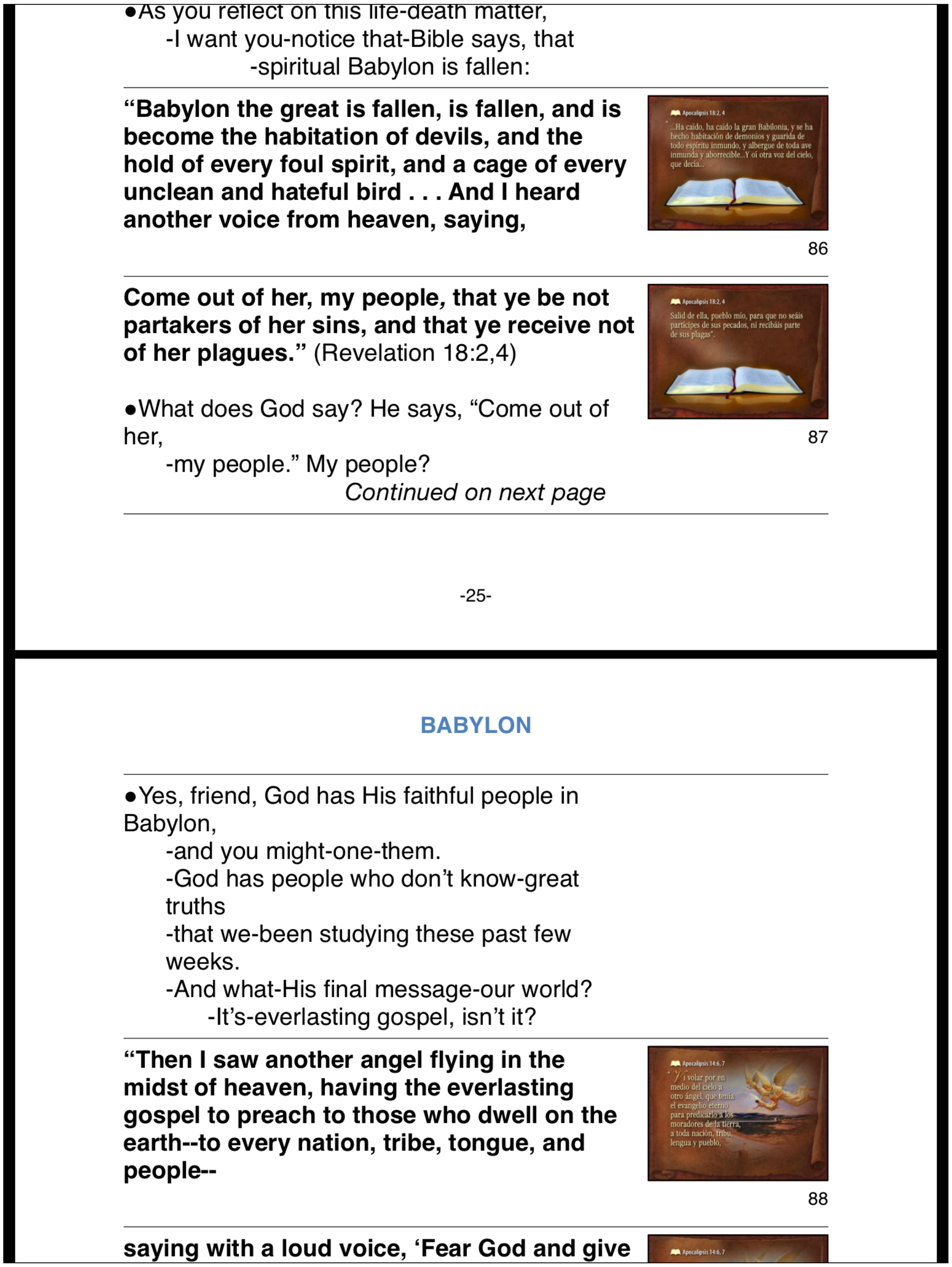
Figure B.3: Presentation mode prominently displays speaker notes for the user. Arrows indicate that advancing the speaker notes will move the presentation to the next slide. When a sermon has an appeal, appeal options are shown in the toolbar along with the change font and presentation mode settings.



“And be kind to one another, tenderhearted, forgiving one another, just as God in Christ forgave you.” (Ephesians 4:32)

•

Figure B.4: ShareSynch lets users edit a sermon presentation in a variety of ways. Notes are editable in rich text. Slides can be inserted from the user’s device or from another sermon within the same series. The user can make a variety of other changes to the presentation from this screen, such as changing sermon order or slide visibility.



●As you reflect on this life-death matter,
-I want you-notice that-Bible says, that
-spiritual Babylon is fallen:

“Babylon the great is fallen, is fallen, and is become the habitation of devils, and the hold of every foul spirit, and a cage of every unclean and hateful bird . . . And I heard another voice from heaven, saying,

Apocalipsis 18:2, 4
“...Ha caído, ha caído la gran Babilonia, y se ha hecho habitación de demonios y guarida de todo espíritu inmundo, y albergue de toda ave inmunda y aborrecible...Y oí otra voz del cielo, que decía...

86

Come out of her, my people*,* that ye be not partakers of her sins, and that ye receive not of her plagues.” (Revelation 18:2,4)

Apocalipsis 18:2, 4
Salid de ella, pueblo mío, para que no seáis partícipes de sus pecados, ni recibáis parte de sus plagas”.

87

●What does God say? He says, “Come out of her,
-my people.” My people?

Continued on next page

-25-

BABYLON

●Yes, friend, God has His faithful people in Babylon,
-and you might-one-them.
-God has people who don’t know-great truths
-that we-been studying these past few weeks.
-And what-His final message-our world?
-It’s-everlasting gospel, isn’t it?

“Then I saw another angel flying in the midst of heaven, having the everlasting gospel to preach to those who dwell on the earth--to every nation, tribe, tongue, and people--

Apocalipsis 14:6, 7
“Vi volar por en medio del cielo a otro ángel, que tenía el evangelio eterno para predicarlo a los moradores de la tierra, a toda nación, tribu, lengua y pueblo,

88

saying with a loud voice, ‘Fear God and give

Apocalipsis 14:6, 7

Figure B.5: ShareSynch lets users preview dynamically generated PDF documents before sharing them with others. The user’s customized speaker notes and sermon order are reflected in these documents. The page-breaking algorithm attempts to minimize the amount of whitespace at the end of each page by placing as much of a slide’s speaker notes on the current page as possible before inserting the rest of the notes on subsequent pages.

Bibliography

[1] "ShareHim programs," http://sharehim.org/about-us/, ShareHim, 2014. 1

[2] "Idc: Smartphone os market share 2014, 2013, 2012, and 2011," http://www.idc.com/prodserv/smartphone-os-market-share.jsp, IDC Corporate USA, 2014. 1

[3] "What qualifies an android "display"?" http://stackoverflow.com/questions/16745935/what-qualifies-an-android-display, Stack Overflow, 2013. 1

[4] "Dashboards," https://developer.android.com/about/dashboards/index.html, Google Inc., 2014. 1

[5] "DisplayManager," http://developer.android.com/reference/android/hardware/display/DisplayManager.html, Google Inc., 2014. 1

[6] "iOS: About apple digital av adapters," http://support.apple.com/kb/HT4108, Apple Inc., 2014. 1

[7] "App store distribution," https://developer.apple.com/support/appstore/, Apple Inc., 2014. 1

[8] "Multiple display programming guide for iOS," https://developer.apple.com/Library/ios/documentation/WindowsViews/Conceptual/WindowAn

dScreenGuide/UsingExternalDisplay/UsingExternalDisplay.html, Apple Inc., 2012. 1

[9] L. Brutschy, P. Ferrara, and P. Müller, "Static analysis for independent app developers," *SIGPLAN Not.*, vol. 49, no. 10, pp. 847–860, Oct. 2014. [Online]. Available: http://doi.acm.org.ezproxy.southern.edu/10.1145/2714064.2660219 2.1

[10] A. I. Wasserman, "Software engineering issues for mobile application development," in *Proceedings of the FSE/SDP workshop on Future of software engineering research*. ACM, 2010, pp. 397–400. 2.1, 2.3

[11] H. Muccini, A. Di Francesco, and P. Esposito, "Software testing of mobile applications: Challenges and future research directions," in *Proceedings of the 7th International Workshop on Automation of Software Test*, ser. AST '12. Piscataway, NJ, USA: IEEE Press, 2012, pp. 29–35. [Online]. Available: http://dl.acm.org.ezproxy.southern.edu/citation.cfm?id=2663608.2663615 2.1

[12] J. Dehlinger and J. Dixon, "Mobile application software engineering: Challenges and research directions," in *Workshop on Mobile Software Engineering*, 2011. 2.1, 2.2

[13] B. N. Nguyen, B. Robbins, I. Banerjee, and A. Memon, "Guitar: An innovative tool for automated testing of gui-driven software," *Automated Software Engg.*, vol. 21, no. 1, pp. 65–105, Mar. 2014. [Online]. Available: http://dx.doi.org.ezproxy.southern.edu/10.1007/s10515-013-0128-9 2.1

[14] "Calabash," http://calaba.sh/, Xamarin Inc., 2015. 2.1

[15] "Uiautomation," https://github.com/kif-framework/KIF, KIF Framework, 2015. 2.1

[16] "Xamarin test cloud," http://developer.xamarin.com/testcloud/, Xamarin Inc., 2015. 2.1

[17] "Uiautomation," https://developer.apple.com/library/ios/documentation/DeveloperTools/Conceptual/InstrumentsUserGuide/UsingtheAutomationInstrument/UsingtheAutomationInstrument.html, Apple Inc., 2014. 2.1

[18] J. Penn, "UI AutoMonkey," https://github.com/jonathanpenn/ui-auto-monkey, 2013. 2.1, 4

[19] B. Biel and V. Gruhn, "Usability-improving mobile application development patterns," in *Proceedings of the 15th European Conference on Pattern Languages of Programs*. ACM, 2010, p. 11. 2.2

[20] B. Lew, "iOS resolution quick reference," http://www.iosres.com/, 2014. 2.2

[21] "Cross platform mobile development tools: Market analysis & forecast - third edition," https://www.reportbuyer.com/product/2266370/cross-platform-mobile-development-tools-market-analysis-and-forecast-third-edition.html, Smith's Point Analytics, 2014. 2.3, 3.2

[22] Y. Ridene and F. Barbier, "A model-driven approach for automating mobile applications testing," in *Proceedings of the 5th European Conference on Software Architecture: Companion Volume*, ser. ECSA '11. New York, NY, USA: ACM, 2011, pp. 9:1–9:7. [Online]. Available: http://doi.acm.org/10.1145/2031759.2031770 2.3, 3.2

[23] A. Charland and B. LeRoux, “Mobile application development: Web vs. native,” *Queue*, vol. 9, no. 4, pp. 20:20–20:28, Apr. 2011. [Online]. Available: http://doi.acm.org/10.1145/1966989.1968203 2.3

[24] K. Haller, “Mobile testing,” *SIGSOFT Softw. Eng. Notes*, vol. 38, no. 6, pp. 1–8, Nov. 2013. [Online]. Available: http://doi.acm.org/10.1145/2532780.2532813 2.4, 4

[25] H. Khalid, E. Shihab, M. Nagappan, and A. Hassan, “What do mobile app users complain about? a study on free ios apps,” *IEEE Software*, vol. 99, no. PrePrints, p. 1, 2014. 2.4

[26] J. Jung, S. Han, and D. Wetherall, “Short paper: Enhancing mobile application permissions with runtime feedback and constraints,” in *Proceedings of the Second ACM Workshop on Security and Privacy in Smartphones and Mobile Devices*, ser. SPSM ’12. New York, NY, USA: ACM, 2012, pp. 45–50. [Online]. Available: http://doi.acm.org/10.1145/2381934.2381944 2.4

[27] “Google slides,” Google, Inc. 2.5, 2.5.1

[28] “Keynote,” https://itunes.apple.com/us/app/keynote/id361285480, Apple Inc., 2014. 2.5, 2.5.1

[29] “Prezi,” https://itunes.apple.com/us/app/prezi/id407759942, Prezi Inc., 2014. 2.5, 2.5.2

[30] “CloudOn,” https://itunes.apple.com/us/app/cloudon-document-editor-ms/id474025452, CloudOn, Inc., 2014. 2.5

[31] “Dropbox,” http://www.dropbox.com/, Dropbox, Inc, 2015. 2.5

[32] "CloudOn," http://www.cloudon.com/, CloudOn, Inc., 2015. 2.5

[33] "Keynote," https://www.apple.com/ios/keynote/, Apple Inc., 2014. 2.5.1

[34] "Microsoft PowerPoint for iPad," https://itunes.apple.com/us/app/microsoft-powerpoint-for-ipad/id586449534, Microsoft Corporation, 2014. 2.5.1

[35] "WPS Office," https://itunes.apple.com/us/app/wps-office-free-+-pdf-compatible/id762263023, Kingsoft Office Software, Inc., 2014. 2.5.1

[36] "Flowvella: Presentation app," https://itunes.apple.com/us/app/flowboard-presentation-app/id630717527, Flowboard LLC, 2014. 2.5.2

[37] "Haiku Deck," https://itunes.apple.com/us/app/haiku-deck-presentation-slideshow/id536328724, Haiku Deck, Inc, 2014. 2.5.2

[38] "SlideShark Presentation App," https://itunes.apple.com/us/app/slideshark-presentation-app/id471369684, Brainshark, Inc., 2013. 2.5.3

[39] "Nearpod," https://itunes.apple.com/us/app/nearpod/id523540409, Panarea, 2014. 2.5.3

[40] "Gotomeeting," Citrix. 2.5.3

[41] "Model-View-Controller," https://developer.apple.com/library/ios/documentation/General/Conceptual/CocoaEncyclopedia/Model-View-Controller/Model-View-Controller.html#//apple_ref/doc/uid/TP40010810-CH14-SW1, Apple, Inc., 2012. 3.2

[42] "Delegates and data sources," https://developer.apple.com/library/ios/documentation/General/Conceptual/CocoaEncyclopedia/DelegatesandDataS

ources/DelegatesandDataSources.html#//apple_ref/doc/uid/TP40010810-CH11, Apple, Inc., 2012. 3.2

[43] "iOS app programming guide," https://developer.apple.com/library/ios/documentation/iPhone/Conceptual/iPhoneOSProgrammingGuide/Introduction/Introduction.html, Apple, Inc., 2013. 3.2

[44] "File system basics," https://developer.apple.com/library/mac/documentation/FileManagement/Conceptual/FileSystemProgrammingGuide/FileSystemOverview/FileSystemOverview.html, Apple, Inc., 2012. 3.2

[45] "SQLite," http://www.sqlite.org/, 2014. 3.2

[46] "The mit license template," http://opensource.org/licenses/MIT, 2015. 3.2

[47] "The bsd 2-clause license template," http://opensource.org/licenses/BSD-2-Clause, 2015. 3.2

[48] "The bsd 3-clause license template," http://opensource.org/licenses/BSD-3-Clause, 2015. 3.2

[49] M. Thompson, "Afnetworking," https://github.com/AFNetworking/AFNetworking, 2015. 3.1

[50] C. T, "Ctassetspickercontroller," https://github.com/chiunam/CTAssetsPickerController, 2014. 3.1

[51] L. Scott, "Draggablecollectionview," https://github.com/lukescott/DraggableCollectionView, 2014. 3.1

[52] O. Drobnik, "Dtcoretext," https://github.com/Cocoanetics/DTCoreText, 2014. 3.1

[53] J. L. D. Silva, “Filehash,” https://github.com/JoeKun/FileMD5Hash, 2014. 3.1

[54] N. Lockwood, “iCarousel,” https://github.com/nicklockwood/iCarousel, 2014. 3.1

[55] M. I. Qurashi, “Iqkeyboardmanager,” https://github.com/hackiftekhar/IQKeyboardManager, 2014. 3.1

[56] J. Nunez, “Jnkeychain,” https://github.com/jeremangnr/JNKeychain, 2014. 3.1

[57] K. Schwaiger, “KSLabel,” https://github.com/vigorouscoding/KSLabel, 2012. 3.1

[58] M. Greenfield, “Mgboxkit,” https://github.com/sobri909/MGBoxKit, 2014. 3.1

[59] M. Rackwitz, “Mrprogress,” https://github.com/mrackwitz/MRProgress, 2014. 3.1

[60] “Mscellaccessory,” https://github.com/bitmapdata/MSCellAccessory, bitmapdata, 2014. 3.1

[61] “Objective-zip,” https://github.com/flyingdolphinstudio/Objective-Zip, Flying Dolphin Studio, 2013. 3.1

[62] A. W. Donoho, “Reachability,” https://github.com/pokeb/asi-http-request/tree/master/External/Reachability, 2012. 3.1

[63] Deadpikle, “ios-rich-text-editor,” https://github.com/Deadpikle/iOS-Rich-Text-Editor, 2015. 3.1

[64] "iOS human interface guidelines," https://developer.apple.com/library/iOS/documentation/userexperience/conceptual/mobilehig/, Apple, Inc., 2014. 3.3.5

[65] "Instruments user guide," https://developer.apple.com/library/mac/documentation/DeveloperTools/Conceptual/InstrumentsUserGuide/InstrumentsUserGuide.pdf, Apple Inc., 2014. 4

[66] "How to calculate iOS app boot time," http://stackoverflow.com/questions/23349570/how-to-calculate-ios-app-boot-time, Stack Overflow, 2014. 5

[67] A. Vollmer, "Tuneup JS," http://www.tuneupjs.org/, 2014. 5