Car Finder App Project Report

for ECE 574, Fall 2014

Srinivas Sambaraju
ECE, University of Michigan-Dearborn
Dearborn, MI, USA
ssambara@umich.edu

***Abstract* — Software Processes play an important role in today's world of ubiquitous software. Any company, team or individual developer can only ignore their importance at their own peril. Software processes originated as a means to bring control over large projects and they have proliferated into many variations depending on the applicability to a situation. A process gives control over a software project and thereby optimizes the time, effort, money, management and ultimately quality. This paper discusses the implementation of an Android App in a short period of time with a single developer by following a software development process discussed in the class ECE 574. The App gives the user the option to search for a car and get specific information while shortlisting a car to purchase. An Agile XP (eXtreme Programming) software process was followed. The paper further explains how the process was followed in each phase of developing this App. The goal of this project to familiarize the author who is a student in the class in using a software process through a learn-by-doing approach.**

Keywords—Software process; Agile; eXtreme Programming; Android; Mobile application.

I. Introduction

This paper discusses the implementation of an Android mobile App for Car Research by strictly following a software process. From the many options available a process that best suits the situation had to be selected and Agile XP (eXtreme Programming) [1] was chosen as it seemed the best fit. XP is just one of the many Agile processes but it is one of the most popular ones. Initially a proposal was made about the concept for the App. After the proposal was approved, the software process was followed in all the phases of the project like requirements, design, coding, testing, and user feedback. Given that Agile is an iterative process, the project evolved over multiple iterations. The developer also had to think from the customer perspective to gather user feedback. The final goal is to detail the software process steps and how the App was developed in multiple iterations in each of those phases.

II. About Agile eXtreme Programming

Agile XP is an iterative development process that is well suited to develop Apps for mobile devices. In this section, the Agile philosophy is described as per software industry sources.

A. Agile definition

In the website "agilemodeling.com" agile software process is defined as follows. "An iterative and incremental (evolutionary) approach to software development which is performed in a highly collaborative manner by self-organizing teams within an effective governance framework with 'just enough' ceremony that produces high quality solutions in a cost effective and timely manner which meets the changing needs of its stakeholders"[4].

B. Selected Principles from the Agile Manifesto [5]

a) Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
b) Welcome changing requirements, even late in development.
c) Deliver working software frequently.
d) Business people and developers must work together daily throughout the project.
e) Build projects around motivated individuals.
f) The most effective method of conveying information within a team is face-to-face conversation.
g) Working software is the primary measure of progress.
h) Agile processes promote sustainable development.
i) Continuous attention to technical excellence and good design enhances agility.
j) Simplicity, the art of maximizing the amount of work not done is essential.

III. Reason to choose the Agile process

This project was about developing an Android App in a short period of time with a single developer. For this, XP is very suitable because it is an iterative process that proceeds in steps where each step could include design, coding, testing and updated requirements. The project can be completely visualized with the GUI itself and obtaining relevant information from the App determined the success. As a developer I could make decisions, run a quick iteration and make corrections.

The process was greatly helpful because development proceeded in many iterations with specific targets for each iteration and within a specific time. Only certain part of the current phase of the project has to be completed to advance to the next phase and the remaining work was covered in the next iteration.

IV. Documentation of the Agile software process

The Agile XP process is explained in detailed in this section with the steps followed in each iteration and the tasks accomplished. The process is documented over four iterations with each iteration involving a subset of the tasks like concept, requirement gathering, requirement review, design, coding, testing and customer review as follows.

A. *Iteration 1*

1) *Concept*
   a) This App helps a buyer of a new or old car to evaluate all the options through a preliminary research by providing information in a concise way.
   b) Car MSRP, True cost to own over five years, engine specs, and ratings from Edmunds are displayed.
   c) The favorite cars can be saved for future reference and also shared with friends via email.
   d) Google maps can be used to locate the dealers in a zip code on a map.

2) *Requirement Gathering*
   a) Requirements were gathered from experience and by talking to friends.
   b) Existing Apps in the market were studied so that new functionality can be provided.
   c) Got an approximate idea for the User Interface and what contents are required in the App.
   d) An API (Application programming interface) from the http://developer.edmunds.com/ website was required to get information about cars. The user asks a query and the App makes a call to the API. Data is obtained in raw format as a JSON file which should be processed and displayed in a user friendly way.

3) *Requirement Review*
   a) The Edmunds API was studied to understand the queries and their content [9].
   b) Registered on Edmunds website to get access to API keys [8].
   c) Once the information that can be obtained from the API was known, the App feature list was updated to complete the work in available time.

4) *Design*
   a) A GUI mock-up was made for the requirements and reviewed for User friendliness from the customer perspective.
   b) A data flow diagram was developed.
   c) Once the Interface design seemed appropriate, the component level design was developed in a Top down manner using pseudo code.
   d) The Agile software development process notes was reviewed from the class.

5) *Coding*
   a) A Top Down coding process was followed with a goal in mind over multiple phases.
   b) The first step was to install Android SDK (software development kit) and Eclipse IDE (Integrated Development Environment) from the website developer.android.com [11].
   c) A sample program was run to understand the IDE and Android framework.
   d) Calling Edmunds API from Android program.
   e) Parsing the raw JSON data from the API and understanding how to present it to the user in a meaningful way [13].
   f) The process was first worked out for one query and Testing was part of this process.
   g) Coding was done in a modular way so that some parts can be re-used later.

6) *Testing*
   a) Testing was done in phases as each part of the App was developed.
   b) Stubs were used in place of final methods for initial testing in the spirit of Top Down design.
   c) Iterative testing was done as code was developed in each phase.
   d) Black box testing of the interface was performed to access if everything is self-explanatory [3].
   e) White box testing of functionality was done. Each button has a specific task, and that was tested [3].
   f) The Android Emulator was used for testing the App at this stage [19] [20].

7) *Customer Review*
   a) In this stage, the App progress was reviewed from developer and customer perspective and future plans were made for refining the UI and the data presentation.

B. *Iteration 2*

1) *Requirement Review*
   a) Reviewed all the required queries from Edmunds API documentation and shortlisted the ones most appropriate for this App.

2) *Design*
   a) Listed Input parameters required by the API to run the queries and parsed the required results from those queries.
   b) For certain results, more than one query had to be run. GUI refactored to make the layout better.

3) *Coding*
   a) After the process was formalized for one query in iteration-1, the remaining queries were also built and parsed in a similar way.
   b) Stepwise refinement was performed to present the results in a readable manner
   c) Code modularity was followed again and some parts were re-used from iteration-1.

4) *Testing*
   a) Once all the GUI buttons and queries were programmed, complete system tests were performed and bugs were fixed.
   b) The App was tested on a Nexus 7 tablet and updates were noted.

C. *Iteration 3*

1) *Design*
   a) Fine tune the GUI style and add buttons for Home, Save, Delete, Email etc.
2) *Coding*
   a) Added code for the Home, Save, Delete and Maps buttons.
   b) Realized that getting Google maps to work is a lot of technical effort.
3) *Testing*
   a) Testing the entire App with latest features and updates.
4) *Maintenance*
   a) Fixing bugs in existing functionality from testing.
   b) Updating and adding new features to improve the utility of the App.
5) *Customer Review*
   a) Reformatting the query results in a more readable way.

D. *Iteration 4*

1) *Design*
   a) Update GUI by removing the map button and add a settings button to store user specific settings like email address [18]. Prepare for final release of the App.
2) *Coding*
   a) Add the settings, and email buttons and make sure all the customer expectations are met before final release.
   b) There was “Coupling” within the major buttons in the home screen where models can only be obtained after make, and year were selected.
   c) The task buttons to display specific information in screen 2 like Ratings, True cost to own are “Cohesive”, i.e., independent of other tasks.
3) *Testing [3]*
   a) Test the complete app for all the functionality as per the updated proposal to meet the customer requirement.
   b) Observability – the results should be relevant to the query.
   c) Controllability – distinct output from each query.
   d) Decomposability – independently test each button.
4) *Release and user Training*
   a) Release the App to the customer and conduct user training sessions.
5) *Maintenance and support*
   a) This should be an ongoing effort over the life cycle of the App.

V. Implementation

In this section, the details of the coding are explained.

A. *Programming process*

Initially, the android SDK was downloaded along with the Eclipse IDE from the developer.android.com website [11]. Some sample programs were run to understand the SDK and the IDE. An API level of 18 was used. As per the requirement, the initial code was developed and tested in iteration-1. After that iteration-2 followed. For specific requirement, code snippets from developer.android.com were useful along with the stackoverflow.com website. The Edmunds API was studied and a simple query to fetch the car models was programmed first. The data was in JSON format and had to be processed. Then the code to process this JSON was developed. Then taking these results and displaying them on the screen was done. Once this cycle was completed for one query, the same process was repeated for the remaining queries. Part of the re-usable code like JSON parsing was converted into functions and be re-used for the other queries. Next the Home, Save and Delete buttons were implemented in iteration-3. After iteration-4 all the programming was completed as per the proposal and updated customer requirements and the App was released for customer use.

Initially the App was tested on the Android emulator. Android API level 18 was chosen with Jelly bean as the android device was available with that configuration for final deployment. At the end of iteration-2, the app was deployed on the device and the user friendliness and the functionality was tested. Improvements were made in iteration-3.

B. *Programming details*

The programming details of the Android project are explained in this section. The main program is an Activity class where the GUI is created [13]. An Async task is a background process where queries to external API's can be made so that the main UI thread is not blocked and the App continues to function while the query is being processed. The folders, files, classes and methods used in this Android project are briefly explained as follows.

1) *src* folder has all source code.
2) *res, layout* folder has an xml file where GUI elements like buttons, number pickers etc., are declared.
3) *res, drawable* folder has all the graphical icons
4) The activities and App permissions are defined in the manifest xml file.
5) *CarResearchActivity* class is the main program
6) *DisplayResultsActivity* class displays the results
7) *onCreate()* method creates all GUI elements.
8) *getModelsEvent()* method is called when the Get Models button is clicked.
9) *GetModelsUrlTask()* is a background Async task to get the models from Edmunds API.
10) *readJsonStream()* method reads the returned JSON data.
11) *ParseModelListJSON()* method parses the JSON to get the models.
12) *getEngTransEvent()* method calls the *GetStatsUrlTask()* background Async task to get Engine transmission data.
13) *getVehConfigEvent()* method calls the *GetStatsUrlTask()* task to get the vehicle configuration data.

14) *getTrueCostToOwnEvent()* method calls the *GetStatsUrlTask()* task to get true cost to own data.

15) *getEdmundsRatingsEvent()* method calls the *GetStatsUrlTask()* task to get Edmunds ratings data.

16) *GetStatsUrlTask()* is a background Async task that is the workhorse for this application. It has the below methods to parse the JSON and display the results [12] [13].

   a) parseStyleIDJSON()
   b) parseEngTransJSON()
   c) parseVehicleConfigJSON()
   d) parseTrueCostToOwnJSON()
   e) parseCarRatingsJSON()

C. Data Flow Diagram [2]

The data flow diagram is a graphical representation of information flow in an application. It is an important part of Analysis modeling and illustrates data transformation in the application by the functions involved [3]. In this project a data flow diagram was developed in the design phase and refined over four iterations. It gives a quick overview of the entire application to the developer, management and customer and forms an integral part of effective communication between the stake holders. It also serves as documentation for future developers who work on any updates to the application.

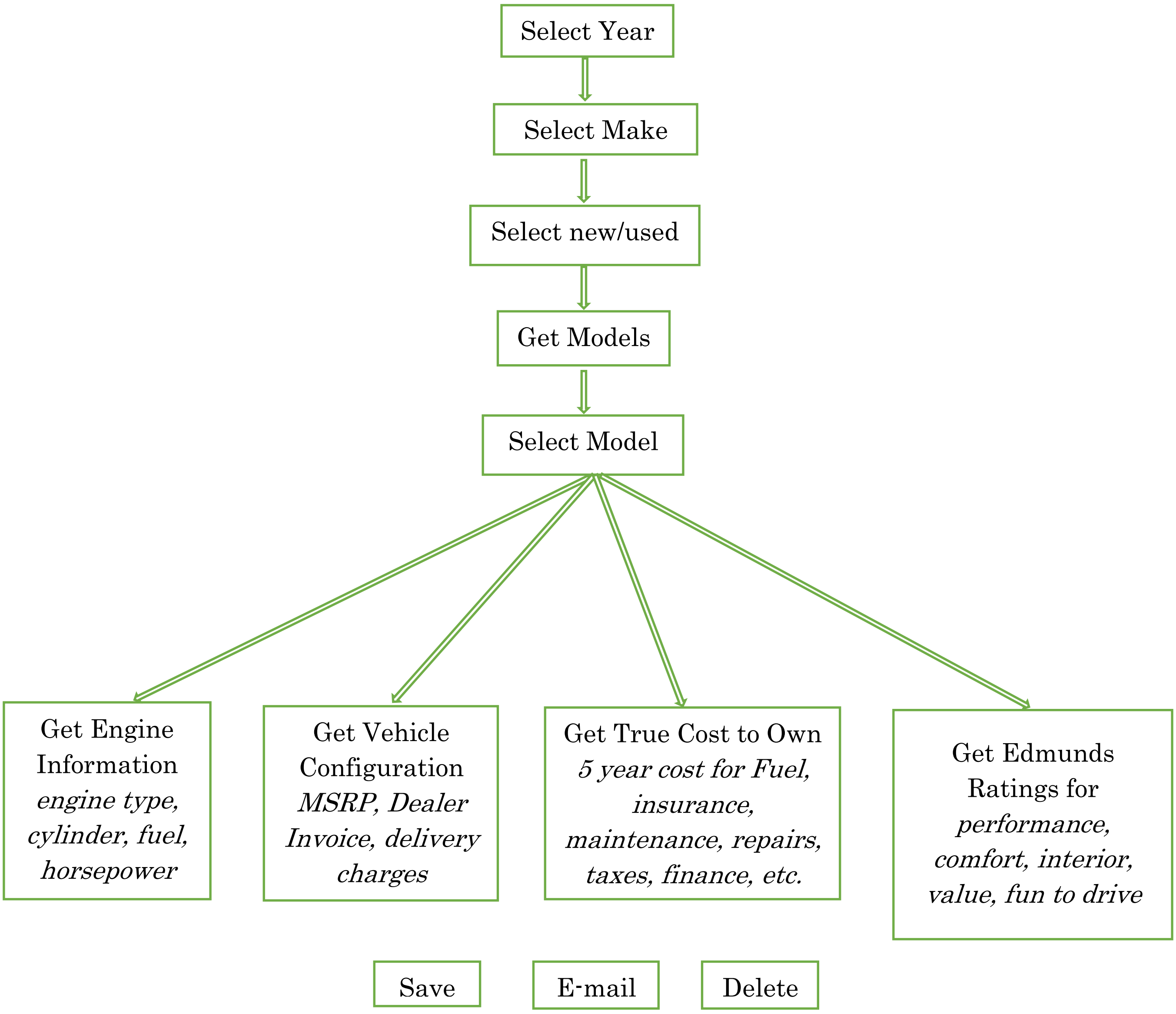

Select Year
Select Make
Select new/used
Get Models
Select Model
Get Engine Information *engine type, cylinder, fuel, horsepower*
Get Vehicle Configuration *MSRP, Dealer Invoice, delivery charges*
Get True Cost to Own *5 year cost for Fuel, insurance, maintenance, repairs, taxes, finance, etc.*
Get Edmunds Ratings for *performance, comfort, interior, value, fun to drive*
Save
E-mail
Delete

Fig. 1. Data Flow Diagram

D. *Pseudo Code*

```
MainAndroidActivity() {
        Setup GUI ()
        Get user selection ()
        Call Edmunds API using Http library ()
        Parse the returned JSON data ()
        Display on the screen ()
        Save, Delete, and Email the data ()
}
```

VI. APP SCREEN SHOTS

These are some of the screen shots of the App.

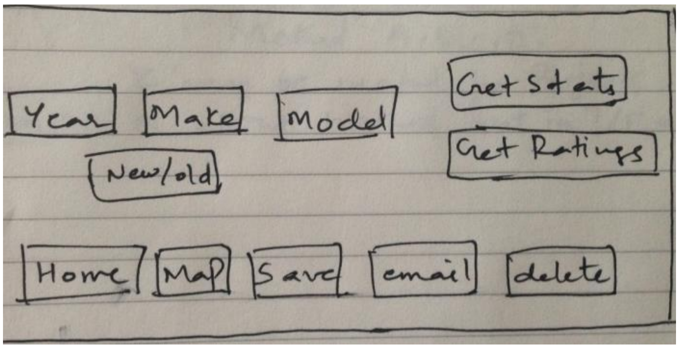


Fig. 2. Initial mockup of the GUI in concept phase

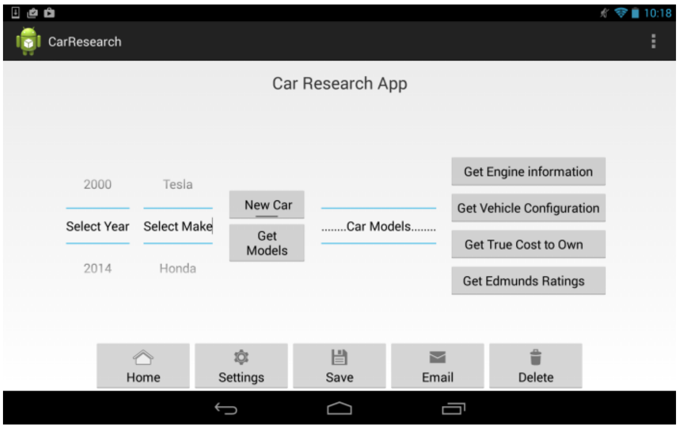


Fig. 3. Home screen

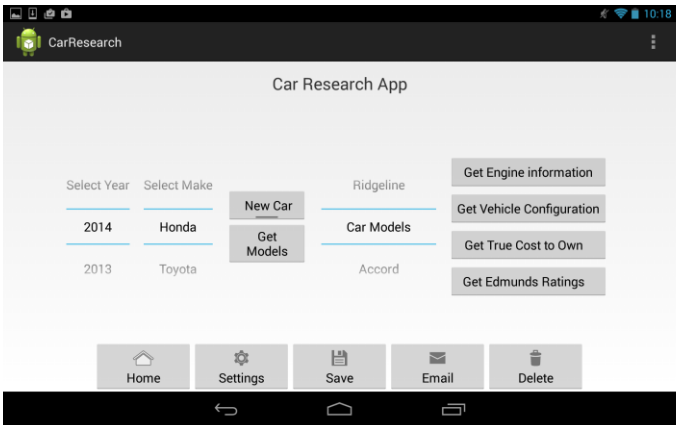
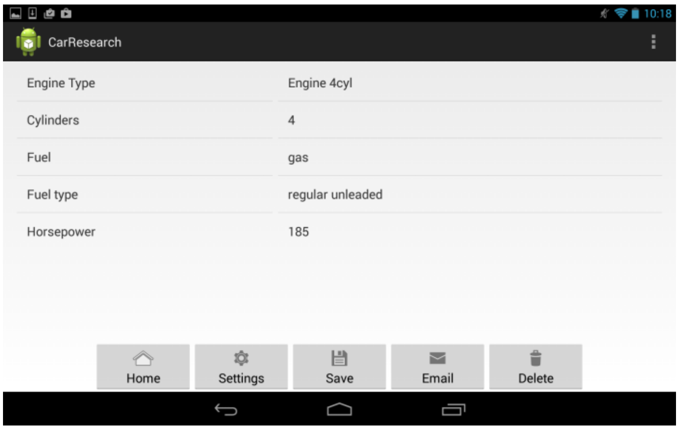


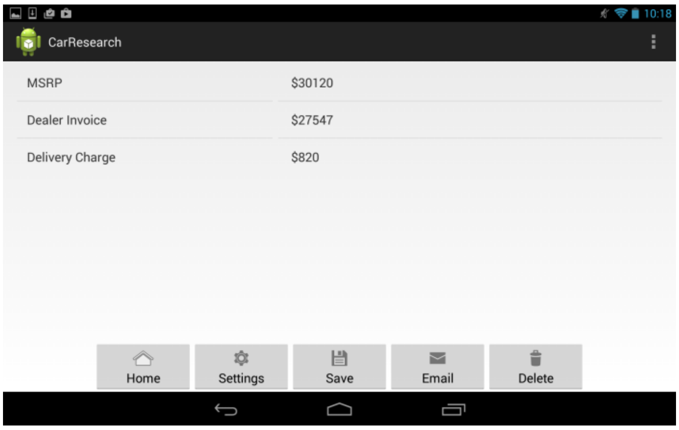
Fig. 4. Ready to run the queries after selection

The following are screenshots of the results displayed in Asus Nexus 7 tablet when the query buttons are selected in the main screen. The screens in order are Engine information, Vehicle configuration, True cost to own, and Edmunds Ratings. These values can be saved to a file and emailed. The file can also be deleted.



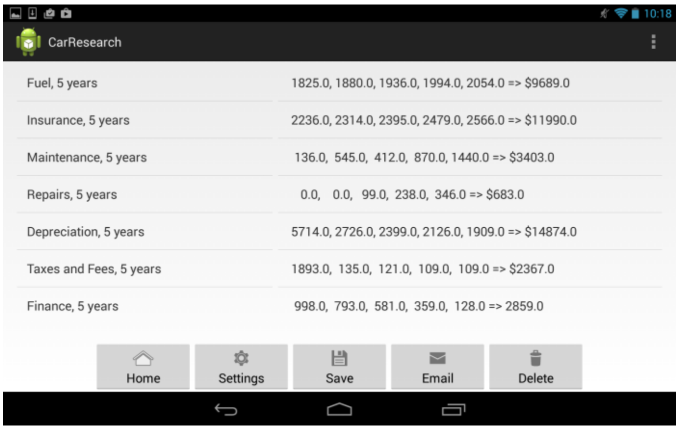
Engine Type	Engine 4cyl
Cylinders	4
Fuel	gas
Fuel type	regular unleaded
Horsepower	185

Fig. 5. Engine information



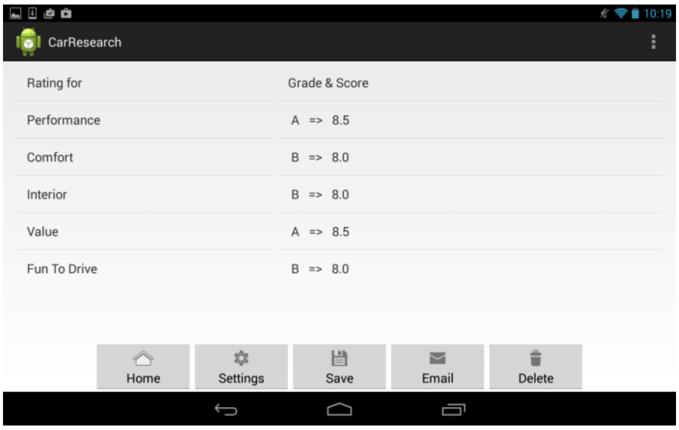
MSRP	$30120
Dealer Invoice	$27547
Delivery Charge	$820

Fig. 6. Vehicle Configuration



Fuel, 5 years	1825.0, 1880.0, 1936.0, 1994.0, 2054.0 => $9689.0
Insurance, 5 years	2236.0, 2314.0, 2395.0, 2479.0, 2566.0 => $11990.0
Maintenance, 5 years	136.0, 545.0, 412.0, 870.0, 1440.0 => $3403.0
Repairs, 5 years	0.0, 0.0, 99.0, 238.0, 346.0 => $683.0
Depreciation, 5 years	5714.0, 2726.0, 2399.0, 2126.0, 1909.0 => $14874.0
Taxes and Fees, 5 years	1893.0, 135.0, 121.0, 109.0, 109.0 => $2367.0
Finance, 5 years	998.0, 793.0, 581.0, 359.0, 128.0 => 2859.0

Fig. 7. True Cost to Own



Rating for	Grade & Score
Performance	A => 8.5
Comfort	B => 8.0
Interior	B => 8.0
Value	A => 8.5
Fun To Drive	B => 8.0

Fig. 8. Edmunds Car Ratings

VII. Edmunds Api usage

This section describes the Edmunds API which forms the foundation for getting the data on the vehicles. The process is illustrated in a few steps as follows [8].

1) Register on http://developer.edmunds.com/ to get API key.
2) http://edmunds.mashery.com/io-docs webpage can be used to run queries directly on the webpage [9].
3) A Sample query is illustrated below. https://api.edmunds.com/api/vehicle/v2/mazda?state=new&year=2013&view=basic&fmt=json&api_key=zj3x5ustr7we2tz2xcwh5ty9
4) This query is built from the user input as follows. String urlString = https://api.edmunds.com/api/vehicle/v2/ + selMake + "?state=" + selNewOld + "&year="+ selYear + "&view=basic&fmt=json&api_key=" + apiKey;
5) To run in the Android App, a *HttpGet* request with the urlString is sent to a *DefaultHttpClient* and *HttpResponse* is obtained as an input stream in JSON format. This JSON data is parsed using *org.json* library. The formatted results are displayed on the screen.

VIII. Conclusion and Future work

The professor recommended during the presentation that an App for searching the least expensive cars in a category will be greatly useful. Because the data availability was restricted to the queries allowed by the Edmunds API, it was not possible in this project to accomplish this. Either talking to Edmunds directly about this functionality or exploring more API's available in the market is required.

In the future this App can be commercialized by adding some more useful features need. The queries allowed by the Edmunds API can be leveraged to the maximum extent in this regard.

The Agile XP process provided a great foundation for future App development and it has also helped in my daily software programming work. Refining the process followed in this project to make it more repeatable is an idea to pursue.

Acknowledgment

The project gave me an opportunity to think of an idea for a mobile app and develop it according to a software process. The author is thankful to Dr.Adnan Shaout for providing this opportunity to learn the software process using a practical example.

References

[1] ECE 574 class slides from Software_Processes_chapter_9.1.ppt
[2] ECE 574 class slides from Analysis_Modeling-Chapter_3.ppt
[3] ECE 574 slides from Software_Testing_Techniques_Chapter_7.1.ppt
[4] http://www.agilemodeling.com/essays/agileSoftwareDevelopment.htm
[5] http://agilemanifesto.org/principles.html
[6] http://blog.sei.cmu.edu/post.cfm/what-is-agile
[7] www.slideshare.net/alanmcsweeney/implementing-agile-iterative-project-delivery-approach-and-achieving-business-responsiveness-4984011
[8] http://developer.edmunds.com/
[9] http://edmunds.mashery.com/io-docs
[10] http://www.javacodegeeks.com/2010/10/android-full-app-part-2-using-http-api.html
[11] http://developer.android.com
[12] http://developer.android.com/reference/android/util/JsonReader.html
[13] http://www.technotalkative.com/android-json-parsing/
[14] http://developer.android.com/reference/android/widget/TableRow.html
[15] http://www.mkyong.com/android/android-tablelayout-example/
[16] http://developer.android.com/training/basics/data-storage/files.html
[17] https://developer.android.com/guide/components/intents-common.html#Email
[18] http://stackoverflow.com/questions/587917/trying-to-attach-a-file-from-sd-card-to-email
[19] http://johnkalberer.com/2013/04/17/testing-android-apps-on-your-nexus-7/
[20] http://developer.android.com/tools/extras/oem-usb.html