Trust Based App Marketing: Design, Implementation and Evaluation

by

Keone D. Hon

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2011

Author . . . .
Department of Electrical Engineering and Computer Science
May 20, 2011

Certified by . . . .
Glen L. Urban
David Austin Professor of Marketing
Chairman, MIT Center for Digital Business
Thesis Supervisor

Accepted by . . . .
Dr. Christopher J. Terman
Chairman, Department Committee on Graduate Theses

Trust Based App Marketing: Design, Implementation and Evaluation

by

Keone D. Hon

Submitted to the Department of Electrical Engineering and Computer Science
on May 20, 2011, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

A **trust-based marketing application** is a web or mobile app which provides a utility to the consumer that is not directly linked to purchasing products or services from the company. In this thesis, I explore the efficacy of trust-based marketing apps by designing and building two apps–one iPhone app and one Adobe Flash app–and test them under market research conditions. These apps attack two important and complementary questions: how trust-based apps compare to traditional stimuli such as television ads; and how different dimensions of app design (degrees of social network embeddedness, customizability, and information discovery) affect the efficacy of a trust-based app. I build a Flash-based design environment capable of basic image manipulations, I design a web engine that organizes tens of thousands of images used in the design environment, and I build modular systems for parsing, logging, and data extraction—essential features for testing how consumers interact with these apps. Analysis of the iPhone app study demonstrates excellent results for trust-based marketing apps, with a favorable comparison against television ads that included a 60% greater increase in consideration and a 34% greater increase in preference for the sponsoring company (compared to television ads). Analysis of the Flash app is forthcoming as the study is in market research at the time of writing.

Thesis Supervisor: Glen L. Urban
Title: David Austin Professor of Marketing
Chairman, MIT Center for Digital Business

Acknowledgments

To begin, I would like to thank my advisor, Professor Glen Urban, for all of his guidance, support, and friendship over the past year. Glen is not only one of the most innovative people that I have met in my life, but he is also one of the kindest. I am constantly amazed by his thoughtfulness, his patience, and his deep insights into whatever problem is at hand. I've learned many things from Glen, who introduced me to the exciting world of digital marketing, and I am greatly indebted to him.

I want to thank Professor Renee Gosline, Patricia Hawkins, and Joyce Salisbury for managing and supporting me on the GM project, as well as for their advice and humor during our many conference calls. I also owe many thanks to Nell Putnam-Farr, Courtney Quinn, Melih Ecertas, Ovul Sezer, Gui Liberali, Douglas Hwang, and Oliver Baecker for their excellent teamwork on the LM and GM projects.

I would like to thank my fellow researchers Emily Zhao and Cordelia Link, who have been great partners to work with over the past year—and I wish all the best to Emily, who will sit in my seat next year. I would also like to thank Jong-Moon ("Jiggity") Kim and Ted Tomlinson for introducing me to Glen and for laying excellent groundwork on both the LM and GM projects during their time as RAs.

This research is conducted at the Center for Digital Business and the Marketing Department at the MIT Sloan School of Management with the financial support of General Motors and Liberty Mutual. I would like to thank our sponsors for making my work financially possible, as well as for being a pleasure with which to work.

Finally, I would like to thank my parents who have supported me in every way possible, up to and including letting me leave the nest early—when I left home in tenth grade for Exeter—because they felt it was best for me. The older I get the more I realize how much they have shaped me, and I am eternally grateful for all of their sacrifices and lessons. I would also like to thank my little brother Kimo, who always puts a smile on my face.

And to all the people I missed out from this list: you know who you are, and I am greatly indebted to you... Thank you!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

An emerging trend in marketing has been the use of web and mobile applications to market to a new generation of consumers. These forays into new media reflect a response to increased use of the internet, a proliferation of devices for skipping commercials on television, and the perception that todays consumers have been trained to tune out traditional marketing stimuli.

Of particular note is the use of the **trust-based marketing app**, a webapp or mobile app which provides a utility to the consumer that is not directly linked to purchasing products or services from the company. Instead of serving as another channel for consumption, a trust-based app attempts to build a relationship with the consumer to pave the way for future sales.

In spite of the interest, little research has been done to assess the efficacy of new media as compared to old media, or to ascertain which elements of app-based marketing are essential ingredients for a winner. For instance, does an app need to have a social component in order to be successful? Can an app advertise products sold by the sponsoring company? Should the apps developers spend more time on developing the social components of the app, or on improving the individuals experience?

In this thesis I explore the efficacy of trust-based apps by designing and building two apps—one mobile and one web application—which help in addressing these questions. This year, Liberty Mutual and General Motors signed on with us to conduct significant studies into trust-based app marketing, and the apps I built provide insights into how these companies can use this marketing strategy to their advantage. The first, **DubbleWrap**, is an iPhone app branded with the Liberty Mutual logo which assists users in moving from one home to another. The second, **DreamCar**, is a Flash-based web game branded with the Chevrolet logo in which users design their own dream car. In this thesis I describe the design and implementation of each app, highlighting key engineering decisions and identifying challenges. I also describe the event logging systems I built to understand how consumers testing the apps in market research panels were using the apps. Lastly, I analyze both systems from an engineering perspective, and review the results of the DubbleWrap study; the DreamCar study is currently being tested in market research panels at the time of writing.

1.1 Summary of Thesis Content

Chapter 2 will provide background information about the DubbleWrap study and the research methods used for assessing the efficacy of the app.

Chapter 3 will give an overview of the design of DubbleWrap and will review the major design decisions.

Chapter 4 will provide results of the market research analysis of DubbleWrap.

Chapter 5 will discuss the significance of the research analysis, and will provide a technical evaluation of the app.

Chapter 6 will provide background information about the DreamCar study and the research methods used for assessing the efficacy of the app.

Chapter 7 will give an overview of the design of DreamCar and will review the major design decisions.

Chapter 8 will review technical details of the architecture and implementation of DreamCar.

Chapter 9 will provide a technical evaluation of the app.

Chapter 10 will discuss personal contributions and future work.

Chapter 2

DubbleWrap: Background

2.1 Introduction

Liberty Mutual Group is a diversified global insurer with a major U.S. and global presence. In the United States, it is the fifth largest property and casualty insurer (based on 2009 direct written premium). Liberty Mutual offers a number of insurance products ranging from personal automobile and homeowners insurance to general liability insurance.

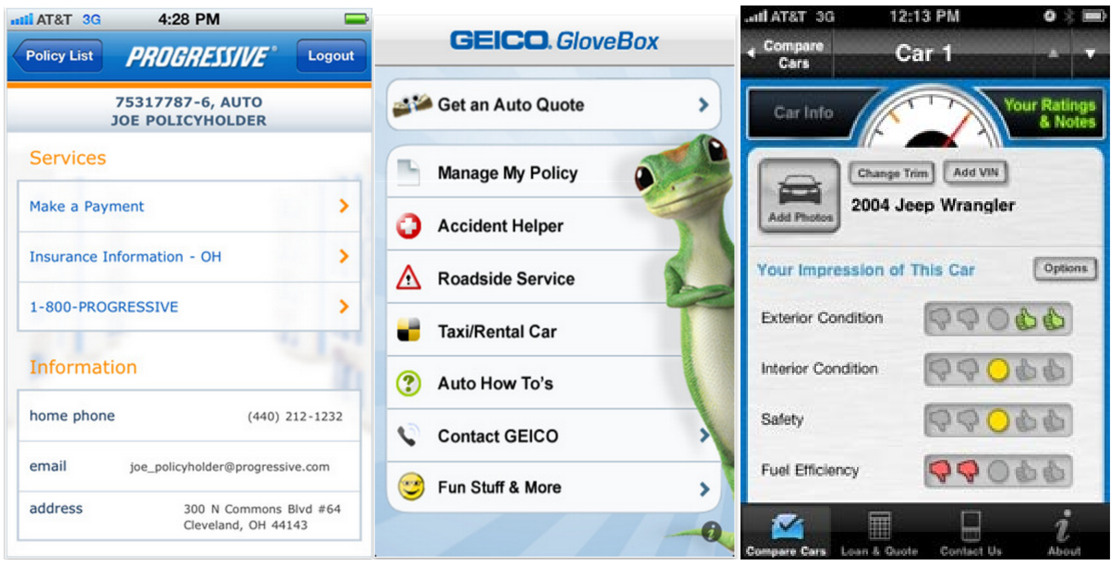
Liberty Mutual has an extensive history of valuing trust-based marketing. As an example, consider the company's 2006 television commercial about people doing things for strangers, which led to the development of "The Responsibility Project," a website with blogs and features about "what it means to do the right thing." On "The Responsibility Project," contributors and users wax philosophic about topics ranging from whether cursive should still be taught in elementary schools to whether regifting an unwanted present is ethical.

Trust-based app marketing thus represents a logical step for the firm. Additionally, while trust-based apps are novel to the industry, the concept of using a mobile app for general insurance marketing has some precedent. In 2009 alone, three of Liberty Mutual's main competitors launched mobile apps, shown in Fig. 2-1.

- Progressive's 2009 iPhone app allowed policyholders to report claims and compare insurance costs for different vehicles.[1]

- GEICO also launched an app in 2009, “GEICO GloveBox,” which allowed customers to pay their insurance bill, view their insurance card, and get step-by-step instructions on what to do in the event of an auto accident. (The app also features allows users to watch videos of the GEICO Gecko, the firm’s popular mascot.)[2]

- Nationwide’s Cartopia iPhone app allowed users to comparise insurance costs for different vehicles, and also provided free AutoCheck vehicle history scores from Experian Automotive—a particularly useful feature because normally consumers must pay for these reports.



75317787-6, AUTO
JOE POLICYHOLDER

home phone (440) 212-1232
email joe_policyholder@progressive.com
address 300 N Commons Blvd #64 Cleveland, OH 44143

2004 Jeep Wrangler

(a) Progressive Mobile (b) GEICO Glovebox (c) Cartopia by Nationwide

Figure 2-1: Competitive landscape for insurance-sponsored mobile apps

It was time to test an app for Liberty Mutual. Our goals for this study: to understand how new media compares to old media, and specifically, to test the efficacy of a trust-building mobile app. The key questions: Does it build trust? Does it help consideration? How does it compare to traditional media exposure?

[1]http://www.progressive.com/resources/progressive-app-iphone.aspx
[2]http://www.geico.com/about/pressreleases/2009/20091209/

2.2 App Features

A trust-based app provides a utility to the consumer that is not directly linked to purchasing services from the company. The first step was to determine what utilities our app should provide to customers. In consultation with Liberty Mutual, we built an app concept called "BubbleWrap" which was a set of tools to assist someone who is moving. The concept's features were tested in a focus group; as a result of that work, we narrowed the focus to eliminate several components (namely a damage-reporting feature for recording damages which would have allowed users to file claims from their mobile device, and an insurance-estimation feature for estimating the monthly cost of an insurance policy). Feedback from the focus group also caused us to alter the name to "DubbleWrap," which is more distinctive.

The app would be branded with the Liberty Mutual logo, but would not attempt to sell insurance to the user. The app's title screen is shown in Fig. 2-2.

Figure 2-2: DubbleWrap title screen

2.3 Methods

Experiment description The primary experiment is a clinical test of the DubbleWrap mobile app versus TV exposure. The experiment uses forced exposure, i.e. test subjects are market research panel members from a leading market research company, and their responses are only considered if they complete the stimulus and respond to survey questions, as is standard market research practice.

Primary control The primary control is exposure to two thirty-second TV ads, "Responsibility (Anthem II)"[3] and "Garage Door"[4], screenshots of which are shown in Fig. 2-3.

Figure 2-3: DubbleWrap study TV control exposures

Secondary control Additionally, we compared the DubbleWrap app to a website, `www.youcovered.com`, which is a website built by Liberty Mutual to inform consumers about the costs and benefits of renters' insurance. Screenshots of the website are shown in Fig. 2-4. Users given website exposure served as a secondary control.

[3]`http://www.youtube.com/watch?v=E9UICosC5aU`
[4]`http://www.youtube.com/watch?v=NFV88IgU_C8`

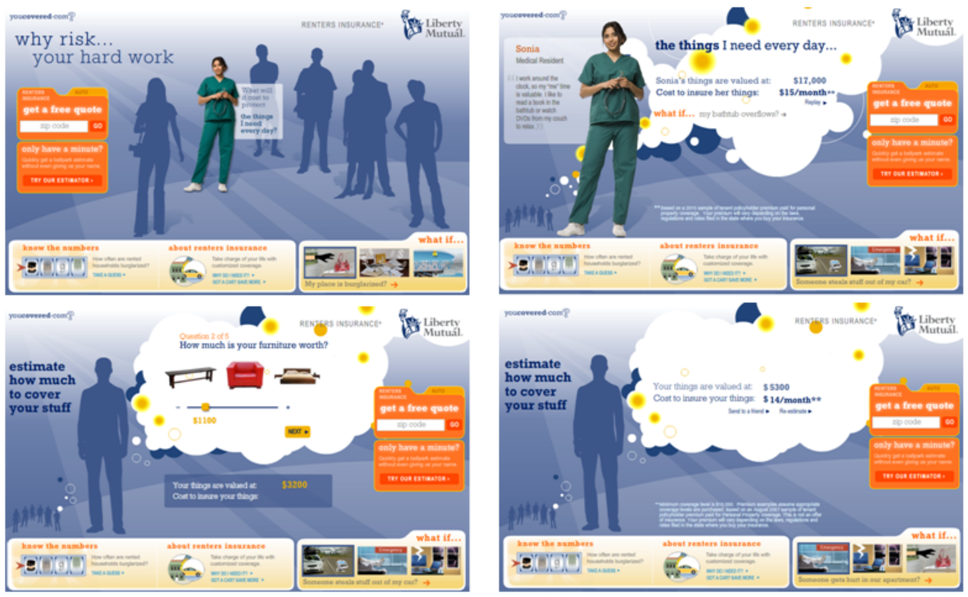


Figure 2-4: DubbleWrap study website control exposure: `www.youcovered.com`

Pre/post measures We measured attitudes toward the Liberty Mutual brand and toward risk by having respondents complete a survey before and after exposure. Pre/post differences were computed and used as a primary basis for comparing treatment groups.

Survey Selections from the survey used in this study are available in Appendix A.

2.3.1 Measurements

Familiarity Assess familiarity with LM and its primary competitors (Allstate, Farmers, GEICO, Nationwide, Progressive, State Farm) (scale 1-4)

Consideration Assess likelihood of considering each insurance company for homeowners/renters insurance (scale 1-10)

LM attributes Assess agreement (scale 1-5) with twelve statements about Liberty Mutual.

1. Overall, I trust Liberty Mutual.

2. Liberty Mutual offers insurance coverage at a better price.
3. Liberty Mutual offers more flexible insurance policies.
4. Liberty Mutual offers better customer service in case of an insurance claim.
5. Liberty Mutual is open, honest, and transparent.
6. Liberty Mutual works hard to meet my changing needs.
7. Liberty Mutual is willing to assist and support me.
8. I would recommend Liberty Mutual to a friend.
9. Liberty Mutual appears more responsible than other insurance companies
10. Liberty Mutual is an insurance company that will deliver on promises made
11. Liberty Mutual provides believable information
12. I have confidence in the information Liberty Mutual provides

Risk attitudes Assess agreement (scale 1-5) with eight statements about attitudes toward protection and risk.

1. Avoid insurance (I try to deal with insurance-related topics as rarely as possible)
2. Plan ahead (I plan most of my activities ahead of time to avoid unexpected events)
3. Safe sports (I engage only in safe sports where I have control)
4. Responsible activities (I follow activities which my friends and colleagues would rate as responsible)
5. Risk-secured moving (When moving it is important to me to be secured against all kind of risks)
6. Afraid of moving damage (I am afraid my belongings might get lost or damaged when moving)
7. Full coverage impt. (It is important to me that all of my belongings are covered by my homeowners / renters insurance)
8. Most valuable coverage impt (It is important to me that my most valuable belongings are covered by my homeowners / renters insurance)

Likelihood of purchase Assess likelihood (scale 1-5) of:

1. Getting a quote for homeowners/renters insurance from LM the next time it is needed
2. Getting a quote for auto insurance from LM the next time it is needed
3. Getting information about additional insurance products from LM the next time they are needed

Points allocation Assess preferences over the insurance space. Survey-takers allocate 100 points across the following companies, with more points indicating greater preference the next time homeowners/renters insurance is needed.

1. Allstate
2. Farmers
3. GEICO
4. Liberty Mutual
5. Nationwide
6. Progressive
7. State Farm
8. Other

2.4 Research Subjects

Test subjects were assigned to one of three groups, as shown in Fig. 2-5:

- **Test group AB**: iPhone users; given the DubbleWrap stimulus. $n = 550$.
- **Control group AC**: iPhone users; given the TV ad stimulus. $n = 100$.
- **Control group AD**: iPhone users; given the website stimulus. $n = 100$.

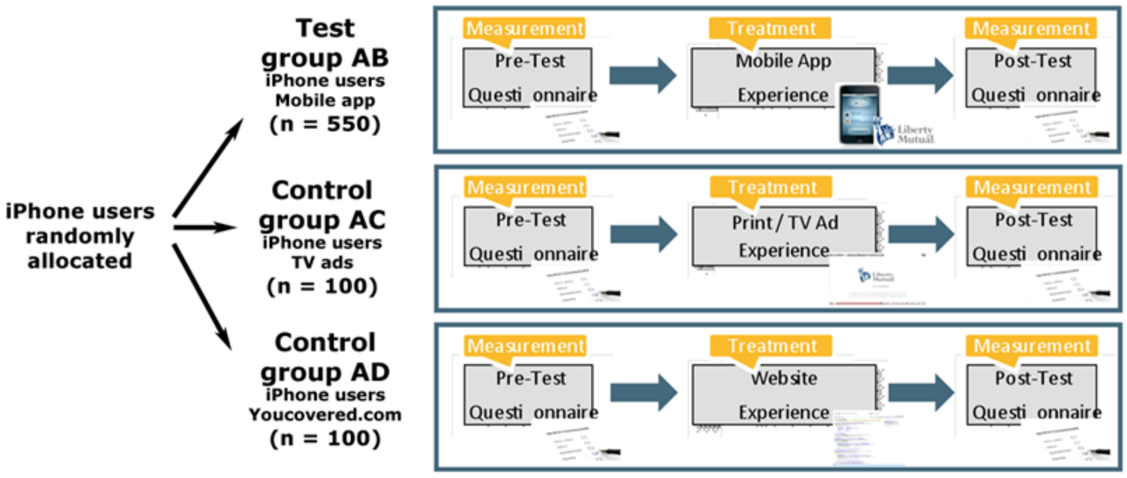


Figure 2-5: Experimental design: test and control groups

Test subjects are panel members from a leading market research company. Because the app provides a service to consumers who are in the process of moving, subjects were screened for having moved in the past six months or for contemplating a move in the next three months. Additionally, participants needed to own an iPhone and needed to be willing to download and test an iPhone app.

To reduce selection bias, we required that all subjects in all experimental groups satisfy these criteria, although subjects who did not own an iPhone were still obviously physically capable of completing the tasks in the control groups. As an alternative control, we measured the brand effects on these consumers, who were assigned to groups **C** and **D**. A comparison of the test group with these alternative controls is given in the appendix.

- **Control group C**: non-iPhone users; given the TV ad stimulus. $n = 100$.
- **Control group D**: non-iPhone uesrs; given the website stimulus. $n = 100$.

The waterfall diagram in Fig. 2-6 depics the screening process.

2.5 App Testing

Because the app is designed for use while actually moving, and most of the survey respondents were not actually moving while taking the survey, we needed to simulate

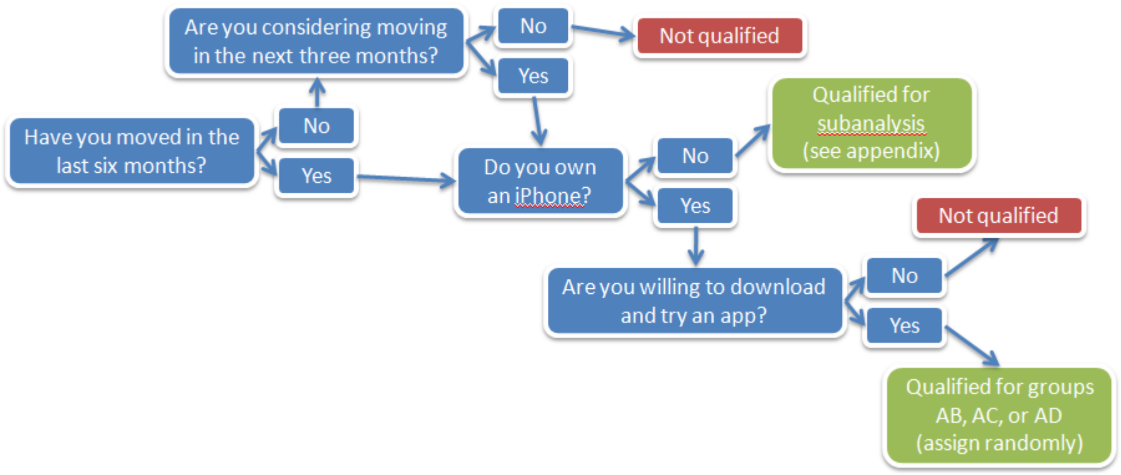
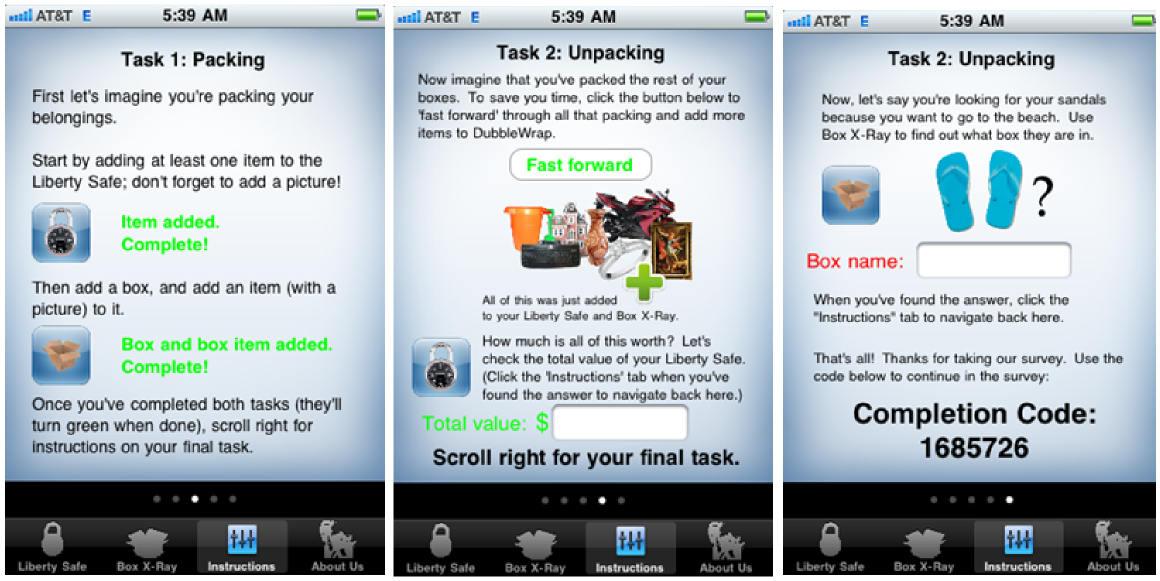

Are you considering moving in the next three months?
No
Not qualified
Yes
Have you moved in the last six months?
No
Yes
Do you own an iPhone?
No
Qualified for subanalysis (see appendix)
Yes
Not qualified
Are you willing to download and try an app?
No
Yes
Qualified for groups AB, AC, or AD (assign randomly)

Figure 2-6: Waterfall diagram depicting user flow through the screening process

a moving scenario to demonstrate the app's true use case. We designed two tasks for users to complete, as shown in 2-7. In the first task, the user must use DubbleWrap to add to an inventory of possessions. In the second task, the app "fast-forwards" through the moving process and adds a number of items to the user's inventory; then the user must use the tools provided in the app to determine the total value of his valuables (e.g. for determining how much renter's insurance to purchase) and to find a desired item packed away in a box.



Task 1: Packing

First let's imagine you're packing your belongings.

Start by adding at least one item to the Liberty Safe; don't forget to add a picture!

Item added. Complete!

Then add a box, and add an item (with a picture) to it.

Box and box item added. Complete!

Once you've completed both tasks (they'll turn green when done), scroll right for instructions on your final task.

Task 2: Unpacking

Now imagine that you've packed the rest of your boxes. To save you time, click the button below to 'fast forward' through all that packing and add more items to DubbleWrap.

Fast forward

All of this was just added to your Liberty Safe and Box X-Ray.

How much is all of this worth? Let's check the total value of your Liberty Safe. (Click the 'Instructions' tab when you've found the answer to navigate back here.)

Total value: $

Scroll right for your final task.

Task 2: Unpacking

Now, let's say you're looking for your sandals because you want to go to the beach. Use Box X-Ray to find out what box they are in.

?

Box name:

When you've found the answer, click the "Instructions" tab to navigate back here.

That's all! Thanks for taking our survey. Use the code below to continue in the survey:

Completion Code: 1685726

Figure 2-7: Survey takers are given tasks to simulate using the app during a moving scenario

Both tasks were embedded directly within the app; users were told to navigate to a specific page which gave them instructions on how to proceed on the task. This design choice was made to allow users to test the app more naturally instead of having to refer to on-screen instructions. When the user completed the task, he was given a completion code to enter into the online survey, which would allow him to complete the post-stimulus survey.

Chapter 3

DubbleWrap: Design

3.1 Features

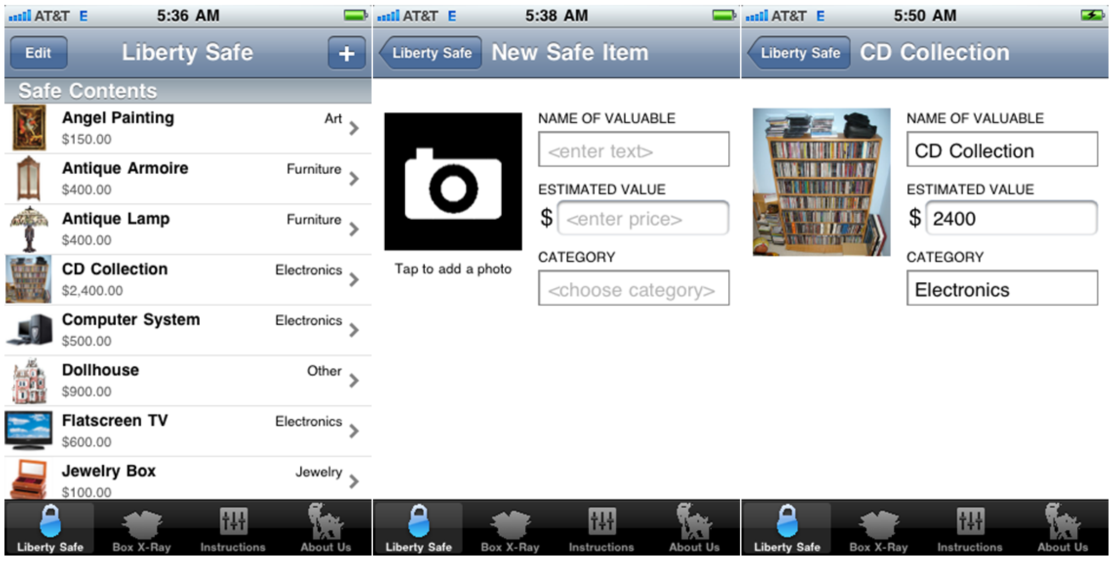
DubbleWrap has two main features: the **Liberty Safe** and the **Box X-Ray**. The user navigates between features using the tabs at the bottom of the screen, a standard iPhone user interface component.

The **Liberty Safe** is an inventory for valuables. Users can add a price, category, and picture to each item in the Liberty Safe, as shown in Figure 3-1.

The photo feature can shoot a new photo using the iPhone's native camera, or can select an existing photo from the Camera Roll, as shown in Fig. 3-2.

At the bottom of the Liberty Safe item list, the app computes some metainformation such as the total value of the items in the safe. This could be useful, for example, in determining how much renters insurance to purchase.

The **Box X-Ray** is a tool for recording the contents of boxes. It is designed to help address the problem of forgetting the contents of a box, or of needing to find something but not remembering which box it was in. Boxes have names, rooms, and categories; boxes also have items, which are similar to Liberty Safe items, inside them. Users can add photos to box items. Several views of the Box X-Ray are shown in Fig. 3-3.



Safe Contents

Angel Painting	$150.00	Art
Antique Armoire	$400.00	Furniture
Antique Lamp	$400.00	Furniture
CD Collection	$2,400.00	Electronics
Computer System	$500.00	Electronics
Dollhouse	$900.00	Other
Flatscreen TV	$600.00	Electronics
Jewelry Box	$100.00	Jewelry

NAME OF VALUABLE <enter text>
ESTIMATED VALUE $ <enter price>
CATEGORY <choose category>

NAME OF VALUABLE CD Collection
ESTIMATED VALUE $ 2400
CATEGORY Electronics

(a) Viewing items in the Liberty Safe (b) Adding a new item (c) Viewing an item

Figure 3-1: Several views of the Liberty Safe feature in DubbleWrap

3.2 Branding

DubbleWrap is branded with the Liberty Mutual logo in several places. It appears on the title screen (shown when the app is loading) and on the app's first page, shown in Fig. 3-4.

3.3 Instructions Tab and Tasks

As mentioned in the previous chapter, we embedded instructions for survey-takers directly into the app. In order to make the app fully functional outside of the scope of a survey, we placed the instructions inside the instructions tab and made them only appear if the user entered a valid survey code on the second page of the instructions tab, as shown in Fig. 3-5.

Entering a valid code turned on a number of features related to survey activity beyond revealing the task instructions. It turned on notifications to inform the user

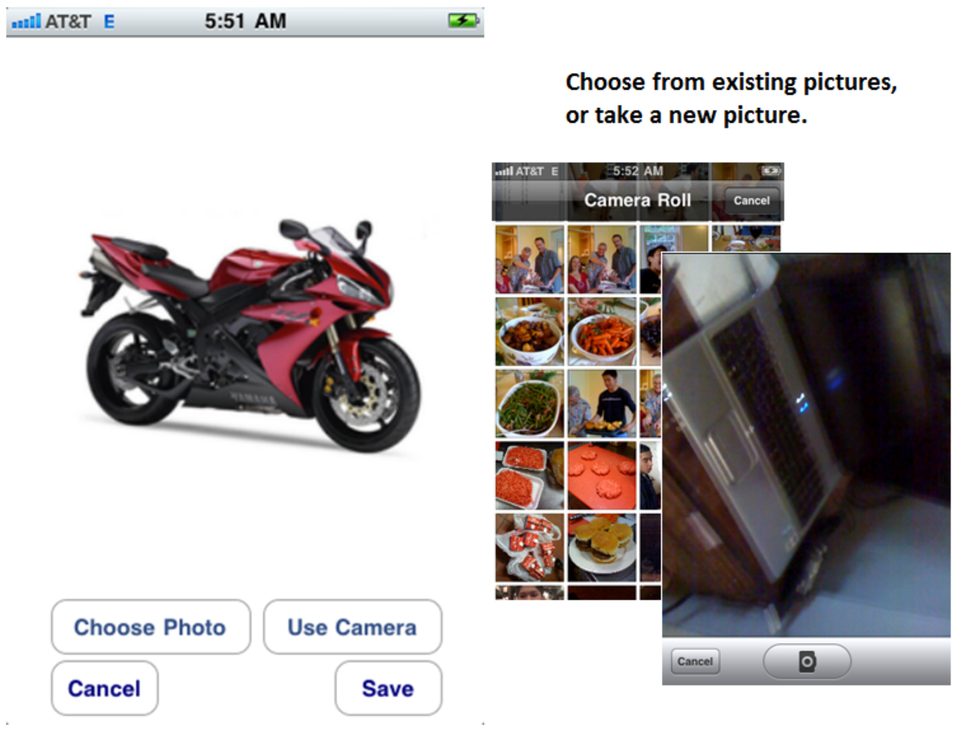
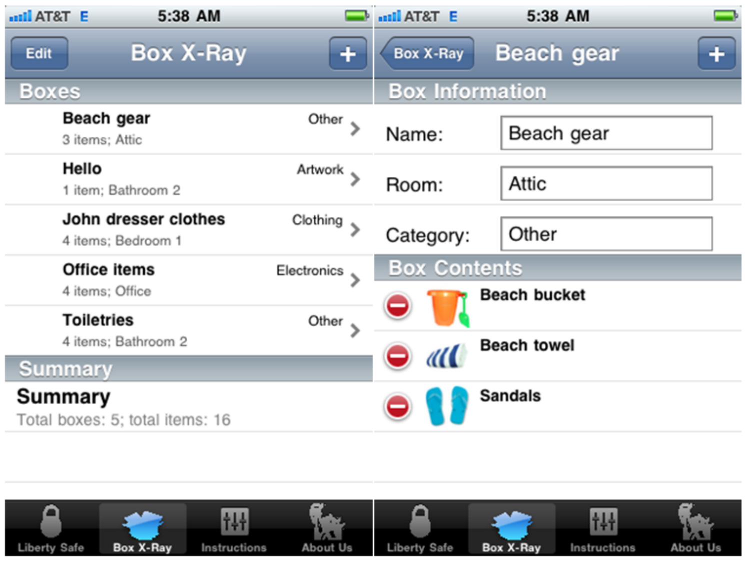


Figure 3-2: Adding photos using DubbleWrap

when he or she had completed a task. Additionally, the app began logging the user's actions within the app and periodically transmitting them back to the server for recording.


AT&T E 5:38 AM

Edit Box X-Ray +

Boxes

Beach gear 3 items; Attic	Other	>
Hello 1 item; Bathroom 2	Artwork	>
John dresser clothes 4 items; Bedroom 1	Clothing	>
Office items 4 items; Office	Electronics	>
Toiletries 4 items; Bathroom 2	Other	>

Summary

Summary
Total boxes: 5; total items: 16

AT&T E 5:38 AM

Box X-Ray Beach gear +

Box Information

Name:	Beach gear
Room:	Attic
Category:	Other

Box Contents

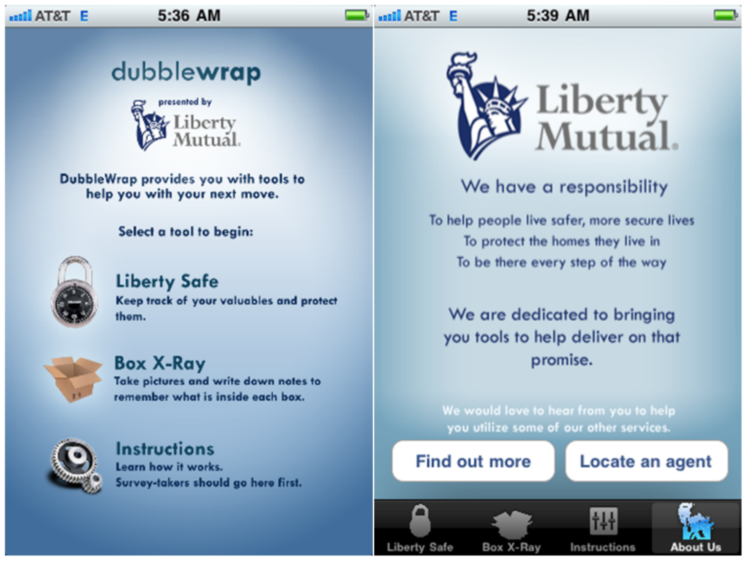
Beach bucket

Beach towel

Sandals

(a) Boxes in the Box X-Ray (b) Viewing a single box

Figure 3-3: Several views of the Box X-Ray feature in DubbleWrap


AT&T E 5:36 AM

dubblewrap

presented by Liberty Mutual

DubbleWrap provides you with tools to help you with your next move.

Select a tool to begin:

Liberty Safe
Keep track of your valuables and protect them.

Box X-Ray
Take pictures and write down notes to remember what is inside each box.

Instructions
Learn how it works.
Survey-takers should go here first.

AT&T E 5:39 AM

Liberty Mutual.

We have a responsibility

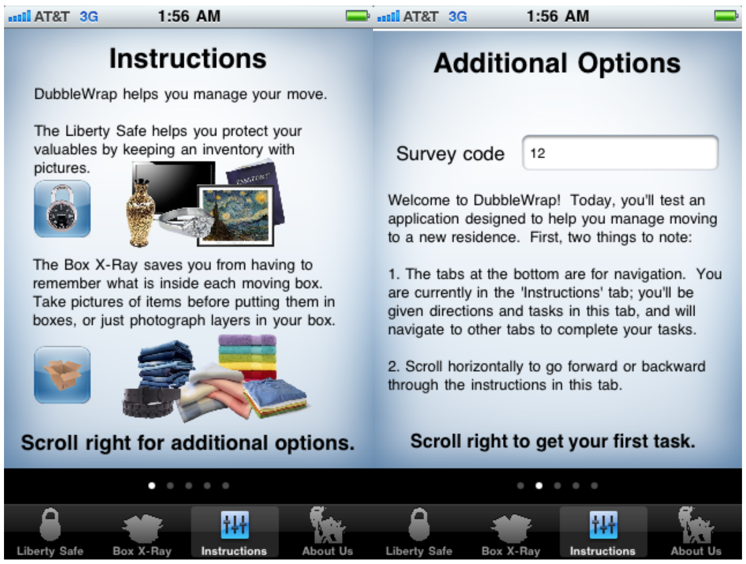
To help people live safer, more secure lives
To protect the homes they live in
To be there every step of the way

We are dedicated to bringing you tools to help deliver on that promise.

We would love to hear from you to help you utilize some of our other services.

(a) The first page (b) The "About Us" tab

Figure 3-4: Liberty Mutual branding in DubbleWrap



Instructions

DubbleWrap helps you manage your move.

The Liberty Safe helps you protect your valuables by keeping an inventory with pictures.

The Box X-Ray saves you from having to remember what is inside each moving box. Take pictures of items before putting them in boxes, or just photograph layers in your box.

Scroll right for additional options.

Additional Options

Survey code 12

Welcome to DubbleWrap! Today, you'll test an application designed to help you manage moving to a new residence. First, two things to note:

1. The tabs at the bottom are for navigation. You are currently in the 'Instructions' tab; you'll be given directions and tasks in this tab, and will navigate to other tabs to complete your tasks.

2. Scroll horizontally to go forward or backward through the instructions in this tab.

Scroll right to get your first task.

(a) The instructions tab (b) Enter a survey code, if applicable

Figure 3-5: The instructions tab

Chapter 4

DubbleWrap: Results

4.1 Pre Measures

To give a sense of the baseline values of each measurement, Table 4.1 shows the pre measures averaged across all iPhone users (groups AB, AC, and AD). There were no significant differences between the three groups on pre measures. The main takeaway from this table is that brand measures were fairly centered around the middle of each response range, and that Liberty Mutual is behind GEICO, Progressive, and Allstate in brand comparisons.

4.2 App Evaluation

App users were asked to evaluate the app. Results of these evaluations are given in Table 4.2. The general conclusion is that users responded positively to the app.

4.3 App Pre/Post

Next we compared pre and post measures for test group (group AB) members using t-tests. The results of these analyses are shown in Table 4.3; starred p-values are significant at $p = 0.10$, and double-starred values are significant at $p = 0.05$. The general conclusion here is that the app positively affected consideration, risk atti-

Measurement	**Mean**	**Std dev**
Likelihood of consideration (1-10; 10 = strongly consider)		
Allstate	6.64	2.550
Farmers	5.81	2.445
GEICO	6.11	2.643
Liberty Mutual	5.50	2.428
Nationwide	5.81	2.390
Progressive	6.32	2.630
State Farm	7.16	2.437
LM attributes: (1-5; 5 = strongly agree)		
Overall, I trust Liberty Mutual.	3.21	.672
Liberty Mutual offers insurance coverage at a better priced.	3.00	.582
Liberty Mutual offers more flexible insurance policies.	3.05	.477
Liberty Mutual offers better customer service in case of an insurance claim.	3.08	.516
Liberty Mutual is open, honest, and transparent.	3.14	.562
Liberty Mutual works hard to meet my changing needs.	3.07	.517
Liberty Mutual is willing to assist and support me.	3.19	.604
I would recommend Liberty Mutual to a friend.	2.98	.667
Liberty Mutual appears more responsible than other insurance companies	3.04	.595
Liberty Mutual is an insurance company that will deliver on promises made	3.17	.581
Liberty Mutual provides believable information	3.23	.601
I have confidence in the information Liberty Mutual provides	3.18	.609
Attitudes toward risk: (1-5; 5 = strongly agree)		
Avoid insurance (I try to deal with insurance-related topics as rarely as possible)	3.58	.976
Plan ahead (I plan most of my activities ahead of time to avoid unexpected events)	3.62	.983
Safe sports (I engage only in safe sports where I have control)	3.22	1.072
Responsible activities (I follow activities which my friends and colleagues would rate as responsible)	3.83	.815
Risk-secured moving (When moving it is important to me to be secured against all kind of risks)	3.73	.875
Afraid of moving damage (I am afraid my belongings might get lost or damaged when moving)	3.39	.970
Full coverage impt. (It is important to me that all of my belongings are covered by my homeowners / renters insurance)	3.94	.917
Most valuable coverage impt (It is important to me that my most valuable belongings are covered by my homeowners / renters insurance)	4.15	.849

Table 4.1: Pre measures across all iPhone users (AB/AC/AD)

Measurement	**Mean**	**Std dev**
Probability of purchase: likelihood of... (1-5; 5 = very likely)		
Getting a quote for homeowners/renters insurance from LM the next time it is needed	2.92	1.117
Getting a quote for auto insurance from LM the next time it is needed	2.85	1.140
Getting information about additional insurance products from LM the next time they are needed	2.85	1.083
Points allocation: preference next time you need insurance (100 points total)		
Allstate	15.89	20.671
Farmers	7.50	14.175
GEICO	13.44	20.364
Liberty Mutual	9.51	15.033
Nationwide	6.68	10.861
Progressive	13.37	20.827
State Farm	21.48	26.824
Other	12.70	25.710

Table 4.1: Pre measures across all iPhone users (AB/AC/AD) (continued)

tudes, and probability of purchase.

Several particularly notable results are colored in grey in Table 4.3. A summary of findings:

- We observed 25% increase in consideration for Liberty Mutual.
- Asked to rate various statements about Liberty Mutual, survey respondents made significant but small increases in their assessment after the stimulus.
- Risk attitudes evidence some significant positive changes, but one small (-2%) negative.
- Likelihood of purchase increased significantly on all measures, including a 16% increase on likelihood of getting a quote for homeowners' insurance from Liberty Mutual.
- On points allocation vs. competitors, Liberty displays an 84% point gain. We also see significant point drops for all competitors; in order: State Farm, GEICO, Progressive, Allstate.

Measurement	**Mean**	**Std dev**
App attributes (1-5; 5 = strongly agree)		
The application is informative	4.06	.821
The application is meaningful	4.00	.874
The application is likeable	4.05	.901
The application is believable	4.10	.758
The application is relevant	4.10	.827
App impact (1-5, 5 = better than before)		
How has the app affected your impression of Liberty Mutual, if at all?	3.86	.860
How has the app affected your likelihood of considering Liberty Mutual for auto or homeowners/renters insurance, if at all?	3.66	.770
Company portrayal (1-5, 5 = strongly agree)		
The app portrays a company that follows through on its responsibilities to its customers	3.75	.750
The app portrays a company that you want to be associated with	3.75	.768
App attributes (1-5, 5=better than before)		
The app is intuitive and easy to use	3.85	.955
The app is helpful in the moving process	3.96	.922
This is an app I would use again during my next move	3.68	1.087
This is an app I would recommend to friends and colleagues	3.66	.979
The app is interesting	4.01	.816
The app led me to think about protecting my possessions	3.66	.939
I found the app to be engaging	3.77	.895
The app led me to consider getting homeowners/renters insurance	2.85	.982
The app is focused on my needs	3.68	.867
The app helps me to structure the process of moving	3.91	.884
I would uninstall the app	3.06	1.080
The app is innovative	4.12	.815
The app is trustworthy	3.61	.765
I have concerns that my data might be used for other purposes than moving	3.24	1.185
I have concerns that the insurance company that sponsored the app has its own interests in mind with this app	3.16	1.025
I would mention the insurance company that sponsored this app when talking to friends and colleagues about the app	3.49	.952

Table 4.2: Evaluation of the iPhone app (group AB)

4.4 Pre/Post differences, app vs TV

We saw in Section 4.3 that the app increased consumers' perception of LM in almost every aspect. However, in order to assess whether the app was more effective than traditional marketing strategies, we would need to compare the level of changes under the app stimulus with the level of changes under a control stimulus. This is done in Table 4.4. Note that we are now dealing with second-order differences: the pre/post differences are first-order, but we are comparing the pre/post differences across two cells.

Some significant findings:

- The app was **60% more effective than TV ads** at increasing consideration.
- The app was more effective than TV ads at making customers feel that
  - Liberty Mutual works hard to meet my changing needs
  - I would recommend Liberty Mutual to a friend
  - When moving it is important to me to be secured against all kind of risks
- The app was **34% more effective than TV ads** at increasing points allocation to LM.
- The app was able to steal points from GEICO and Progressive where TV ads were not.

4.5 Pre/Post differences, app vs website

We repeat the analysis from Section 4.4, but for group AD (website). This is done in Table 4.5.

Variable	**Pre**	**Post**	**Diff**	**t**	**df**	**p**
Likelihood of consideration (1-10; 10 = strongly consider)						
Allstate	6.55	6.37	-0.18	-2.58	518	0.010**
Farmers	5.71	5.58	-0.13	-1.65	499	0.099**
GEICO	6.08	5.70	-0.37	-4.95	518	0.00**
Liberty Mutual	5.38	6.69	1.31	14.36	518	0.00**
Nationwide	5.72	5.65	-0.07	-0.94	507	0.350
Progressive	6.31	6.01	-0.30	-4.01	518	0.00**
State Farm	7.07	6.82	-0.24	-3.57	518	0.00**
Liberty Mutual attributes: (1-5; 5 = strongly agree)						
Overall, I trust Liberty Mutual.	3.19	3.31	0.11	4.32	517	0.00**
LM insurance is better priced.	3.00	3.10	0.10	4.37	517	0.00**
LM policies more flexible.	3.04	3.16	0.11	5.20	517	0.00**
LM has better customer service	3.06	3.19	0.13	5.61	517	0.00**
LM is open, honest, and transparent.	3.12	3.23	0.11	4.52	517	0.00**
LM works hard to meet my changing needs.	3.06	3.41	0.35	12.15	517	0.00**
LM is willing to assist and support me.	3.19	3.48	0.29	9.62	517	0.00**
I would recommend LM to a friend.	2.97	3.24	0.27	9.03	517	0.00**
LM appears more responsible than other insurance companies	3.03	3.25	0.23	7.80	517	0.00**
LM will deliver on promises made	3.15	3.27	0.13	5.09	517	0.00**
LM provides believable information	3.21	3.40	0.18	7.40	517	0.00**
I have confidence in the information LM provides	3.16	3.36	0.20	7.73	517	0.00**
Attitudes toward risk: (1-5; 5 = strongly agree)						
Avoid insurance	3.62	3.65	0.04	1.31	517	0.191
Plan ahead	3.63	3.63	0.00	0.14	517	0.888
Safe sports	3.20	3.18	-0.02	-0.75	517	0.456
Responsible activities	3.82	3.72	-0.09	-3.54	517	0.00**
Risk-secured moving	3.70	3.76	0.06	1.79	517	0.075*
Afraid of moving damage	3.36	3.55	0.19	5.13	517	0.00**
Full coverage is important	3.92	3.87	-0.04	-1.50	517	0.134
Most valuable coverage important	4.12	4.03	-0.09	-3.00	517	0.003**

Table 4.3: Pre/post comparison for app users. *significant at $p = 0.10$; **significant at $p = 0.05$

Variable	**Pre**	**Post**	**Diff**	**t**	**df**	**p**
Probability of purchase: likelihood, next time it is needed, of... (1-5; 5 = very likely)						
Getting quote from LM for homeowners/renters insurance	2.87	3.33	0.45	11.65	517	0.00**
Getting quote from LM for auto insurance	2.79	3.12	0.33	8.47	517	0.00**
Getting information about additional insurance products from LM	2.82	3.21	0.39	9.50	517	0.00**
Points allocation: preference next time you need insurance (100 points total)						
Allstate	15.73	14.58	-1.15	-3.37	516	0.00**
Farmers	6.98	6.44	-0.54	-1.76	497	0.080*
GEICO	13.30	11.60	-1.70	-5.12	516	0.00**
Liberty Mutual	9.18	16.93	7.76	13.12	516	0.00**
Nationwide	6.91	6.40	-0.51	-2.16	505	0.031**
Progressive	14.14	12.51	-1.63	-4.44	516	0.00**
State Farm	21.99	20.23	-1.76	-3.84	516	0.00**
Other	12.37	11.87	-0.50	-1.55	514	0.122

Table 4.3: Pre/post comparison for app users (continued).

There were no significant differences in consideration, risk attributes, likelihood of purchase, or points allocation.

On attributes, some significant differences vs. the website, some in favor of website and some in favor of app. Perhaps this is because the website has specific insurance information while the app does not provide insurance facts. The app still has a greater effect on the statement “Liberty Mutual works hard to meet my changing needs.”

Variable	**AB (App)**	**AC (TV)**	**t**	**p**
Likelihood of consideration (1-10; 10 = strongly consider)				
Allstate	-0.18	-0.55	2.076	0.040**
Farmers	-0.13	-0.35	1.183	0.239
GEICO	-0.37	-0.52	0.822	0.413
Liberty Mutual	1.31	0.82	2.083	0.039**
Nationwide	-0.07	-0.43	1.718	0.089*
Progressive	-0.30	-0.56	1.387	0.168
State Farm	-0.24	-0.47	1.290	0.200
Liberty Mutual attributes: (1-5; 5 = strongly agree)				
Overall, I trust LM	0.11	0.14	-0.367	0.714
LM insurance is better priced.	0.10	0.20	-1.354	0.179
LM policies more flexible.	0.11	0.13	-0.188	0.851
LM has better customer service	0.13	0.11	0.241	0.810
LM is open	0.11	0.14	-0.398	0.692
LM works hard to meet my changing needs.	0.35	0.10	3.558	0.001**
LM is willing to assist and support me.	0.29	0.16	1.642	0.103
I would recommend LM to a friend.	0.27	0.09	2.393	0.018**
LM appears more responsible than other insurance companies	0.23	0.22	0.092	0.927
LM will deliver on promises made	0.13	0.18	-0.803	0.424
LM provides believable information	0.18	0.23	-0.647	0.519
I have confidence in the information LM provides	0.20	0.11	1.049	0.297
Attitudes toward risk: (1-5; 5 = strongly agree)				
Avoid insurance	0.04	0.15	-1.286	0.201
Plan ahead	0.00	-0.06	0.963	0.337
Safe sports	-0.02	0.05	-0.971	0.334
Responsible activities	-0.09	-0.09	-0.010	0.992
Risk-secured moving	0.06	-0.11	2.166	0.032**
Afraid of moving damage	0.19	0.10	0.952	0.343
Full coverage is important	-0.04	-0.06	0.176	0.861
Most valuable coverage important	-0.09	-0.21	1.469	0.144

Table 4.4: Comparing pre/post difference levels for app (AB) vs TV (AC). *significant at $p = 0.10$; **significant at $p = 0.05$

Variable	**AB (App)**	**AC (TV)**	**t**	**p**
Probability of purchase: likelihood, next time it is needed, of... (1-5; 5 = very likely)				
Getting quote from LM for homeowners/renters insurance	0.45	0.28	1.564	0.121
Getting quote from LM for auto insurance	0.33	0.30	0.271	0.787
Getting information about additional insurance products from LM	0.39	0.34	0.406	0.685
Points allocation: preference next time you need insurance (100 points total)				
Allstate	-1.15	-1.91	1.041	0.300
Farmers	-0.54	-0.37	-0.231	0.818
GEICO	-1.70	0.08	-2.224	0.028
Liberty Mutual	7.76	5.78	1.702	0.091
Nationwide	-0.51	-0.28	-0.553	0.581
Progressive	-1.63	0.46	-2.061	0.042
State Farm	-1.76	-3.03	1.063	0.290
Other	-0.50	-0.80	0.397	0.692

Table 4.4: Comparing pre/post difference levels for app (AB) vs TV (AC) (continued).

Variable	**AB (App)**	**AD (Website)**	**t**	**p**
Likelihood of consideration (1-10; 10 = strongly consider)				
Allstate	-0.18	-0.59	2.203	0.030
Farmers	-0.13	-0.47	1.330	0.187
GEICO	-0.37	-0.53	0.695	0.488
Liberty Mutual	1.31	1.38	-0.243	0.808
Nationwide	-0.07	-0.68	3.068	0.003**
Progressive	-0.30	-0.55	1.055	0.294
State Farm	-0.24	-0.62	1.837	0.069
Overall	0.11	0.11	0.111	0.912
Liberty Mutual attributes: (1-5; 5 = strongly agree)				
Overall, I trust LM	0.11	0.11	0.111	0.912
LM insurance is better priced.	0.10	0.22	-1.691	0.094*
LM policies more flexible.	0.11	0.28	-2.408	0.018**
LM has better customer service	0.13	0.07	1.081	0.282
LM is open	0.11	0.16	-0.819	0.415
LM works hard to meet my changing needs.	0.35	0.17	2.805	0.006*
LM is willing to assist and support me.	0.29	0.33	-0.525	0.601
I would recommend LM to a friend.	0.27	0.29	-0.273	0.785
LM appears more responsible than other insurance companies	0.23	0.14	1.115	0.267
LM will deliver on promises made	0.13	0.09	0.539	0.591
LM provides believable information	0.18	0.34	-2.449	0.016**
I have confidence in the information LM provides	0.20	0.20	-0.007	0.994
Attitudes toward risk: (1-5; 5 = strongly agree)				
Avoid insurance	0.04	-0.05	1.079	0.283
Plan ahead	0.00	0.01	-0.169	0.866
Safe sports	-0.02	-0.05	0.501	0.617
Responsible activities	-0.09	-0.07	-0.427	0.670
Risk-secured moving	0.06	0.03	0.518	0.606
Afraid of moving damage	0.19	0.08	1.380	0.170
Full coverage is important	-0.04	-0.05	0.120	0.905
Most valuable coverage important	-0.09	-0.13	0.524	0.601

Table 4.5: Comparing pre/post difference levels for app (AB) vs TV (AC). *significant at $p = 0.10$; **significant at $p = 0.05$

Variable	**AB (App)**	**AC (TV)**	**t**	**p**
Probability of purchase: likelihood, next time it is needed, of... (1-5; 5 = very likely)				
Getting quote from LM for homeowners/renters insurance	0.45	0.50	-0.478	0.634
Getting quote from LM for auto insurance	0.33	0.32	0.116	0.908
Getting information about additional insurance products from LM	0.39	0.49	-0.910	0.365
Points allocation: preference next time you need insurance (100 points total)				
Allstate	-1.15	-2.05	0.929	0.355
Farmers	-0.54	-1.00	0.558	0.578
GEICO	-1.70	-1.04	-0.591	0.556
Liberty Mutual	7.76	8.37	-0.484	0.629
Nationwide	-0.51	-0.64	0.200	0.842
Progressive	-1.63	-1.42	-0.230	0.818
State Farm	-1.76	-1.07	-0.677	0.500
Other	-0.50	-1.12	0.474	0.637

Table 4.5: Comparing pre/post difference levels for app (AB) vs TV (AC) (continued).

Chapter 5

DubbleWrap: Analysis and Evaluation

5.1 Discussion

Trust-based marketing has been studied and implemented in various forms over the past few years; strategies range from TV spots that focus on a brand's reputation of trustworthiness to online product advisors that provide consumers objective advice. Web and mobile applications represent the forefront of this movement because they offer the opportunity for greater interactivity—compare watching a commercial on trust to interacting with a useful mobile app that provides good advice—and because they can much more closely integrate into the consumer's life. Additionally, they can create actual value for a consumer; the most value a TV commercial can add is a laugh or a conversation topic at the bar.

The DubbleWrap study demonstrates exciting results for researchers and marketers considering trust-based app marketing strategies. At a basic level, the results point to superior brand effects compared to television commercials, and similar effects compared to website stimuli, with further research required to further weigh the two.

A few points bear clarification to fully appreciate the significance of the results.

5.1.1 App Design

The app designed for this study, DubbleWrap, exists to conduct research. It does not have an extensive featureset; it does not have cutting-edge graphics; it sits somewhere between a proof of concept and a final release candidate. It exists to provide a simple service to consumers, and to allow us to test the effects of that simple service.

If this simple app can move brand measures the way it did, imagine what a fully-featured app written by a software team specializing in iPhone development, and supported by a top-notch graphic designer, could do.

Even within this design, there are a number of features that could be easily implemented to provide additional value to consumers. Textual descriptions and a search feature to search these descriptions would be helpful for finding items. A mass imaging mode, which would allow the user to take a series of quick snapshots (all to be added to the same box) would make cataloguing all of the items in a box much easier. Additional features for assisting with a move, such as locating a mover or a truck rental company, or information about changing one's address, could be wrapped into another tab. A website coupled with the app, to which the user could upload his inventories, would allow the user to keep track of his possessions online. In fact, the app could be expanded into an entire webapp for keeping track of one's valuables.

In short: DubbleWrap (in its current implementation) is only the beginning of a possibly much larger marketing strategy, which now has data to support investment.

5.1.2 Combating Selection Bias

Every effort was made to avoid selection bias in the study. We elected to use iPhone users who were willing to download and test an app in our control groups, even though research protocols might allow for a wider definition of control, such as users that passed the other requirements about moving and owning a smartphone, but who did not own an iPhone. We collected data for these non-iPhone users in response to control stimuli; repeating the analyses for these users, we found that app evaluation was generally more favorable with the wider control group. This analysis appears in

Appendix A.

Additionally, we closely monitored attrition rate to make sure that no stimulus group dropped out at an unreasonably high rate. Waterfall diagrams for each experimental group are given in Fig. 5-1. In the app group, users could drop out by failing to download the app, by failing to get to the last task, or by failing to complete the post survey—a large number of filters—so we worried that they might drop at a higher rate than the groups receiving a (comparatively easier) control stimulus. However, attrition rates were at acceptabe levels. In fact, the real concern ended up being the website task, which gave users two chances to find a correct answer to a question about the website before dropping them from the analysis.

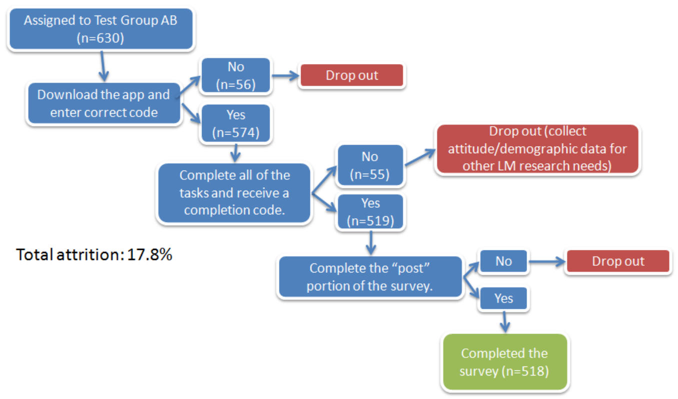
5.1.3 Branded vs Unbranded

Another caveat is regarding the use of the Liberty Mutual branding on the app.

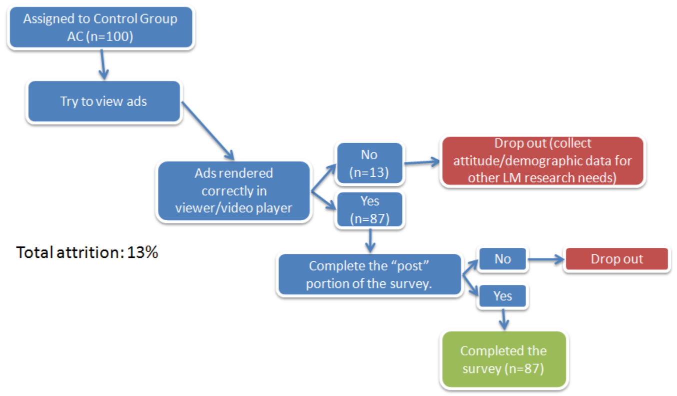
Initially, we developed two versions of the app—a branded version with LM logos, and an unbranded version with no mention of Liberty Mutual. However, the unbranded version performed very similarly to the branded version in the tests; t-tests comparing the two found essentially no significant differences. We hypothesize that the unbranded version was labeled with LM in minds of consumers due to the presence of questions in the pre- and post-surveys which were about Liberty Mutual only. Given these results, we grouped the results of the branded and unbranded apps together to become group AB.

5.1.4 Demand Effects

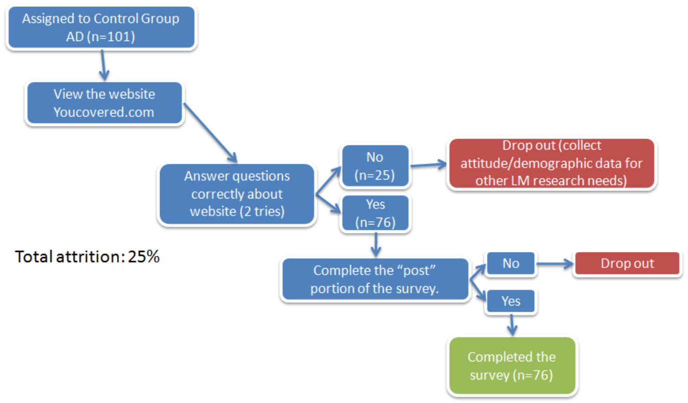
Due to the presence of questions in both the pre- and post-surveys which were only asked about Liberty Mutual, it was possible for respondents to identify the sponsor and give the "correct' answer. This limitation could be corrected in future studies by using another control with a "filler" task not related at all to Liberty Mutual, or by using an unbranded version of the app with the Liberty-specific questions asked about a different brand (e.g. Nationwide).


Assigned to Test Group AB (n=630)
Download the app and enter correct code
No (n=56)
Drop out
Yes (n=574)
Complete all of the tasks and receive a completion code.
No (n=55)
Drop out (collect attitude/demographic data for other LM research needs)
Yes (n=519)
Total attrition: 17.8%
Complete the "post" portion of the survey.
No
Drop out
Yes
Completed the survey (n=518)

(a) App stimulus (group AB)


Assigned to Control Group AC (n=100)
Try to view ads
Ads rendered correctly in viewer/video player
No (n=13)
Drop out (collect attitude/demographic data for other LM research needs)
Yes (n=87)
Total attrition: 13%
Complete the "post" portion of the survey.
No
Drop out
Yes
Completed the survey (n=87)

(b) TV stimulus (group AC)


Assigned to Control Group AD (n=101)
View the website Youcovered.com
Answer questions correctly about website (2 tries)
No (n=25)
Drop out (collect attitude/demographic data for other LM research needs)
Yes (n=76)
Total attrition: 25%
Complete the "post" portion of the survey.
No
Drop out
Yes
Completed the survey (n=76)

(c) Website stimulus (group AD)

Figure 5-1: Waterfall diagrams for each experimental group

Even if there was a demand effect, however, the comparisons to TV ads are still valid since we assume the demand effect would affect them equally.

5.2 Research Conclusions

5.2.1 Trust-based apps are positive marketing tool

The DubbleWrap study demonstrates the efficacy of trust-based apps as a marketing strategy. In addition to receiving positive ratings about a variety of attributes ("the app is informative", "the app is believable", "this is an app I would use again during my next move"), the app increased many of the brand ratings. Among other highlights, the app increased consideration by 25%, probability of purchase for rental insurance by 16%, and preference for Liberty Mutual by 84%.

Moreover, the app was more effective than TV ads. We saw a 60% greater increase in consideration and a 34% greater increase in preference for Liberty Mutual compared to TV ads.

The results strongly suggest that Liberty Mutual consider incorporating a trust-based app in their marketing strategy. With DubbleWrap functional and in the app store, developing a production version would be fairly simple.

With a demonstration of efficacy completed, our next question concerns the costs of this strategy. How much does it cost to drive people to the app? What is the cost relative to forced exposure to ads, or to driving users to a website like `youcovered.com`?

5.2.2 Contributions

The DubbleWrap study demonstrates an advanced methodology for comparing old and new media that can be extended to future marketing meta-analysis. As researchers work to understand the role of apps in marketing, we believe this is the first clinical experimental evidence comparing the efficacy of trust-based apps and television/web advertisements. Future work would include running a market experiment to validate app impact, using metrics such as clickthrough rate to the website or effect on product sales.

Chapter 6

DreamCar: Background

6.1 Introduction

General Motors is the world's second-largest automaker, producing cars under the Buick, Cadillac, Chevrolet, GMC, Opel, Vauxhall, and Holden brands. An American multinational corporation, it produces cars and trucks in 31 countries and does business in 157 countries. GM's global scale accounts for its status as one of the world's largest companies, with annual revenue of $135B (2010) and the world's largest IPO on November 18, 2010.

Branding has always been a critical component of GM's strategy. After emerging from bankruptcy in July 10, 2009, GM reorganized its brand portfolio, eliminating its Pontiac, Saturn, and Hummer brands and selling Saab. It is now focusing in North America on four core brands, Buick, Cadillac, GMC, and Chevrolet.

With more invested in fewer brands, the stakes have increased. As GM's leadership looks to the future, it seeks to build its brands for future success. One way of investing in the future is by building a relationship early with younger consumers, to whom we will refer as **Generation Y**. These consumers may not be in the market for a new car for 15-20 years; in 2008 the age of the average new car purchaser was 48. However, in addition to paving the way for future sales, building a relationship with younger consumers can make GM cars aspirational for a younger generation that drives trends followed by older consumers. Given the pattern of consumer trends being started by younger buyers and later imitated by their older counterparts—for example in social media or fashion—a more immediate effect of marketing to Gen Y should not be discounted.

GM was thus faced with a natural question: how to market to Gen Y users? In considering possible strategies, several broad themes emerged.

The first: Gen Y users have adapted to technology quickly, and have integrated it into their daily lives to obtain and share information, make connections, and consume and produce entertainment. The average adult sends ten texts per day, while the average teen (ages 12-17) sends five times more texts. One in three teens sends over one hundred texts per day.[1] Gen Y has adopted the internet for seemingly everything; the phenomenon of the *haul video* exemplifies this trend. A haul video is a video recording where a shopper (usually a young woman) shows items he or she recently purchased, adding commentary and product information. The phenomenon is a growing trend, with over 160,000 haul videos on YouTube and with the most notable haul video bloggers earning sponsorship deals from major brands.

The second: mobile and web applications are pervasive. There are over 350,000 apps in the iTunes app store, which has served over ten million app downloads as of January 2011.[2] There are over 500,000 web apps on Facebook. In the automobile space, several of GM's competitors have webapps designed to engage users; examples include Toyotaracing.com, Ford on Facebook, and Mazda's Driverville webapp.

6.2 Objective

In light of these findings, GM and MIT converged upon the goal of developing a test webapp to engage Gen Y consumers, and measuring its efficacy in affecting brand measures. Given the large degree of variability in app design, the goal would be to develop an app that could be easily varied along several dimensions, and to test each combination of variables. This would be more informative than testing a single app

[1]Pew Research Center, `http://pewresearch.org/pubs/1716/adults-cell-phones-text-messages`
[2]`http://www.apple.com/pr/library/2011/01/22appstore.html`

design, since the output would show how important each dimension is to efficacy as a promotional vehicle.

The dimensions selected were the following:

Customizability How much control do consumers want over their app experience? Naively, we might expect a higher level of customizability to automatically translate to higher app ratings and a more effective brand perception shift, but this is expectation is not necessarily valid. In some cases, simpler is better.

Discovery To what extent should the app actively push GM products? Will an app that recommends GM products move the brand ratings up because more consumers will follow the links? Or will an app that recommends third-party auto blogs gain more credibility and result in better brand ratings?

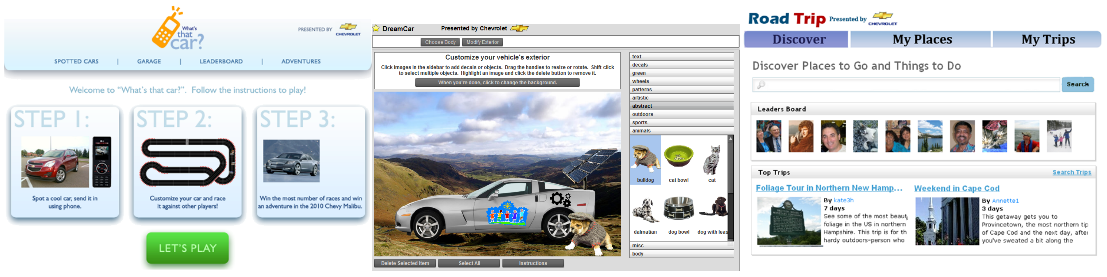
Embeddedness How does embeddedness in social networks affect app dynamics, usage incentives, and overall experience? Does an app become more meaningful for users if it can be shared with friends? Is an app that takes the format of a Facebook app effective if it does not share its activity to users' friends?

In order to provide greater alignment between the target audience and the product space, GM elected to brand the app with the Chevrolet logo. The next step was to figure out what sort of app to build.

6.3 Concepts Tested

Based on the experimental dimensions we sought to measure, we developed three concepts for webapps: "What's That Car?", "DreamCar," and "Road Trip!". Static mockups of each concept were built and prototyped in focus groups. These concepts, shown below in Figure 6-1.

- “What’s That Car?”: Take photos of cool cars, have them identified, and “race” them.
- “DreamCar”: Design a cool car, share it with friends, rate others’ designs.
- “Road Trip!”: Learn about, review, and check in with pictures at cool roadside locations.


What's that car?
PRESENTED BY
SPOTTED CARS | GARAGE | LEADERBOARD | ADVENTURES
Welcome to "What's that car?". Follow the instructions to play!
STEP 1:
Spot a cool car, send it in using phone.
STEP 2:
Customize your car and race it against other players!
STEP 3:
Win the most number of races and win an adventure in the 2010 Chevy Malibu.
LET'S PLAY
DreamCar
Presented by Chevrolet
Customize your vehicle's exterior
Delete Selected Item
Select All
Instructions
Road Trip
Presented by
Discover
My Places
My Trips
Discover Places to Go and Things to Do
Search
Leaders Board
Top Trips
Search Trips
Foliage Tour in Northern New Hamp...
Weekend in Cape Cod

(a) “What’s That Car?” (b) “DreamCar” (c) “Road Trip!”

Figure 6-1: Webapp concepts for GM Gen Y Study

Some key findings from the focus groups:

- Users prefer simple “time killer” apps; nothing too complicated.
- Control and customization are very important.
- There should be real participation and competition to the game.
- Social aspect and “bragging rights” help engage users.

Based on the focus groups, we selected the “DreamCar” app, which combines simplicity and fun with ease of testability.

6.4 App Specification

“DreamCar” is a game where users create their own cars and compete with their friends to see who can create the most popular car. Winners of this “competition” win a weekend rental with a Chevrolet car, and are asked to share their experiences (both with respect to the car, and with respect to their adventures over the weekend) with other DreamCar users. This last aspect of the app experience underscores the view that *experiences and stories are the currency of the social economy*, and that offering the opportunity for users to stand in the spotlight (if they so elect) has real social value.

A more complete description of the app experience, as given in the focus groups, follows:

> **DreamCar Game Description**
>
> If you could design the perfect car for you, what would it look like? This app gives you the power to create custom car designs that reflect your personal style, right in the palm of your hand. You can choose it all: colors, features, and body builds. Then, when you're ready to reveal your masterpiece to the world, you can post it on your Facebook page and sit back while the compliments roll in.
>
> You can also check out the custom designs of other users of the 'Dream Car' app and see whether the competition stacks up to your creation. The best part? You can vote on each custom car design, and the car with the most votes each week wins a free weekend with a brand new Chevrolet! Entering to win is simple: get the app, share your masterpiece with your friends, and spread the word that you need their votes. If you win, your friends will want to hear all about your amazing weekend. And so will we, so we'll spotlight you and your adventures on our website.

6.5 Experimental Methods

6.5.1 Cohort Definition

Generation Y is a term used to describe the demographic group following Generation X. As with Generation X, there is no universally agreed upon time range for Generation Y; as a generalization, the group is defined to range in birth year from around 1980 to the early 2000s. In our study, we defined Gen Y to be users under the age of 30.

However, we considered the possibility that other older consumers with similar behaviors to Gen Y might respond similarly to our stimuli. To incorporate these consumers into our tests, we carried out a substudy to identify Gen Y traits by asking Gen Y market research participants about their attitudes and behaviors. Based on our findings in the substudy, we developed a screener questionnaire for older respondents to gague their similarity to Gen Y; subject responses were scored compared to a discriminant function benchmark, and scores above a given threshold were accepted into the study as well. The screener questions are given in Fig. 6-2.

6.5.2 Experimental Groups

A total of $n = 1710$ market research test subjects, split across twelve cells, participated in the DreamCar study.

For each of the three dimensions (customizability, discovery, and embeddedness), we developed two sets of specifications for the app, one with a higher level along that dimension, and one with a lower level; these specifications are detailed in Chapter 8. The app experience was then served in eight different versions ($2^3 = 8$), each to a different group of experimental users. A ninth group of users was given a control stimulus (two thirty-second TV commercials) and asked the same pre-stimulus and post-stimulus questions to serve as a benchmark against which to compare the app. Each of these nine cells had $n = 150$, with 120 members in the age range from 21 to 30 (Gen Y proper) and 30 members over 30 but qualified as Gen Y-like.

Additionally, three cells of younger Gen Y users (age range 18 to 21) were assembled; two cells served as experimental cells, which varied only along the embeddedness dimension, and a third cell served as a control group. Each of these cells had $n = 120$.

A chart summarizing cell composition is shown in Figure 6-3.

6.5.3 Research Environment

The study was conducted in a market research environment. Subjects were recruited from several major consumer panels and assigned to the aforementioned cells using standard quota-filling algorithms, which attempt to balance along several demographic variables such as gender.

1. Do you own a smart phone (with internet access, texting, email and app capabilities)?
2. Do you have a land line in your name?
3. Of which of the following sites are you currently a member? Facebook, MySpace, LinkedIn, Twitter, Foursquare, myYearbook, Some other (SPECIFY), None of the above
4. How many Facebook friends do you have?
5. To what extent do you agree or disagree with the following statements? (5 pts scale: Strongly Agree to Strongly Disagree)
   - (a) It is important to me to make more money and always be improving my standard of living.
   - (b) I like being noticed by others.
   - (c) It bothers me if people disapprove of my decisions.
6. I have a body piercing in a place other than an ear lobe. (Y/N)
7. Please read each statement and answer accordingly (Y/N)
   - (a) I watch TV shows online.
   - (b) I use an online-only radio service (e.g., Pandora, AOL radio, etc).
   - (c) I make efforts to avoid paying for entertainment (e.g., music, movies)
   - (d) I keep in touch with friends online (Facebook, chat, MySpace) more so than by phone.
   - (e) I regularly use an instant messaging program (skype, ichat, BBM, Gtalk) to talk to friends.
   - (f) I have used a video-game console (e.g., Playstation, Wii), in the past 24 hours.
   - (g) I regularly (every week) watch video on my mobile phone.
   - (h) I have posted a video of myself online
8. How many total texts do you usually send per day?
9. How many hours do you spend watching television in a typical week?

Figure 6-2: Screener survey to identify older Gen Y-like participants

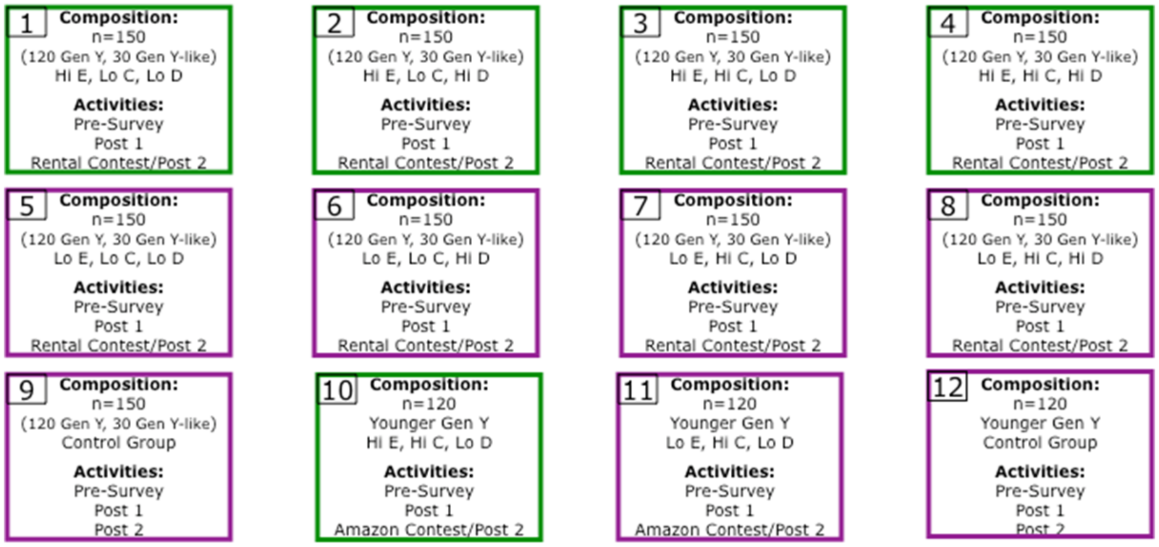

1 Composition: n=150 (120 Gen Y, 30 Gen Y-like) Hi E, Lo C, Lo D Activities: Pre-Survey Post 1 Rental Contest/Post 2
2 Composition: n=150 (120 Gen Y, 30 Gen Y-like) Hi E, Lo C, Hi D Activities: Pre-Survey Post 1 Rental Contest/Post 2
3 Composition: n=150 (120 Gen Y, 30 Gen Y-like) Hi E, Hi C, Lo D Activities: Pre-Survey Post 1 Rental Contest/Post 2
4 Composition: n=150 (120 Gen Y, 30 Gen Y-like) Hi E, Hi C, Hi D Activities: Pre-Survey Post 1 Rental Contest/Post 2
5 Composition: n=150 (120 Gen Y, 30 Gen Y-like) Lo E, Lo C, Lo D Activities: Pre-Survey Post 1 Rental Contest/Post 2
6 Composition: n=150 (120 Gen Y, 30 Gen Y-like) Lo E, Lo C, Hi D Activities: Pre-Survey Post 1 Rental Contest/Post 2
7 Composition: n=150 (120 Gen Y, 30 Gen Y-like) Lo E, Hi C, Lo D Activities: Pre-Survey Post 1 Rental Contest/Post 2
8 Composition: n=150 (120 Gen Y, 30 Gen Y-like) Lo E, Hi C, Hi D Activities: Pre-Survey Post 1 Rental Contest/Post 2
9 Composition: n=150 (120 Gen Y, 30 Gen Y-like) Control Group Activities: Pre-Survey Post 1 Post 2
10 Composition: n=120 Younger Gen Y Hi E, Hi C, Lo D Activities: Pre-Survey Post 1 Amazon Contest/Post 2
11 Composition: n=120 Younger Gen Y Lo E, Hi C, Lo D Activities: Pre-Survey Post 1 Amazon Contest/Post 2
12 Composition: n=120 Younger Gen Y Control Group Activities: Pre-Survey Post 1 Post 2

Figure 6-3: The twelve experimental cells in the DreamCar experiment.

Participants in high-embeddedness (Hi-E) cells (cells 1, 2, 3, 4, and 10) were embedded in separate communities developed by the market research company. The communities feature the basics of an online community in 2011—forums, user profiles, media gallery, commenting, and "friending" between users. However, since each community was created only for the duration of the study, the users in each community did not know each other, so they were given a "scavenger hunt" task to build some low-salience relationships between users. The task asked users to find at least ten other users with whom they have something in common, and friend them. Users who completed the task were entered into a drawing for an Amazon gift card.

6.5.4 Measured Variables

Measured variables include: brand awareness, consideration, trust, brand equity, preference, and probability of purchase.

6.6 Application Flow

Participants were given a survey three times during the testing process: before beginning the experience, after a week of using the DreamCar app, and after winners have rented their cars and stories have been posted. A flowchart summarizing the experiment for each combination of experimental variables is given in Figure 6-4.

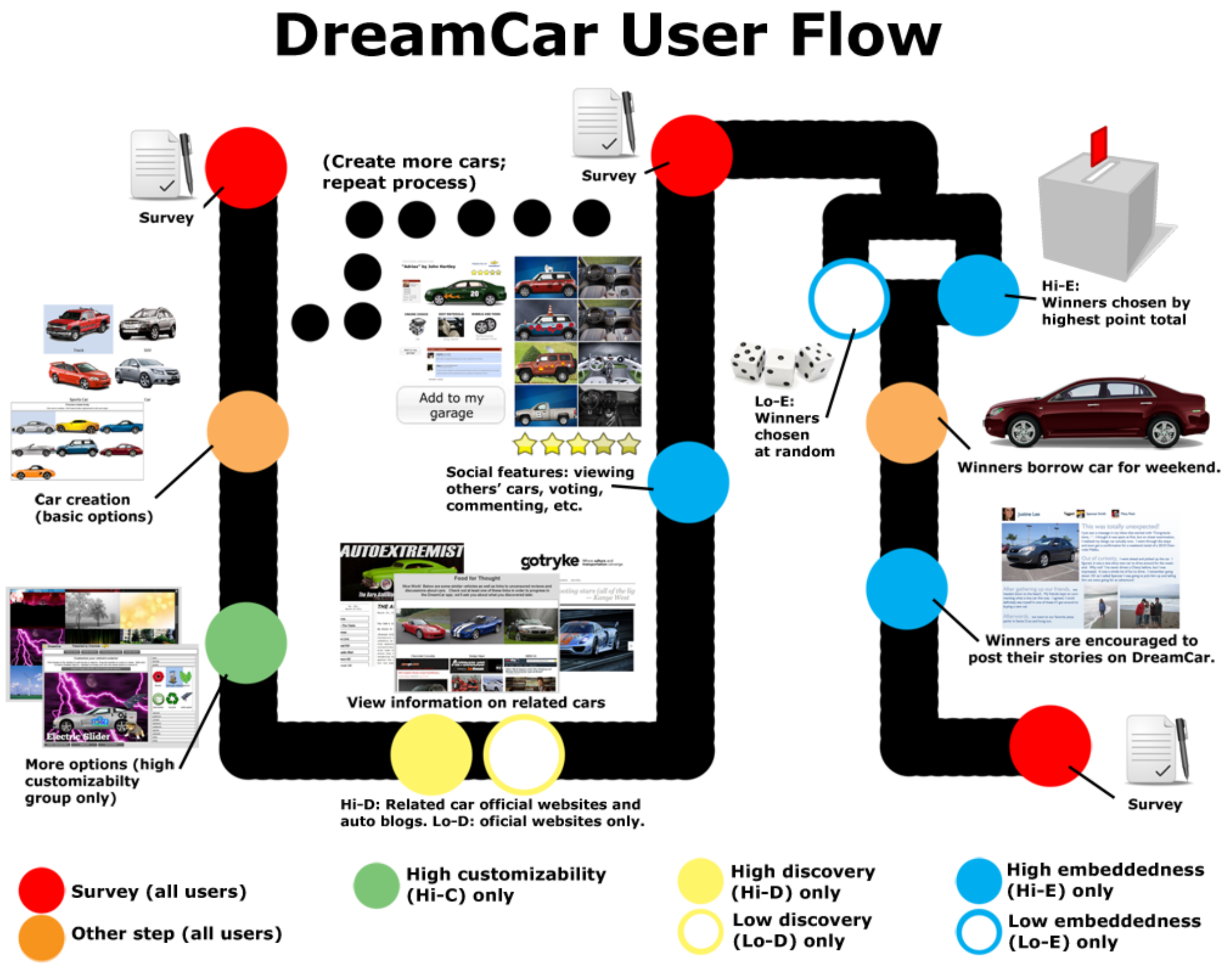

DreamCar User Flow
Survey
(Create more cars; repeat process)
Survey
Hi-E: Winners chosen by highest point total
Add to my garage
Lo-E: Winners chosen at random
Winners borrow car for weekend.
Car creation (basic options)
Social features: viewing others' cars, voting, commenting, etc.
Winners are encouraged to post their stories on DreamCar.
View information on related cars
More options (high customizabilty group only)
Hi-D: Related car official websites and auto blogs. Lo-D: oficial websites only.
Survey
Survey (all users)
Other step (all users)
High customizability (Hi-C) only
High discovery (Hi-D) only
Low discovery (Lo-D) only
High embeddedness (Hi-E) only
Low embeddedness (Lo-E) only

Figure 6-4: User flow through the DreamCar experiment.

Chapter 7

DreamCar: Design

7.1 App Walkthrough

7.1.1 Selecting a car body

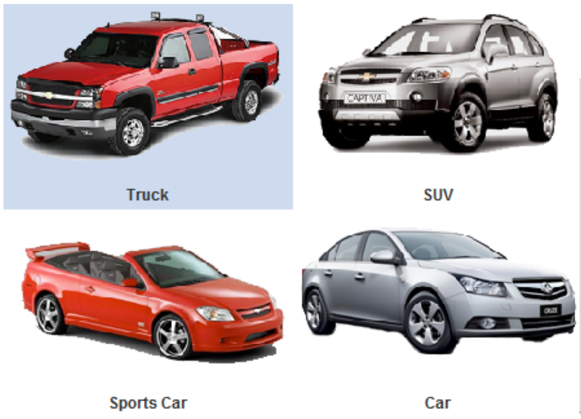
First the user is directed to select a car body by navigating through a series of menus, shown in Fig. 7-1. The options to choose a color variant and stretch are only available in the Hi-C cells.

7.1.2 Selecting a car theme

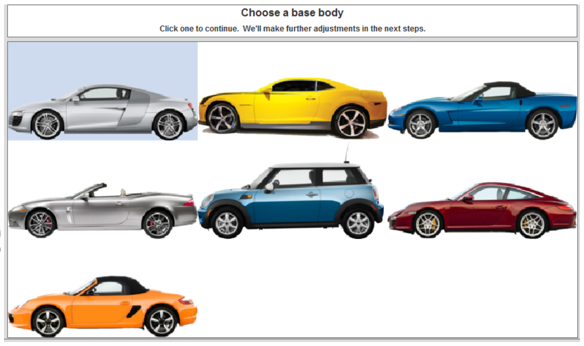
In the Hi-C cells, the user is then asked to choose a theme for his or her car. This choice affects the background with which the car is initialized; for instance, choosing the `outdoorsy` theme starts the car in front of a mountain range, while choosing the `sports fan` puts the car in front of a football stadium. The choice of themes is shown in Fig. 7-2.

7.1.3 The exterior designer

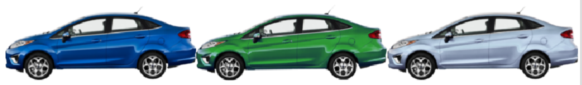
At this point, the user is presented with an interface for modifying the car's exterior by overlaying images or text. A menu of available components, organized by type, appears on the right side of the screen. Users click on components in the menu



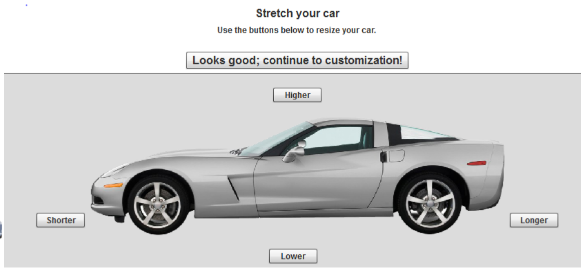
(a) Choosing a car type in DreamCar


Choose a base body
Click one to continue. We'll make further adjustments in the next steps.

(b) Choosing a car model in DreamCar



(c) Choosing a car color variant in DreamCar


Stretch your car
Use the buttons below to resize your car.
Looks good; continue to customization!
Higher
Shorter
Longer
Lower

(d) Choosing a car stretch in DreamCar

Figure 7-1: Several steps in the DreamCar body selection process

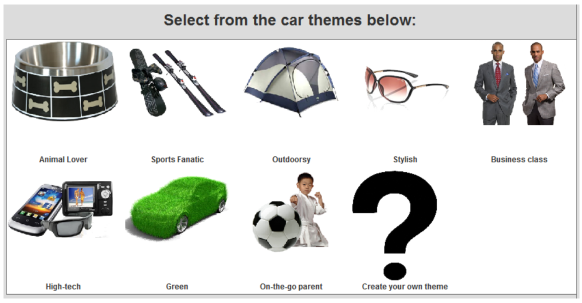


Figure 7-2: Choosing a car theme (Hi-C only)

to add them to the canvas, then drag the components around on the canvas by clicking on them. Resizing and rotations are also permitted. Pressing the delete key or clicking the “delete selected” button while components are selected removes the items. Additional interfaces include a utility for selecting all components and the car (useful for moving everything as a group), group selections using the shift key, and an instructions popup.

In the Hi-C version of the app, there are over 150 components that can be added to the car. In the Lo-C version, this list is reduced to about 30 components—a deliberately small number to ensure sufficient differences between Hi-C and Lo-C conditions. A selection of components is shown in Fig. 7-4.

7.1.4 Selecting a background

Users then have the option of changing their car’s background. The currently-selected background is presented in the top left corner and the user can choose from other options, as shown in Fig. 7-6. The Hi-C version has 44 background choices and the Lo-C version has 9.

After selecting a background, the user can make further changes.

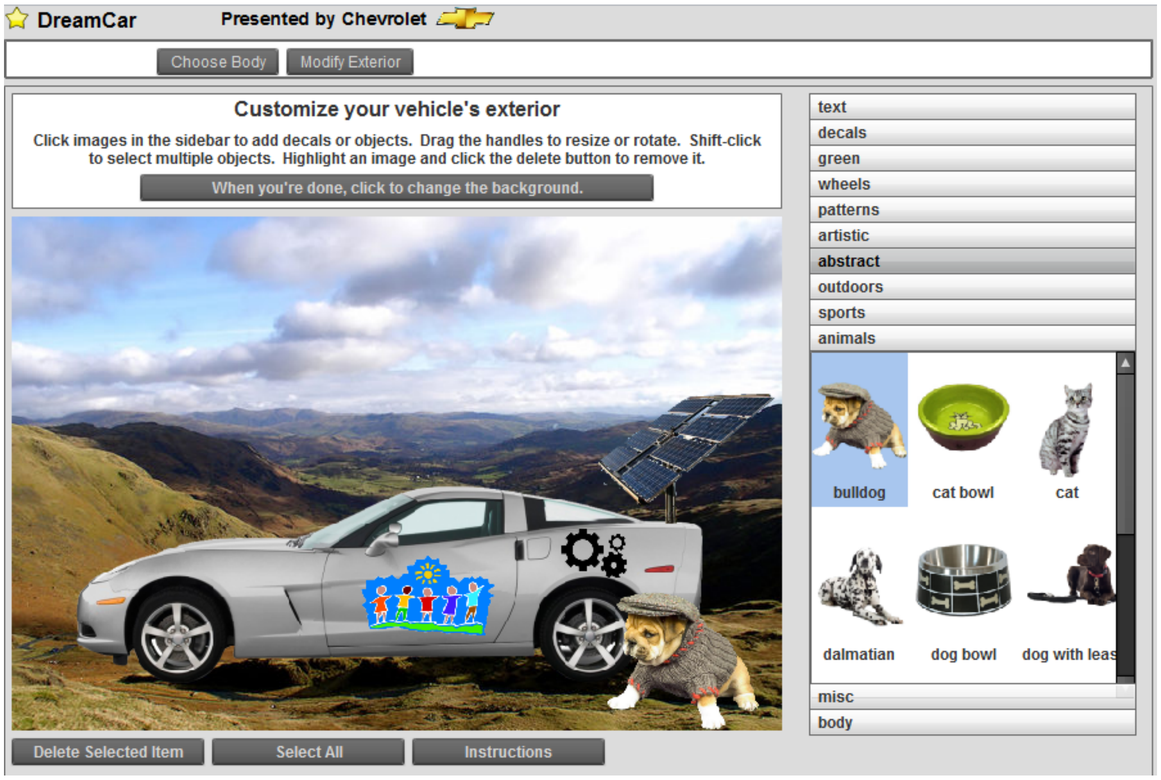
7.1.5 Selecting an interior

The user is then given a choice of interiors, as shown in Fig. 7-7. The Hi-C version features 49 choices, while the Lo-C only features 31 choices. The most exotic interiors, such as the leopard-print interior, are only available in the Hi-C version.

Subsequently, the user is taken to the interior designer (shown in Fig 7-8), which is very similar to the exterior designer but with some different options for components.

7.1.6 Endgame

When the user is satisfied with the interior, the final step is to name the car. After this, the images are submitted to the server, where they appear in the DreamCar gallery.

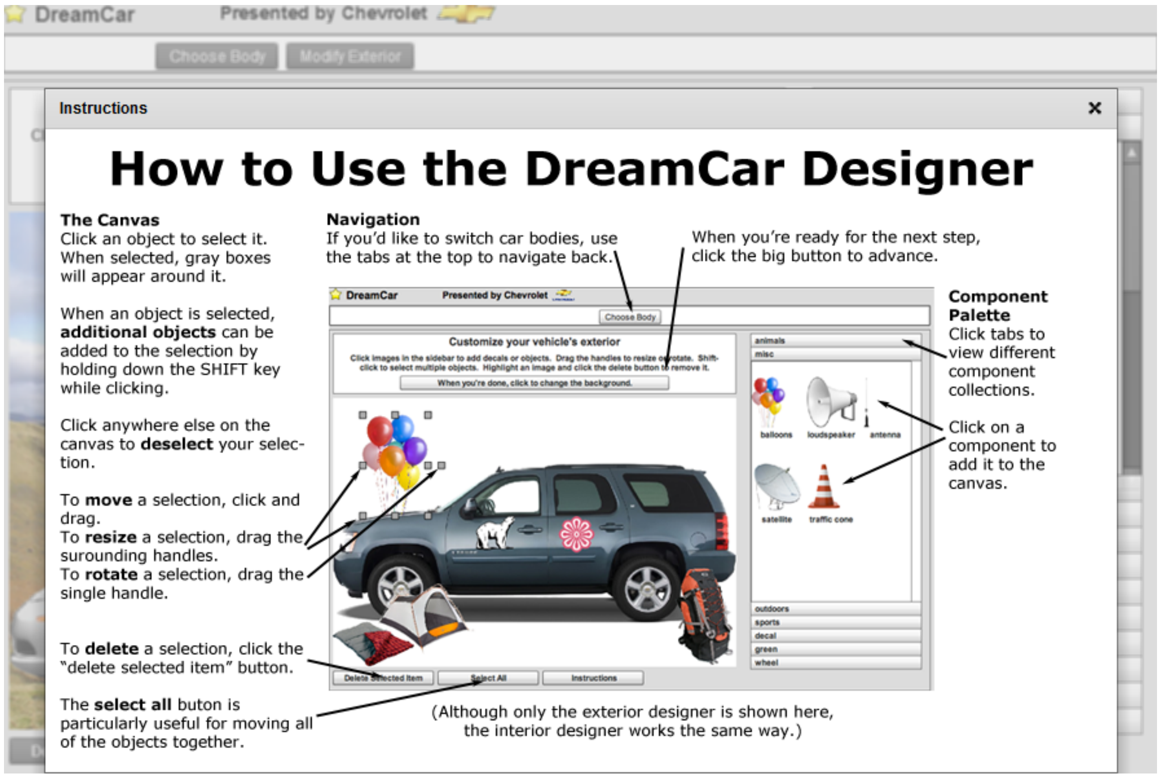


Customize your vehicle's exterior

Click images in the sidebar to add decals or objects. Drag the handles to resize or rotate. Shift-click to select multiple objects. Highlight an image and click the delete button to remove it.

When you're done, click to change the background.

(a) The exterior designer



How to Use the DreamCar Designer

The Canvas
Click an object to select it. When selected, gray boxes will appear around it.

When an object is selected, **additional objects** can be added to the selection by holding down the SHIFT key while clicking.

Click anywhere else on the canvas to **deselect** your selection.

To **move** a selection, click and drag.
To **resize** a selection, drag the surounding handles.
To **rotate** a selection, drag the single handle.

To **delete** a selection, click the "delete selected item" button.

The **select all** buton is particularly useful for moving all of the objects together.

Navigation
If you'd like to switch car bodies, use the tabs at the top to navigate back.

When you're ready for the next step, click the big button to advance.

Component Palette
Click tabs to view different component collections.

Click on a component to add it to the canvas.

(Although only the exterior designer is shown here, the interior designer works the same way.)

(b) Instructions for the exterior designer, displayed when the "Instructions" button is clicked

Figure 7-3: The exterior designer

Figure 7-4: A sampling of the 150+ components in DreamCar

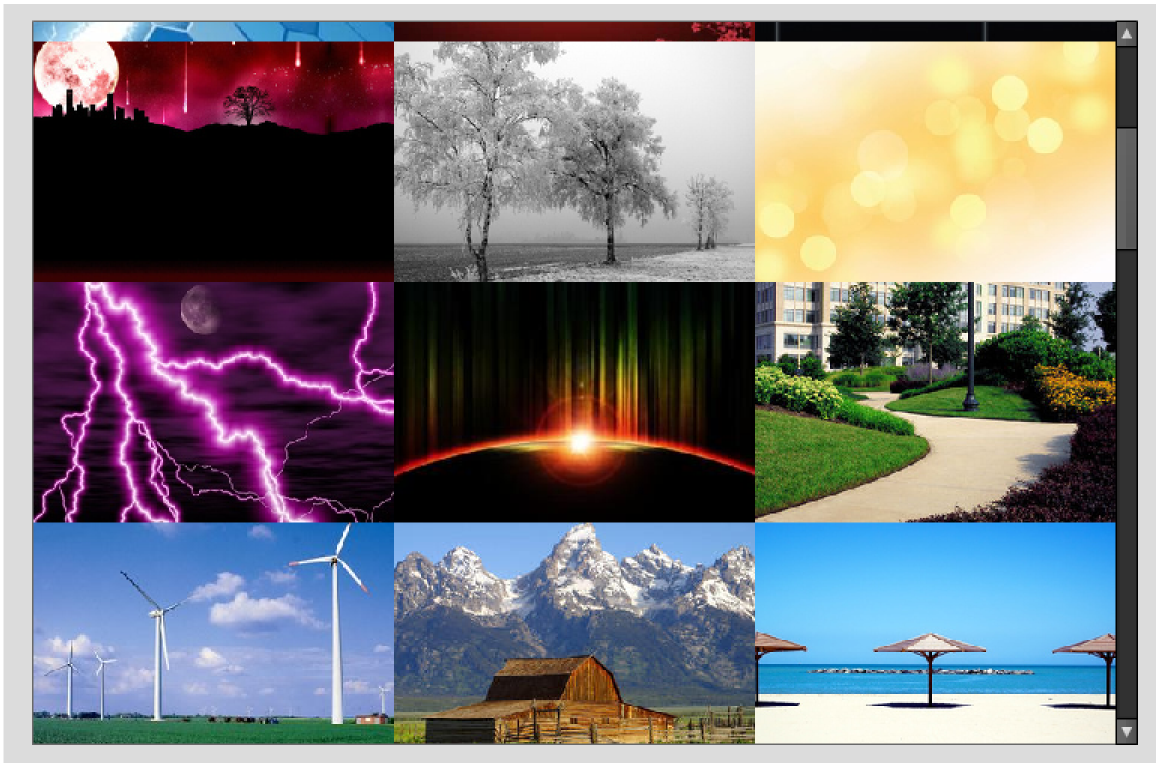
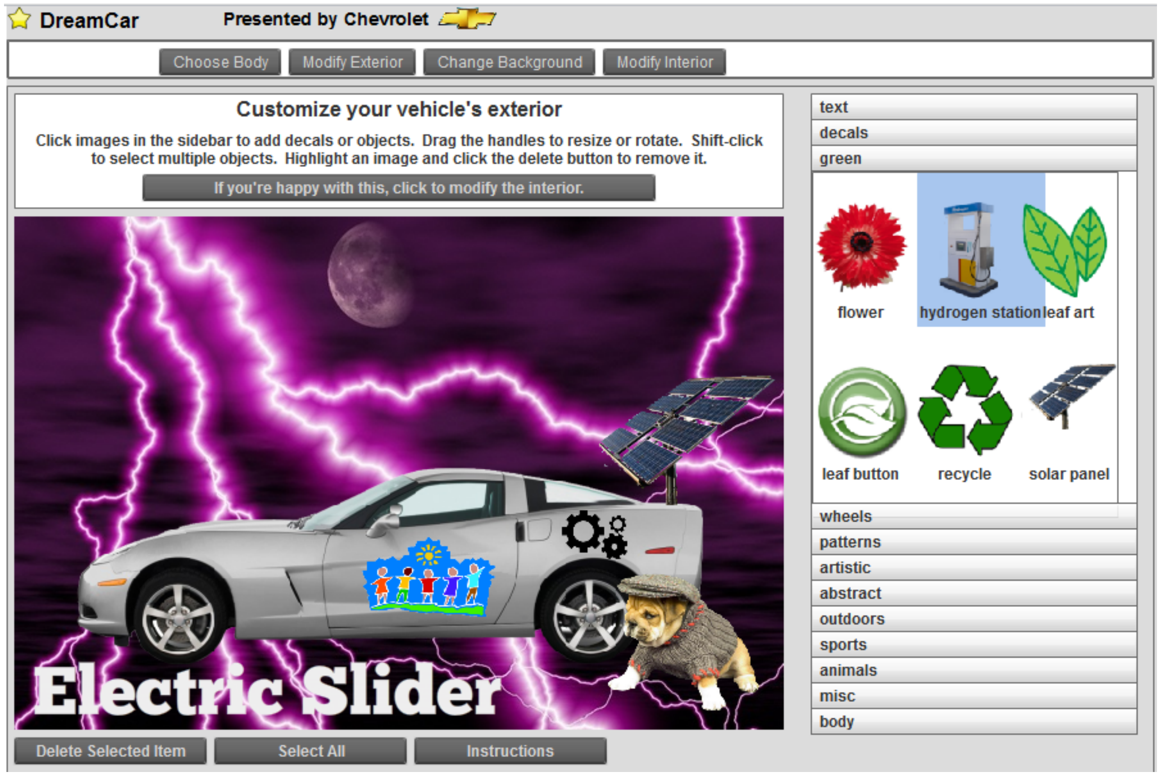


Figure 7-5: Choosing a background



Customize your vehicle's exterior

Click images in the sidebar to add decals or objects. Drag the handles to resize or rotate. Shift-click to select multiple objects. Highlight an image and click the delete button to remove it.

Figure 7-6: Back to the exterior designer to make further changes. Here, we have added text.

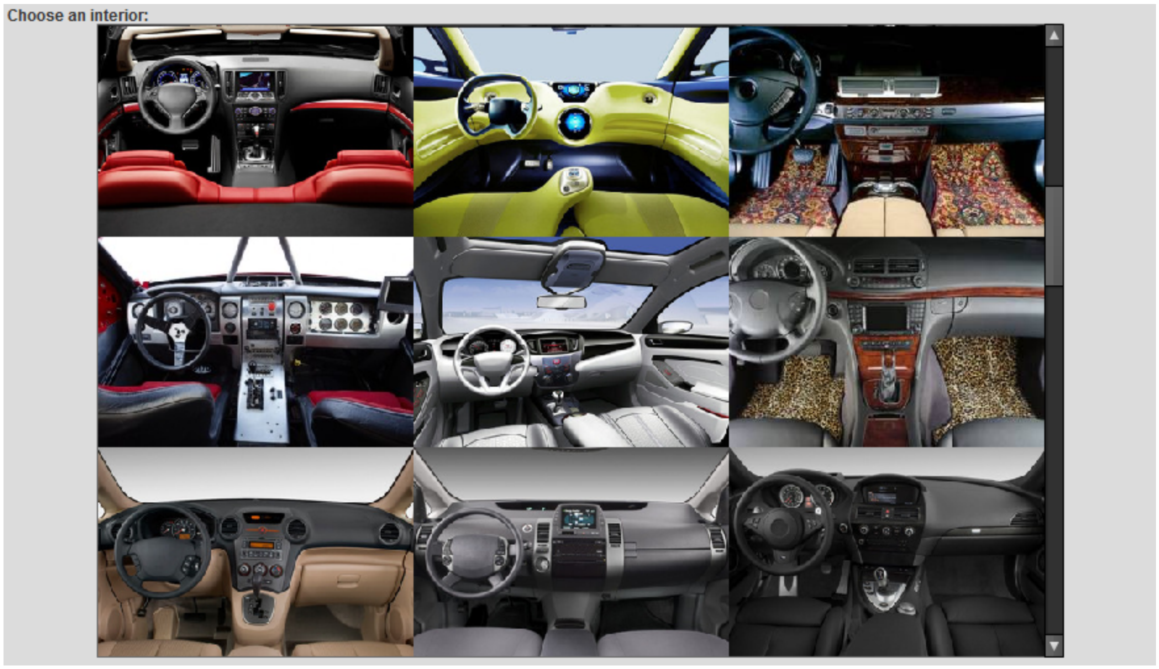
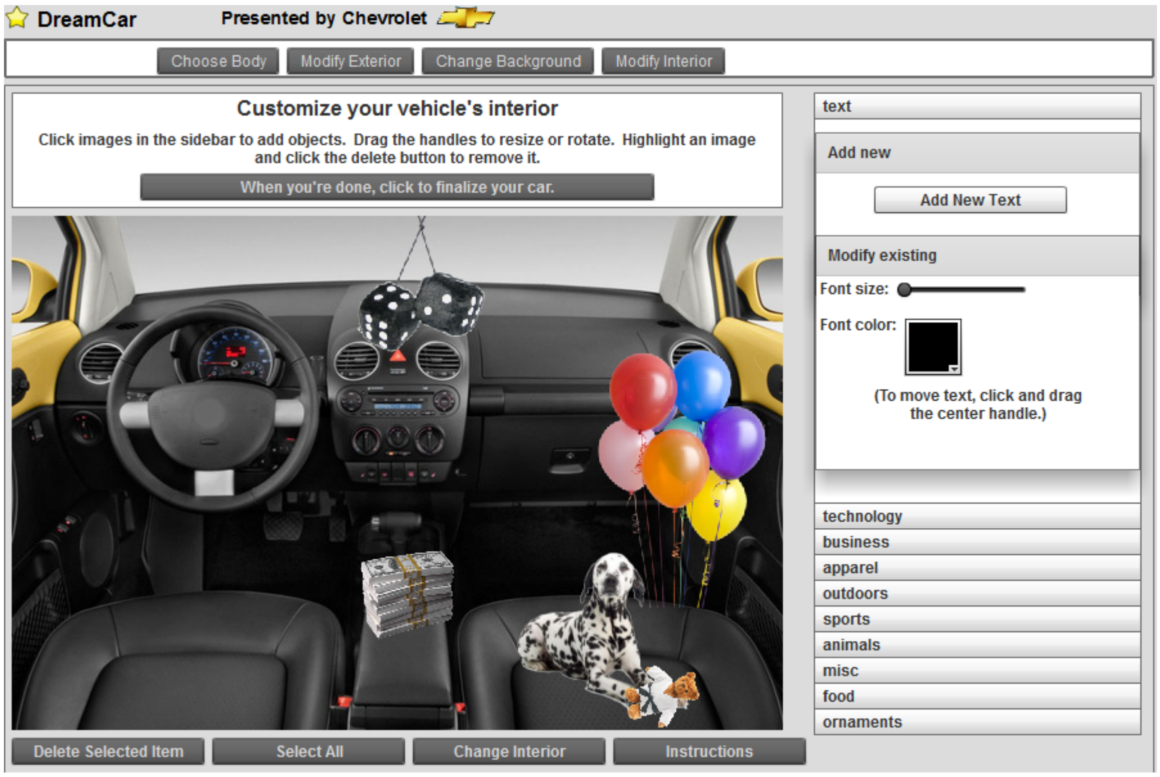


Figure 7-7: Choosing an interior



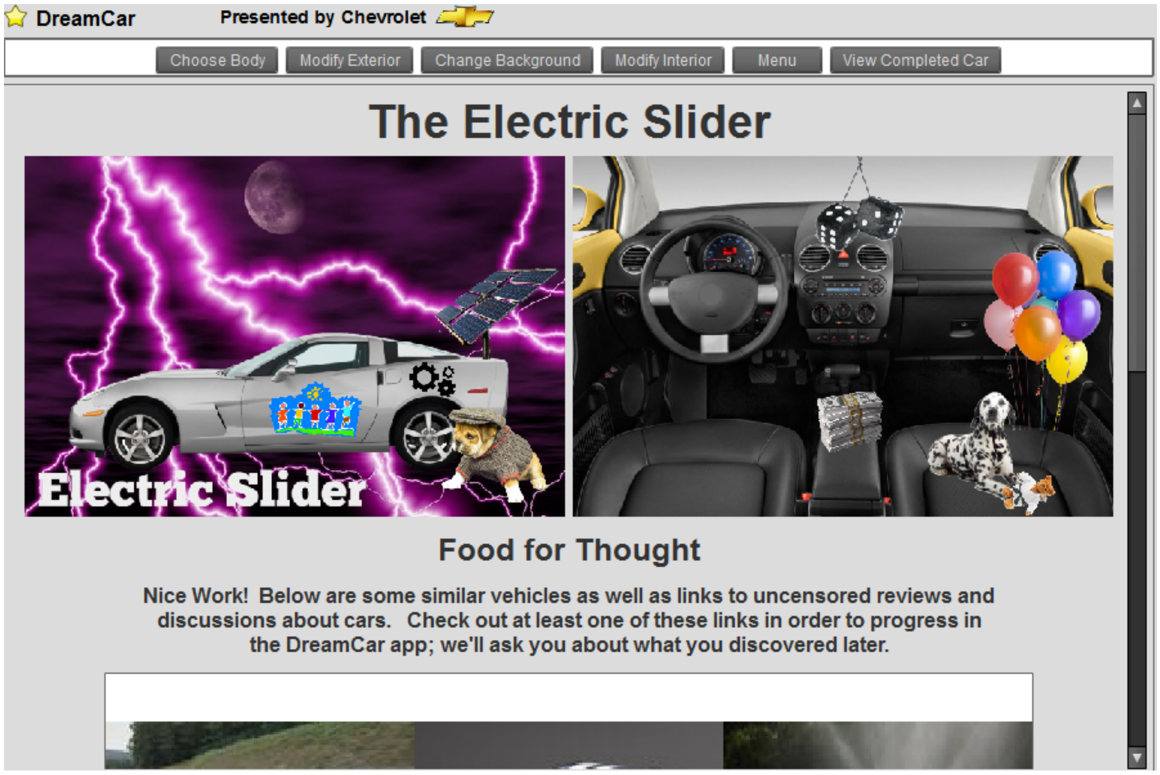
Customize your vehicle's interior

Click images in the sidebar to add objects. Drag the handles to resize or rotate. Highlight an image and click the delete button to remove it.

(To move text, click and drag the center handle.)

Figure 7-8: The interior designer

Meanwhile, the user is presented with a "finished" screen showing the completed car along with some next steps, as shown in Fig. 7-9.



The Electric Slider

Food for Thought

Nice Work! Below are some similar vehicles as well as links to uncensored reviews and discussions about cars. Check out at least one of these links in order to progress in the DreamCar app; we'll ask you about what you discovered later.

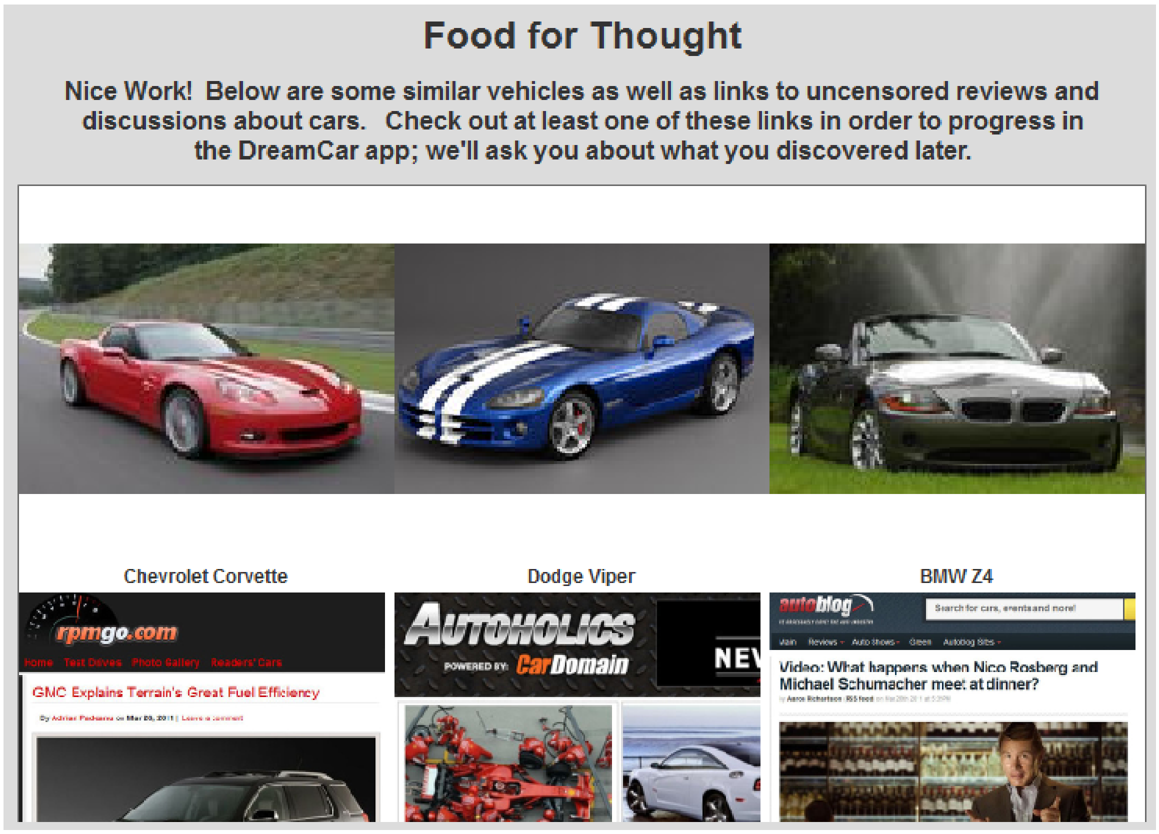
Figure 7-9: The completed car view

In the Hi-D condition, the user is then offered a set of links related to the car, as shown in Fig. 7-10. This includes three related cars (chosen based on the base model of the user's dream car) and three automobile-related blogs (randomly selected from a set of about twenty auto blogs). In order to progress, the user must click at least one link.

In the Lo-D condition, the user is shown three related cars, but the cars do not spawn links and there is no requirement to click them before continuing.

7.1.7 What's Next?

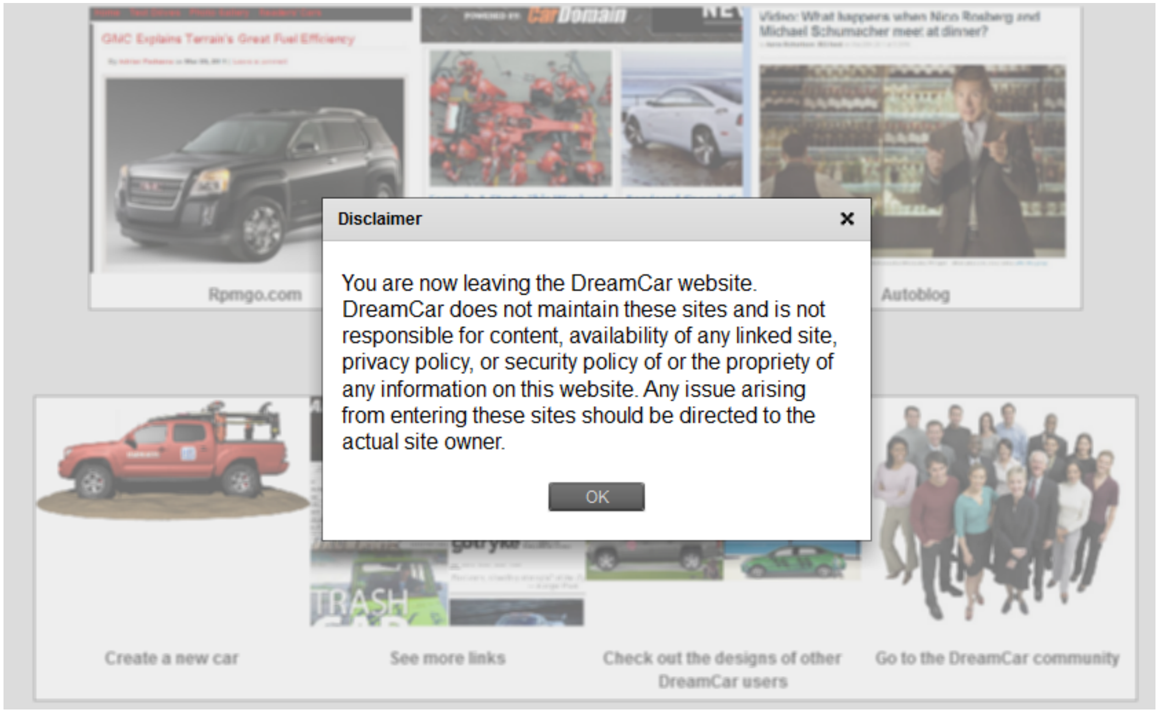
The menu at the bottom of the Completed Car view gives a few options. In all conditions, the user can create another car; in the Hi-D condition, the user can view a full list of links (three related cars plus all of the blogs); and in the Hi-E condition


Food for Thought

Nice Work! Below are some similar vehicles as well as links to uncensored reviews and discussions about cars. Check out at least one of these links in order to progress in the DreamCar app; we'll ask you about what you discovered later.

Chevrolet Corvette
Dodge Viper
BMW Z4
rpmgo.com
Home Test Drives Photo Gallery Readers' Cars
GMC Explains Terrain's Great Fuel Efficiency
Autoholics
Powered by: CarDomain
autoblog
Search for cars, events and more!
Video: What happens when Nico Rosberg and Michael Schumacher meet at dinner?

(a) Links shown on the completed car view


Disclaimer

You are now leaving the DreamCar website. DreamCar does not maintain these sites and is not responsible for content, availability of any linked site, privacy policy, or security policy of or the propriety of any information on this website. Any issue arising from entering these sites should be directed to the actual site owner.

OK
×
Rpmgo.com
Autoblog
Create a new car
See more links
Check out the designs of other DreamCar users
Go to the DreamCar community

(b) Upon clicking a link, the user is shown a disclaimer before the link is opened in a new window.

Figure 7-10: After completing a car, the user is shown links related to his creation

he can view other users' designs in the DreamCar gallery or navigate back to the forums, while the Lo-E condition only gives the option to "exit" which takes him back to the main market research page. The four options for the Hi-D Hi-E conditions are depicted in Fig. 7-11; other combinations have a similar but more limited menu.

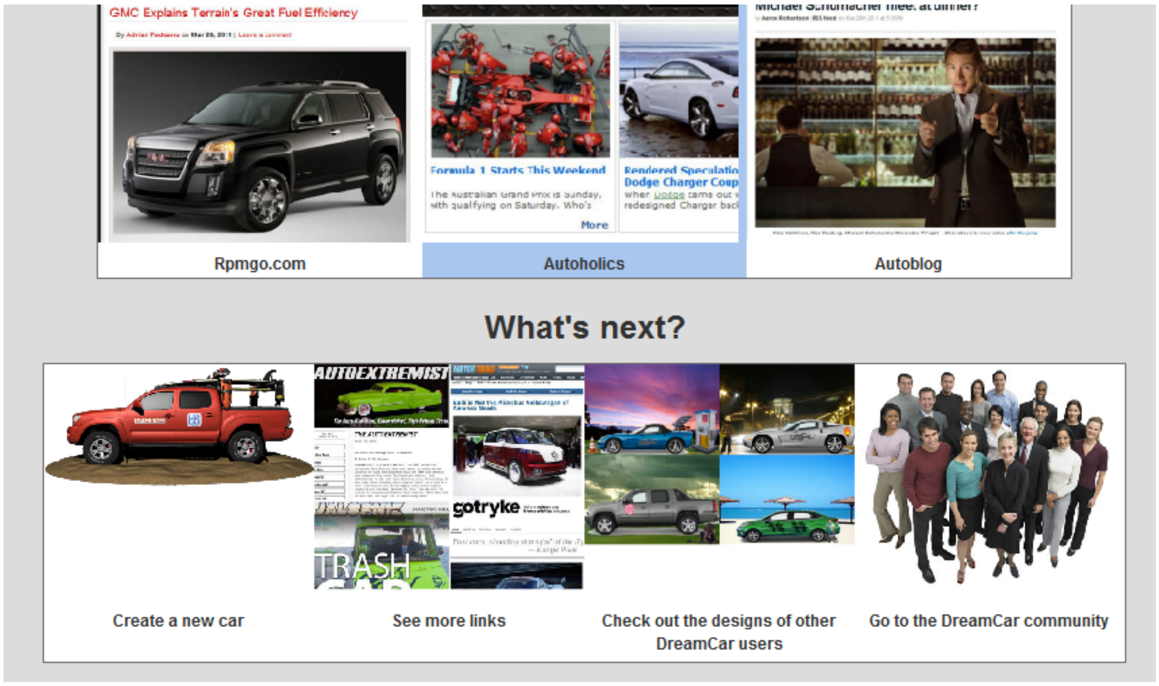
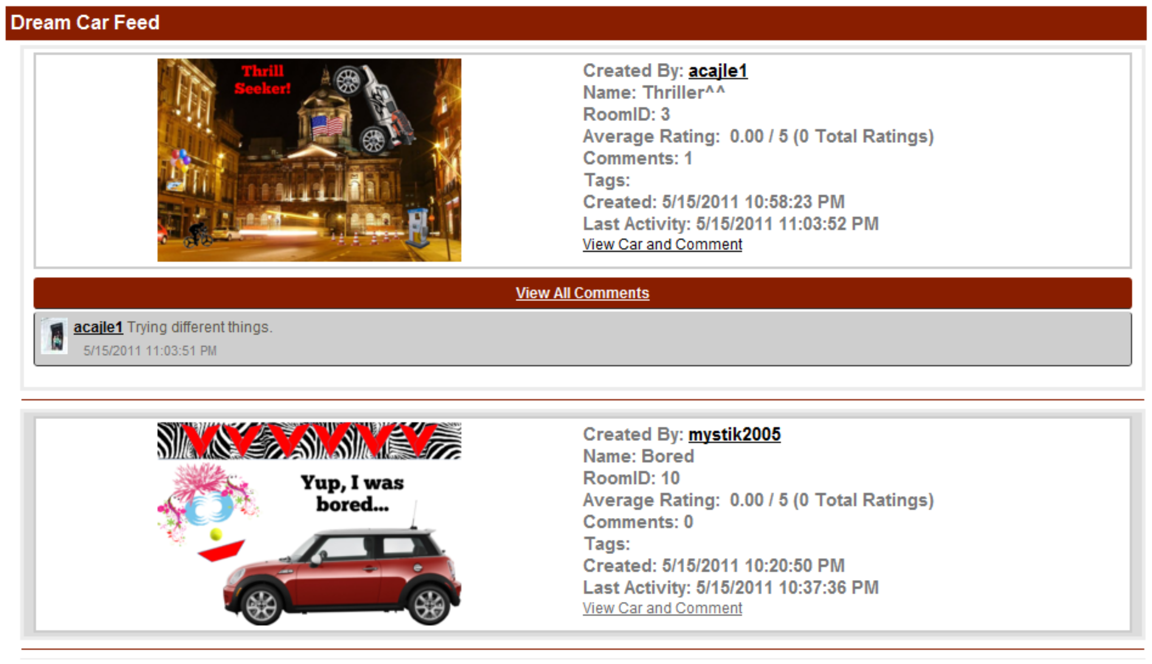

GMC Explains Terrain's Great Fuel Efficiency
Rpmgo.com
Formula 1 Starts This Weekend
Rendered Speculation Dodge Charger Coup
Autoholics
Autoblog
What's next?
Create a new car
See more links
Check out the designs of other DreamCar users
Go to the DreamCar community

Figure 7-11: What's next?

7.2 Social

For half of the participants—those in the Lo-E cells—this marks the end of the experience. To achieve parity with the Hi-E cells, the same proportion of users win the weekend rental prize, and we ask them to share their story of their experience; but the stories are not posted to the cell's community because there *is* no community.

For the other half of the participants, creating a car is only part of the experience. The users have roughly a week to look at other users' cars in the DreamCar gallery portion of their cell's website, depicted in Fig. 7-12. Each car has its own page, depicted in Fig. 7-13; there they can comment on other cars and rate them on a scale of one to five stars. Additionally, they can modify a car created by another user, which loads the DreamCar designer with the user's choices as a starting point. Finally, they can adopt other cars that they like by adding them to their own "garage," which is a collection of their favorite cars.


Dream Car Feed

Created By: acajle1
Name: Thriller^^
RoomID: 3
Average Rating: 0.00 / 5 (0 Total Ratings)
Comments: 1
Tags:
Created: 5/15/2011 10:58:23 PM
Last Activity: 5/15/2011 11:03:52 PM
View Car and Comment

View All Comments

acajle1 Trying different things.
5/15/2011 11:03:51 PM

Created By: mystik2005
Name: Bored
RoomID: 10
Average Rating: 0.00 / 5 (0 Total Ratings)
Comments: 0
Tags:
Created: 5/15/2011 10:20:50 PM
Last Activity: 5/15/2011 10:37:36 PM
View Car and Comment

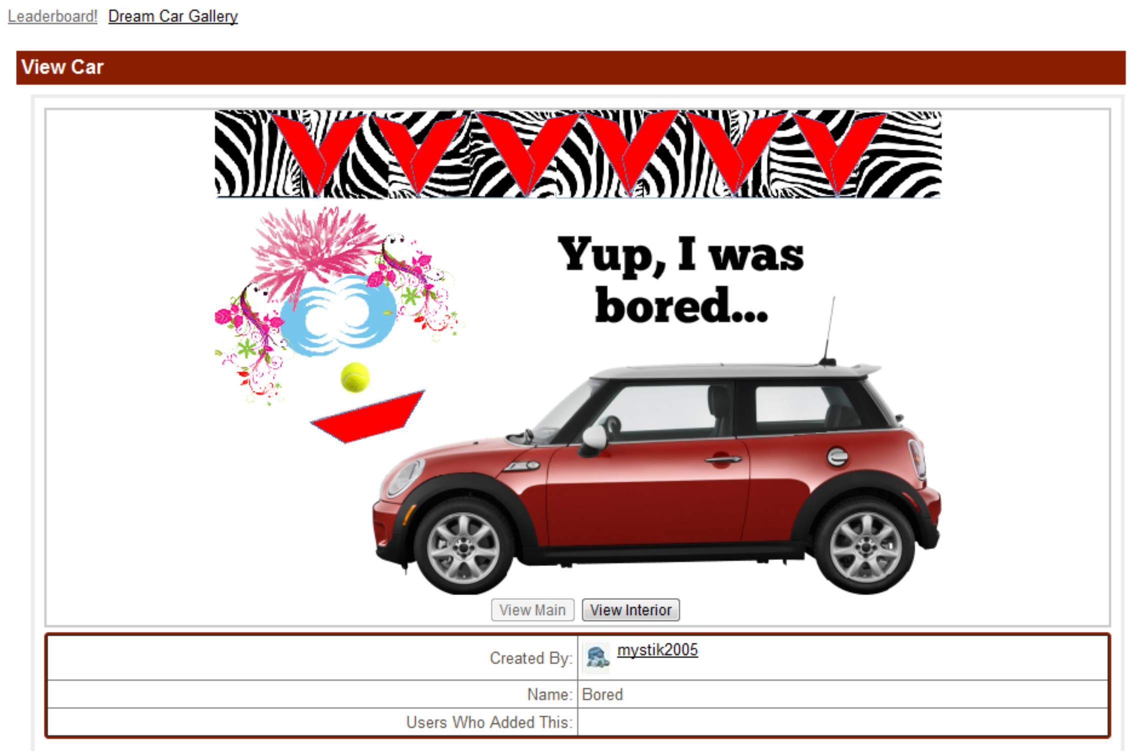
Figure 7-12: Viewing cars in the community's gallery

Users accumulate points by partaking in these social activities and by designing the best cars which generate the most activity. At the end of the social period, the users with the greatest number of points in each cell win the weekend rental. A description of the points structure is given in Fig. 7-14.

7.3 Summary of Cross-Cell Differences

7.3.1 Customizability Variation

- Stretching and themes are only available in Hi-C.
- Many more choices of background and car interior in Hi-C.
- Many more components available in Hi-C.



Created By:	mystik2005
Name:	Bored
Users Who Added This:	

Figure 7-13: Viewing a single car in the gallery

7.3.2 Discovery Variation

- After car creation, Hi-D cells get referred to both official websites for related cars, and auto blogs.
- After car creation, Lo-D cells get shown images of related cars; no links.

7.3.3 Embeddedness Variation

- Social features only available in Hi-E.
- Winners of rental are highest point earners in Hi-E; they are chosen randomly in Lo-E.
- Winners stories posted in Hi-E only.

1. You can earn 4 points per car for each car you create (up to 20 points).

   To create a car, click on the Create a Car tab on the top banner of the site. You can view others' cars by clicking the View Gallery link, and you can create your own car by clicking the Create your own link.

   The concept app will provide you with design tools to customize your dream car.

2. You can earn points when other members rate the cars that you have created.

   For each 3-star rating, you get 1 point. For each 4-star rating, you get 2 points and for each 5-star rating, you get 3 points.

   To rate another member's car, highlight the number of stars (out of 5) that you would like to give that car. The stars can be found next to the car in the gallery.

3. You can earn 10 points every time someone adds your car to their garage.

   Your "garage" is the area on your profile where you store your favorite cars, both those created by you and those created by others.

   The cars you create will automatically appear in your garage. To add another member's car to your garage, click on the Share link next to the car in the gallery. Make sure the Post to my wall box is checked, and then click Share.

   If you would like to share the car with other members of the community as well, add their names into the Share with box before you click Share.

4. You can earn 2 points every time someone comments on one of your cars.

5. You can earn 2 points every time you comment on someone elses car (up to 20 points).

   To comment on another member's car, click on the View Car and Comment link next to the car in the gallery. Type your comment in the comment box and click Add Comment.

Figure 7-14: Rules for earning points in Hi-E cells

Chapter 8

DreamCar: Architecture and Implementation

8.1 Overview

8.1.1 Platform Selection

DreamCar is built using a combination of Ruby on Rails and Adobe Flex.

Backend A Rails application provides the backend, managing the data and images shown in the application and receiving car creations when they are completed. Rails was chosen because it handles many of the messy details of web programming, such as routing and interfacing with a database. Convention over configuration and scripts for code generation make it easy to add new data structures, and adherence to the Model-View-Controller (MVC) philosophy forces the developer to keep code clearly separated. A large collection of plugins developed by the massive open-source community help reduce the need to write code for standard functionality. Although there has been some discussion in developer circles about performance issues in Rails, we assessed our needs (1710 users total, certainly not all accessing the app simultaneously) and felt that the ease of development outweighed any performance concerns. As it stands, there are a number of simple optimization steps (caching, moving image processing to a `delayed_job` queue) which have not even been undertaken but which could easily boost performance by several orders of magnitude.

Frontend A flash application, built in the Adobe Flex framework, provides the frontend. Although Ruby on Rails has traditionally been used to display data in a browser using HTML, Javascript, and CSS, the Flex environment was chosen because it provides a great deal of functionality out-of-the-box, such as common user interface components and drag-and-drop functionality. Additionally, Flash binaries are cross-platform and not browser-dependent; one of the most painful parts of developing on the web is supporting a variety of browsers ranging from Internet Explorer 6 to Google Chrome.

Communication The two sides communicate with each other using XML. Although there are several ongoing projects to couple Rails and Flex using the AMF object remoting protocol used in Flash (notably, `rubyamf`, `WebORB`, and `rails3-amf`), these projects have not reached maturity and use cases are quite brittle. XML was selected due to its simplicity and versatility. Examples of XML formats used in this project are given further in this document.

8.1.2 Server Architecture

The DreamCar app runs on an Ubuntu Linux machine. Web requests are handled by Apache 2, which handles direct requests (for image files and the Flex binary, which is in SWF format) and rails requests by way of the Phusion Passenger (`http://www.modrails.com/`) framework.

Data is stored using the SQLite database engine, a simple, single-file relational database management system. As Ruby on Rails provides a wrapper over the underlying SQL database, the actual app implementation is engine-agnostic and could be converted to MySQL (perhaps for a larger-scale implementation) by changing a few lines of code.

8.2 The Flex Frontend

The Flex application is designed in a modular fashion. At the top level is `Dreamcar.mxml`, which acts as the "radio officer" of the metaphorical DreamCar ship: it contains hooks for communicating with the surrounding page (to receive data and to notify the page when the user has completed his car), and it calls utility DreamCar methods for sending car image and XML data back to the server. `DreamCar.mxml` displays one of several views in its main body, depending on the user's state: these are:

- `Wizard.mxml`: displays the sequence of menus for choosing car bodies, etc. Displays one of several views as *its* main body, many of which have self-explanatory names
  - `Cover.mxml` the frontpage of the DreamCar app
  - `ChooseCarType.mxml`, `ChooseCarModel.mxml`, `ChooseCarMorph.mxml`: each displays choices in the hierarchy of car body selection. Choices are shown in a `spark List` environment using the `TileLayout` organization, which shows list items in a grid. Grid elements are images; the user navigates to the next level by clicking a grid element. These components make HTTP requests to the DreamCar rails server to supply their data.
  - `ChooseCarStretch.mxml` displays controls to vary the size of a car along width and height dimensions. Cars are morphed using content-aware scaling, which adds or removes pixels with minimal information content (low contrast with neighbor pixels) to reduce or extend image size without apparent distortion. Images are pregenerated and fetched from the rails server.
  - `ChooseBackground.mxml` and `ChooseInterior.mxml` display image choices in a `TileLayout`-formatted `List` control, again fetching from the rails server.
- `ExteriorDesigner.mxml`: displays the exterior design interface; a subclass of `Designer.mxml` which has common code for dragging elements, responding to

key presses, etc.

- `InteriorDesigner.mxml`: displays the interior design interface; also a subclass of `Designer.mxml`
- `CompletedCar.mxml`: shows a completed car and provides high/low discovery information

8.2.1 The Design Environment

The core activity and the place where a user will spend the most time in `DreamCar` is the design environment. In it, the user can add and manipulate images and text, overlaying objects to create a unique design. I built a versatile environment for manipulating images by leveraging the excellent `ObjectHandles` project, an open-source package for which provides square handles around on-screen objects to allow dragging, resizing, and rotations.[1]

I baked the `ObjectHandles` functionality into an abstract class called `Designer.mxml` which handles most of the common design environment functionality, and subclassed it in `ExteriorDesigner.mxml` and `InteriorDesigner.mxml` to handle details specific to one or the other—for instance, the exterior designer should show a background image, while the interior designer users the interior choice as the background canvas.

`ObjectHandles` uses a model-view-controller (MVC) organization; to use it, one registers a data model which "knows" about the objects on the canvas which are to have handles drawn around them, and a controller object manages adding to the data model. In this case, the `DesignModel.as` actionscript class contains an `ArrayCollection` of components, along with a method for dumping the data to XML. `DesignModel` watches its `ArrayCollection` for changes and dispatches an event when a change is made. This event is listened for by the `Designer.mxml` component, which updates the canvas when changes are made. Meanwhile, the `Controller` has various convenience methods for altering the `DesignModel`. This architecture reflects

[1] `http://www.rogue-development.com/objectHandles.html`

a standard MVC style, where the model is the "authority" on the state of the world and the view uses listeners to continue to display current and relevant information.

Each `Designer` has a `ComponentOrganizer` which consists of an accordion list of menus, one menu for each component type. Each menu displays images and names of components. When the user clicks on a menu item, the image is added to the design canvas.

8.2.2 Image Rendering

One of the more subtle points of this implementation has to do with image rendering in tile-layout lists. Flex `ItemRenderer`s are recycled in lists by default, which means that when the user scrolls down and an `ItemRenderer` is obscured from view, it will normally be reused to display the contents of another item. The result is that by default, if images are loaded during `ItemRenderer` creation, the wrong image will be shown when the user scrolls down (in particular, the same images at the top of the scroll window will recur in random order further down); and if images are reloaded on every `ItemRenderer` update, the correct images will be displayed but Flex will reload the an image from the server every time it is obscured and then redisplayed, which causes unnecessary server load. To address this, I integrated an open-source extension of the `Image.as` class which caches copies of images in a static dictionary mapping urls to bitmaps. Thus, when the class is given an image URL to display, it checks the cache for a copy and retrieves it from the internet only if it does not have it in the cache.

8.2.3 Data Model

A car is represented by the `CarVO` class. It has a `BodyModel` instance which has information about the car's base model. Additionally, it has links to the interior and exterior `DesignModel`s mentioned above; these objects are responsible for keeping track of the components inside of them.

The `UserPreference` object keeps track of most user choices, such as car type or background, were made and what the user's ID is. This class follows the singleton pattern, so that each class doesn't have to have a link to it that is passed down upon instantiation. This makes the code easier to follow but slightly brittle in the sense that singletons are similar to global variables. This was a conscious design choice, made after carefully considering the tradeoffs.

8.2.4 MultipartDataHandler

A custom class called `MultipartDataHandler.as` is responsible for creating a POST request with two images (one of the interior, one of the exterior) and a large XML string with information about the car. The class writes data into a byte array in preparation for communication with the rails server.

8.3 The Rails Backend

8.3.1 Gem Dependencies

Ruby plugins, known as gems, are libraries that extend the core ruby functionality. DreamCar is built using the following gems, which are written in ruby and which are each licensed under the MIT License:

- `rails` (version 2.3.10)
- `warden` (0.10.7) and `devise` (1.0.8) for user management and authentication
- `factory_girl` (1.3.3) for generating starter data and test data
- `haml` (3.1.1), a more concise markup language
- `hpricot` (0.8.4) and `nokogiri`, markup language parsers, used for parsing XML generated by the flex app
- `mocha` (0.9.12), a testing framework for mocking and stubbing features
- `paperclip` (2.3.11), a file attachment library built on the unix utility `ImageMagick`
- `mechanize` (1.0.0), a library for automating interaction with websites

8.3.2 Models

A listing of models is given below:

```
keone@glurban10:/var/www/manual/amfdreamcar/app/models$ ls
  background.rb  car_type.rb            discovery_website.rb  relation.rb
  body.rb        component_instance.rb  event.rb              text_instance.rb
  car_model.rb   component.rb           image_instance.rb     theme.rb
  car_morph.rb   component_theme.rb     interior_view.rb      user.rb
  car.rb         component_type.rb      related_car.rb
```

The rails backend has two main functions. The first is to serve data to support the flex application; for instance, if a user chooses to build a truck, then the flex app needs to know which truck base models are available, as well as the locations of thumbnails of each base model. The second is to receive data about a user creation, including static images of the completed exterior and interior and information about which components were added (and where they were added, what size they were stretched to, etc.).

As such, most models fall into one of these two categories. Either they encode information what choices are available and are effectively immutable; or they encode information about what users chose, and are created every time a new car is saved.

Immutable Data

Exteriors Users have a choice of exteriors; these are organized hierarchically into `car_types`, then `car_models`, then `car_morphs`, then finally bodies (`body.rb`). Note that bodies are different stretches of `car_morphs`.

Interiors Users also have a choice of interiors; these are organized in a flat manner only as `interior_views`.

Components The components that users can add to their car are encoded as `components`, which are organized under `component_types`.

Themes Each theme is encoded as a `theme`. Each theme has one `background` associated with it, and many `components` associated with it through the `component_theme` join table object. In the original app design, when the user would select a theme, DreamCar would prepopulate his or her car with some components related to that theme; however, this functionality was seen as confusing and was removed, so now there are no entries in the `component_themes` table.

Links Each `car_model` has several `related_cars` which are joined using the `relation` object. We also have links to blogs (with screenshots); each link is encoded as a `discovery_website` object.

User-Generated Data

Cars The fundamental unit of user creation is a car. A car has many text objects and components attached to it; we represent these as `component_instances` (an abstract class with two concrete subclasses, `image_instances` for instances of components and `text_instances` for instances of text).

Events As a user progresses through the app, we record things that they do as events. (Actually, we record a stream of events as XML, and then pass the stream back all at once when the app talks to the server.) Events have timestamps and a code for what happened; this can be translated back later. Events are described in greater detail in Section 8.4; in particular, Table 8.1 gives a listing of event codes.

8.3.3 Data Loading

`rake` is a ruby utility similar to `make` in unix, which runs a variety of useful tasks. Developers can also write custom `rake` tasks; I wrote a series of `rake` tasks which load immutable data into the database from the filesystem.

Case study: loading components

The rails app needs a row in the `components` table for each component. This table has information about whether the component should appear for low-customizability cells, or whether it should be high-customizability only. It has information about whether the component can go in exteriors, or interiors, or both. It has information about which `ComponentType` the component belongs to. It also has the name of the component to be displayed below the image in the component selector. It also has a pointer to an image.

How do we get all of this data into the database? Clearly, we could manually input each element into the database by hand, but that would take a long time and we would also have to redo this process if we had to clear the database (as often happens during testing), or migrate the setup to a new computer. Also, none of this work is really under version control.

Another approach sometimes advocated by rails developers, which is certainly an improvement over doing things by hand, is to write scripts to load the data in a migration. Migrations are scripts run when the database is destroyed and recreated, which each recreate a portion of the database's structure. For example, I have a migration which creates the `components` table and specifies which columns are in it. The basic strategy here would be to specify all of the data, perhaps in a series of parallel lists, in another migration, so that when the database is reloaded the data is in there.

There are two problems here. One is that a migration can really only be run once for the lifetime of a database. So if we created some components, created the database and ran the migrations, then we would get a database full of components—but then if we decided to add more components, we would have to clobber the database, recreate it, and rerun the migrations, which would result in any cars created before being lost. If during the development process we had all of our components finalized, and that set never changed, then we this problem would probably be tolerable. But the development process lasted several months, and every so often team members (from GM or MIT) would suggest a new component to add, and dumping the whole database to add a new component is not reasonable.

The second problem is that for each additional component, we need to alter the text file fairly substantially. Ideally, we would like to just be able to put a new image somewhere, run a script, and—presto—have the new component in the database with all the appropriate metadata. Other approaches such as the standard rails method of specifying object properties in fixtures, are also subject to this problem.

The solution I came up with is to write a rake task which scans a specified folder for images, trying to find new images to be added as components. We specify the component's `ComponentType`, as well as its appropriateness in interiors and exteriors, by placing it in the appropriate folder. We specify the name to be used in the app by giving the image that name. This hierarchy is shown in Figure 8-1; in this example, we see that `balloons` and `loudspeaker` should appear as options in both the interior and exterior designers, under the `misc` category.

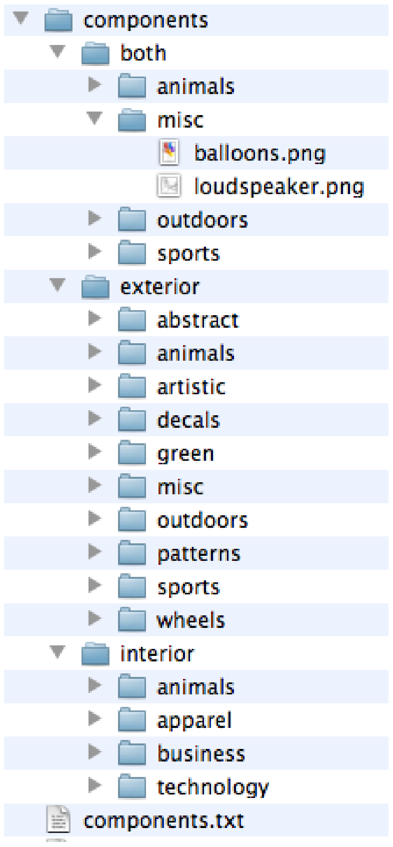
Lastly, we handle the issue of low and high customizability by scanning a file called `components.txt` which simply lists the names of all of the components we want to be in both Lo-C and Hi-C.

This makes adding new components simple and easy. All we do is put the image in the appropriate folder and run `rake db:data:load_components` at the command line. And because the files are checked into version control, any changes made to the system (new components added, components reclassified, etc.) persist.

A code snippet is given in Figure 8-2.

Loading other data

Much of the other data is loaded a similar approach—by putting images in an appropriate directory and using the `Dir.glob` ruby command to inspect the directory for children. An extreme example is loading the car bodies, which has four levels of hierarchy, as shown in Figure 8-3. Interiors, backgrounds, themes, and blogs are loaded in a similar fashion.



```
components
  both
    animals
    misc
      balloons.png
      loudspeaker.png
    outdoors
    sports
  exterior
    abstract
    animals
    artistic
    decals
    green
    misc
    outdoors
    patterns
    sports
    wheels
  interior
    animals
    apparel
    business
    technology
components.txt
```

Figure 8-1: Folder structure for importing components in DreamCar

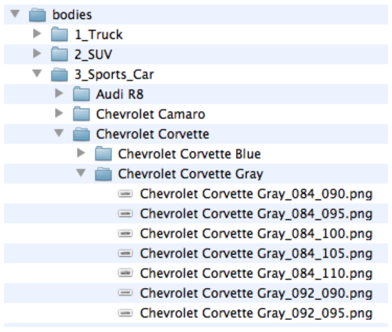
```
namespace :db do
  namespace :data do
    desc "Load components"
    task :load_components => :environment do |t|
        load_components
    end
end

def load_components
  load_components_for_type("both/*",true, true)
  load_components_for_type("exterior/*",true, false)
  load_components_for_type("interior/*",false, true)

  f = File.open("test/factories/data/components.txt", "r")
  lines = f.readlines
  lines.each_with_index do |line,i|
    c = Component.find_by_name(line.strip)
    if c
      c.high_customizability_only = false
      c.save
    end
  end
end

def load_components_for_type(path, exterior,interior)
  component_type_paths = Dir.glob("test/factories/data/components/"
      + path).sort
  component_type_paths.each do |component_type_path|
    ct_name = component_type_path.split("/")[-1]
    component_type = ComponentType.find_or_create_by_name(ct_name)
    component_paths = Dir.glob(component_type_path + "/*").sort
    component_paths.each do |component_path|
      component_name = component_path.split("/")[-1].split(".")[0]
      if(!Component.find_by_name(component_name))
        c = Component.new( :photo => get_factory_file(component_path),
                          :component_type => component_type,
                          :name => component_name,
                          :exterior => exterior, :interior => interior,
                          :high_customizability_only => true)
        c.save
      end
    end
  end
end
```

Figure 8-2: Selections from a rake script to load components



```
bodies
  1_Truck
  2_SUV
  3_Sports_Car
    Audi R8
    Chevrolet Camaro
    Chevrolet Corvette
      Chevrolet Corvette Blue
      Chevrolet Corvette Gray
        Chevrolet Corvette Gray_084_090.png
        Chevrolet Corvette Gray_084_095.png
        Chevrolet Corvette Gray_084_100.png
        Chevrolet Corvette Gray_084_105.png
        Chevrolet Corvette Gray_084_110.png
        Chevrolet Corvette Gray_092_090.png
        Chevrolet Corvette Gray_092_095.png
```

Figure 8-3: Folder structure for importing car bodies in DreamCar

Loading related cars

I mention this `rake` task because it is particularly novel. The problem can be described as follows: for each base model, we have a list of three to five related cars which should be displayed after the user finishes his car. For example, if the user selects the Ford Ranger base model off of which to build his dream car, then after he completes his car we want to show him three cars selected from the Ford Ranger, Chevrolet Colorado, GMC Canyon, and Toyota Tacoma. This relationship mapping was provided by GM.

For each related car, we need the URL of its official website and an image of it. There are about one hundred related cars, which means a few hours of repetitively searching for cars and recording a URL and saving an image. This information could be stored in a text file and dealt with the same way the other data was loaded, so at least it can persist, but it is still irritating. And as the number of related cars grows, this task would become more difficult.

The solution is to use `mechanize`, a nifty ruby gem that allows the ruby to pretend to be a browser to make web requests. In particular, we pretend to be a user searching for each car on Google, and record the URL of the top result. Additionally, we pretend to be a user searching for each car on Bing image search, we record the URL of the linked image, and then we download the image. We also write down the URLs we found in a text file so that the information will persist the next time the script gets run. A code snippet is given in Fig. 8-4.

```
def get_related_urls
  require 'rubygems'
  require 'mechanize'
  agent = Mechanize.new

  input = File.open("test/factories/data/relationships.txt", "r")
  output = File.open("test/factories/data/related_cars_w_urls.txt","a")
  lines = input.each do |line|
    get_one_car(line.strip, agent, output)
  end
end

def get_one_car(line, agent, output)
  out_line = line.strip             #### GET URL
  query = line.strip.gsub(" ","+")
  begin
    page = agent.get('http://www.google.com/search?q='+query)
  rescue Net::HTTPServiceUnavailable
    puts "google denied; waiting 5 seconds"
    sleep 5
    page = agent.get('http://www.google.com/search?q='+query)
  end

  a=page.search("h3.r a.l")
  url = a.first.attributes.first[1].to_s
  out_line += "\t" + url

  query = line.strip.gsub(" ","+")  #### GET IMAGE
  begin
    page = agent.get("http://www.bing.com/images/search?q="+query)
  rescue Net::HTTPServiceUnavailable
    puts "google denied; waiting 5 seconds"
    sleep 5
    page = agent.get("http://www.bing.com/images/search?q="+query)
  end
  a=page.search("span.sg_u").first
  url = a.attributes["metadata"].value.split(",")[2].split(
      ":")[1..-1].join(":")[1..-2]
  out_line += "\t" + url
  output.write(out_line+"\n")
  return out_line
end
```

Figure 8-4: Selections from a `rake` script to find related cars.

8.4 Event Logging

One of the key innovations in the DreamCar app is the integration of event logging into the entire app workflow. This allows researchers to trace through all of the actions each user made when interacting with the app, determining how long the user spent at each step. This level of detail can provide tremendous insights into how users are using the app, identifying which features most engaged users, which features were confusing, etc. Additionally, it allows an additional level of segmentation in the analysis: we can make statements such as "users who played with the app for more than X minutes changed their brand consideration ratings by Y%."

We implement logging on the Flex side using a singleton class, `EventLogger.as`. This writes a continuous stream of events to an `ArrayList`, which are dumped to XML when the appropriate time comes. Each event contains a timestamp, an event code (listed in Table 8.2), and an integer payload, which provides additional information about the event. For example, for the `URLChange` event, the URL code is used as the payload. URL codes are listed in Table 8.2.

Events are dumped to XML and passed back to the rails server when the user completes his or her car and saves it, and also when he clicks one of the outgoing buttons to exit DreamCar and return to the community. The event `ArrayList` is not cleared, but redundant events are caught by the rails server; this redundancy is tolerated in case the transmission was dropped the first time.

A code snippet is given in Figure 8-5.

8.5 Rails-Flex Communication

8.5.1 Loading Data

When the app needs data, it makes a GET request to the rails server. For instance, if the user clicks on car type 1 (trucks), the flex app loads the ChooseCarModel component, which makes a web request to `car_type/1.xml` on the rails server. This page returns a list of `car_models` under `car_type=1`.

```
// EventLogger.as

// event codes are omitted in this snippet as they are provided in the
// event code table; they look something like:
//   public static const URL_CHANGE:int = 1;
//   public static const EVENT_MAPPING:Array = ["n/a", "URLChange",...

private var events:ArrayList = new ArrayList();

private static var _instance:EventLogger = new EventLogger();

public static function recordEvent(eventTypeId:int,
        payload:int = -1):void{
    var currentTime:int = int((new Date()).getTime() / 1000);
    _instance.events.addItem( [currentTime, eventTypeId, payload]);
}

public static function toXML():XML{
    var xml:XML = <events/>;
    var i:int;
    for(i = 0; i < _instance.events.length; i++){
        var eventXML:XML = <event/>;
        var arr:Array = _instance.events.getItemAt(i) as Array;
        eventXML.time = arr[0];
        eventXML.eventTypeId = arr[1];
        eventXML.eventName = EVENT_MAPPING[arr[1]];
        if(arr[2] != -1){
            eventXML.payload = arr[2];
        }
        xml.appendChild(eventXML);
    }
    return xml;
}
```

Figure 8-5: Selections from `EventLogger.as`

code	name	description
1	`URLChange`	user navigated to a new page
2	`BlogOutboundClick`	user clicked a blog from `CompletedCar` and got directed there (Hi-D)
3	`RelatedCarOutboundClick`	user clicked a related car from `CompletedCar` and got directed there (Hi-D)
4	`RelatedCarNonactionClick`	user clicked a related car from `CompletedCar` but nothing happened because of Lo-D
5	`DesignerComponentAdd`	user added a component
6	`DesignerComponentRemove`	user deleted a component
7	`DesignerTextAdd`	user added text
8	`DesignerTextRemove`	user deleted text
9	`DesignerSelectAll`	user selected all
10	`DesignerShowInstructions`	user viewed instructions
11	`FollowNextSteps`	user followed one of the ‘Next Steps’ options
12	`NewCar`	user created a new car after already creating one
13	`Save`	user saved car
14	`LoadCar`	user loaded new car
15	`Exit`	user exited app; `payload = 3` goes to gallery, `4` to community, `5` to instructions

Table 8.1: Event codes recorded by the DreamCar app

code	name	description
0	`cover`	the cover page
1	`specs`	name the car
3	`car_type`	choose a car type
4	`car_model`	choose a car model
5	`car_morph`	choose a car morph
6	`car_stretch`	choose a car stretch
7	`theme`	choose a theme
8	`interior_view`	choose an interior
9	`background`	choose a background
10	`exterior`	exterior designer
11	`interior`	interior designer
12	`car_completed`	view completed car
13	`next_steps`	view next steps
14	`links`	view additional links

Table 8.2: URL codes recorded by the DreamCar app

On the rails side, these request are handled, like any other web request, by the appropriate controller delegated by `config/routes.rb`. In our example, `routes.rb` matches the rule

```
map.car_type 'car_type/:id.:format', :controller => "bodies",
:action => "car_type"
```

and passes the request to `BodiesController` (in `app/controllers/bodies_controller`), specifically to the `car_type` method, passing in `1` as the value of the `id` parameter. That method finds the `car_models` under the appropriate car_type.

```
def car_type
  @car_type = CarType.find(params[:id])
  @car_models = @car_type.car_models

  respond_to do |format|
    format.html
    format.json { render :json => @car_models.to_json }
    format.xml { render :xml }
  end
end
```

Lastly, the method recognizes that the request was of format xml, so it returns a response of the format specified in `app/views/bodies/car_type.xml.builder`. The contents of that file are given in Fig. 8-6.

This produces the XML file shown in Fig. 8-7

This same pattern repeats for all of the other data fetches.

8.5.2 Saving Data

Data is also passed from the flex app to the rails app using XML. A series of recursive calls, starting with the `CarVO` object (which calls its `DesignModels`, which call their member `ComponentModels`, assemble XML to pass back to the rails server, as shown in Fig. 8-8.

```
xml.instruct!
xml.car_models do
  @car_models.each do |car_model|
    xml.car_model(:id => car_model.id) do
      xml.id car_model.id
      xml.name car_model.name
      xml.photo_thumb car_model.photo_url_thumb
      xml.default_body do
        xml.id car_model.default_body_id
        xml.photo car_model.default_body.photo_url
        xml.pixel_width car_model.default_body.pixel_width
        xml.pixel_height car_model.default_body.pixel_height
      end
      xml.child_count car_model.car_morphs.count
      if car_model.car_morphs.count == 1
        xml.child do
          xml.id car_model.car_morphs.first.id
          xml.name car_model.car_morphs.first.name
          xml.default_body do
            xml.id car_model.car_morphs.first.default_body_id
            xml.photo
              car_model.car_morphs.first.default_body.photo_url
            xml.pixel_width
              car_model.car_morphs.first.default_body.pixel_width
            xml.pixel_height
              car_model.car_morphs.first.default_body.pixel_height
          end
        end
      end
    end
  end
end
```

Figure 8-6: Ruby code for exporting car models to XML

```
<car_models>
<car_model id="1">
<id>1</id>
<name>Chevrolet Avalanche</name>
<photo_thumb>
/system/photos/13/thumb/Chevrolet Avalanche_100_100.png
</photo_thumb>
<default_body>
<id>13</id>
<photo>
/system/photos/13/original/Chevrolet Avalanche_100_100.png
</photo>
<pixel_width>942</pixel_width>
<pixel_height>331</pixel_height>
</default_body>
<child_count>1</child_count>
<child>
<id>1</id>
<name>default morph</name>
<default_body>
<id>13</id>
<photo>
/system/photos/13/original/Chevrolet Avalanche_100_100.png
</photo>
<pixel_width>942</pixel_width>
<pixel_height>331</pixel_height>
</default_body>
</child>
</car_model>
...
(the actual xml is much longer)
...
```

Figure 8-7: XML format for retrieving children of Car Type=1

```
//in CarVO.as
public function toXML():XML{
  var myXML:XML = <car />;
  myXML.body=bodyModel.toXML();
  myXML.interiorViewId=interiorViewId;
  myXML.backgroundId=backgroundId;
  myXML.themeId=themeId;
  myXML.roomId=roomId;
  myXML.userId=userId;
  myXML.name=name;
  myXML.exterior=exteriorModel.toXML();
  myXML.interior=interiorModel.toXML();
  return myXML;
}

//in DesignModel.as
public function toXML():XML{
  var myXML:XML;
  if(exterior){
    myXML = <exterior />;
  }
  else{
    myXML = <interior />;
  }

  for(var i:int = 0; i<shapes.length; i++){
    myXML.appendChild((shapes.getItemAt(i) as RectangleModel).toXML());
  }
  return myXML;
}

// in ComponentModel.as
public override function toXML():XML{
        var myXML:XML = <component/>;
        myXML.componentId = componentId;
        myXML.componentInstanceId = componentInstanceId;
        myXML.width = width;
        myXML.height = height;
        myXML.x = x;
        myXML.y = y;
        myXML.rotation = rotation;

        return myXML;
}
```

Figure 8-8: Actionscript code for writing to XML for saving.

8.5.3 Modifying a Car

Storing all of the information about a car's construction allows us to reconstruct it in the design environment. This enables users to modify existing cars. We need code on the rails side to output a car, and code on the flex side to read it in. Figure 8-9 shows the XML format for a created car and Fig. 8-10 shows code for loading this XML into an actionscript object representing the car.

8.6 Summary

The code examples provided in this section are by no means exhaustive, but hopefully they illustrate the diversity of tasks required in the app, and the use of XML to tie functionality on the frontend and backend together. Adding a new feature in the app (for instance, adding backgrounds or themes) typically entails the following steps:

- Create data model in rails (database migration and ruby model class)
- Create controller (using a code generator) to serve web requests about this model
- Create XML view (using builder) to output data in XML
- Augment `CarVO` data structure (or other structure as apporpriate) on Flex side to store this data
- Create new menu (child of `Wizard.mxml` to display choices
- Create importer in flex to read/import incoming XML
- Create or augment data exporter in flex to include information about the user's choice in XML

```
<car id="829">
  <body>
    <id>756</id>
    <photo>
      /system/photos/756/original/09jaguar-xkr_092_090.png
    </photo>
    <x>326</x>
    <y>17</y>
    <width>273.019714775201</width>
    <height>79.1757172848084</height>
    <rotation>90.1676198483874</rotation>
  </body>
  <lifestyleTypeId>9</lifestyleTypeId>
  <interiorViewId>32</interiorViewId>
  <interiorViewName>2001 Chrysler Crossfire Concept Car
      Interior</interiorViewName>
  <interiorPhotoUrl>
    /system/photos/32/original/2001 Chrysler Crossfire
        Concept Car Interior.jpg
  </interiorPhotoUrl>
  <backgroundId>88</backgroundId>
  <backgroundName>dandelion.jpg</backgroundName>
  <backgroundPhotoUrl>/system/photos/88/full/dandelion.jpg
      </backgroundPhotoUrl>
  <user_id>3</user_id>
  <photo>/system/photos/829/medium/car.png</photo>
  <photo_thumb>/system/photos/829/thumb/car.png</photo_thumb>
  <exterior>
    <component>
      <componentId>44</componentId>
      <componentInstanceId>5077</componentInstanceId>
      <width>101.4</width>
      <height>130.0</height>
      <x>238</x>
      <y>267</y>
      <rotation>0.0</rotation>
      <photo>/system/photos/44/original/Flames.png</photo>
    </component>
    ...
  </exterior>
  <interior>
    ...
  </interior>
</car>
```

Figure 8-9: XML format for the car

```
public static function initializeFromXML(xml:XML):CarVO{
  var car:CarVO = new CarVO();
  car.id = int(xml.id);
  car.name = xml.name;
  car.bodyModel = BodyModel.fromXML(new XML(xml.body));

  if(xml.backgroundId != ""){
    car.backgroundId = int(xml.backgroundId);
    car.backgroundName = xml.backgroundName;
    car.backgroundPhotoUrl = xml.backgroundPhotoUrl;
  }

  if(xml.interiorViewId != ""){
    car.interiorViewId = int(xml.interiorViewId);
    car.interiorViewName = xml.interiorViewName;
    car.interiorPhotoUrl = xml.interiorPhotoUrl;
  }

  for each (var itm:XML in xml.exterior.component){
    car.exteriorModel.shapes.addItem(ComponentModel.fromXML(itm));
  }
  for each (itm in xml.exterior.text){
    car.exteriorModel.shapes.addItem(TextModel.fromXML(itm));
  }
  for each (itm in xml.interior.component){
    car.interiorModel.shapes.addItem(ComponentModel.fromXML(itm));
  }
  for each (itm in xml.interior.text){
    car.interiorModel.shapes.addItem(TextModel.fromXML(itm));
  }
  //Alert.show(xml.toString());
  return car;
}
```

Figure 8-10: Code to load actionscript objects from car XML

Chapter 9

DreamCar: Discussion and Evaluation

9.1 User Response

Over about two weeks, users created about 2500 cars. A few are shown in Fig. 9-1.

Many users in the Hi-E cells took advantage of the discussion feature and used it to make comments on cars they saw. A few cars with their comments are shown in Fig. 9-2.

A full analysis of the survey and app results is forthcoming, as the study is currently still running. We hope to link survey results closely with various measures of how the user interacted with the app such as level of engagement, degree of creativity demonstrated. One important measurement will be scoring car creations by degree of creativity, which will require external judges, but there are other measurements we can make by writing queries to the rails database:

- For each car:
    - Number of components used
    - Whether text was used; length of text
    - Size of car within image

  - Time spent creating the car
  - Number of comments made by others
  - Number of pageviews
  - Average rating
  - Number of times car was modified
  - Number of users that added the car to their garage

- For each user:
  - Number of cars created
  - Average rating of cars created
  - Max of all ratings of cars created
  - Aggregate time spent on all cars
  - Number of ratings made
  - Number of comments made on others' cars
  - Number of comments made on own cars
  - Number of comments made on user profiles ('wall')
  - Number of comments received on own user profile ('wall')
  - Number of friends

A few observations, which are anecdotal and which would need to be quantified with further analysis upon study completion, follow.

- Some users did not like the design features and used their car creations to express their dislike of the task and the app.
- Some users displayed immense creativity, finding unexpected uses for components such as layering components to create textures. Fig. 9-3a, Fig. 9-3b, and Fig. 9-3d give just a few examples of this, but perhaps the most striking example was Fig. **??**, which created an entire background scene—featuring a mountain and a tree—using other components.

- Many "designs" look like ads, with words not just on cars—the original intended use case—but backgrounds. This unexpected result shows creativity and reveals more about the users and how they view their auto aspirations. A few examples of this are shown in Fig. 9-3e and Fig. 9-3f.

- Some users made the car fairly small in the image and placed more emphasis on the creations; in extreme cases, the car was completely obscured from view. Fig.9-3d provides a nice example of this.

- Over time, in the high-embeddedness cells, the designs became increasingly extreme, as users saw what their peers were doing and tried to imitate or outdo them.

9.2 App Evaluation

Overall, the response to the app during testing was overwhelming. Users came up with ways to use the various components that were quite unexpected, and some of the creations were very impressive.

The purpose of the app was to provide a framework for testing hypotheses about trust-based app-based marketing to Gen Y, and particularly to test the effect of three dimensions (embeddedness, discovery, and customizability) on app efficacy. There is a strong level of variation in the app experience across the eight test cells, and after analyzing the data we will find out how much this translated into differences in user responses. Regardless, for market implementation, one would need more options and more design features before going live. An incomplete list of changes follows:

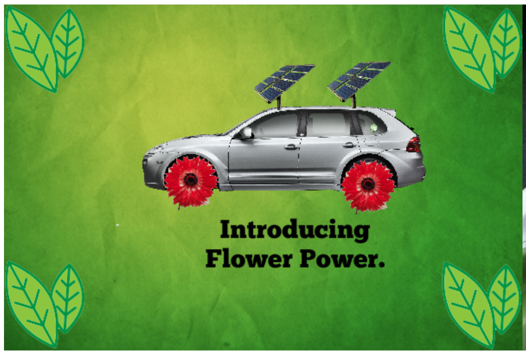
- More realistic shape morphing, such as an interface to drag the front or roof to continuously stretch the car

- Ability to change the car's color. I built a feature for recoloring the images using the `flash.geom.ColorTransform` tool which allows an image to be recolored, but I ran into the problem of non-colored elements (windshields, wheels, etc.)

being recolored in the process. I developed a solution to this problem involving splitting the car images into two parts (colored parts and non-colored parts) using a mask, but we elected to skip this step for the experiment as it would have required manually creating masks for each car body.

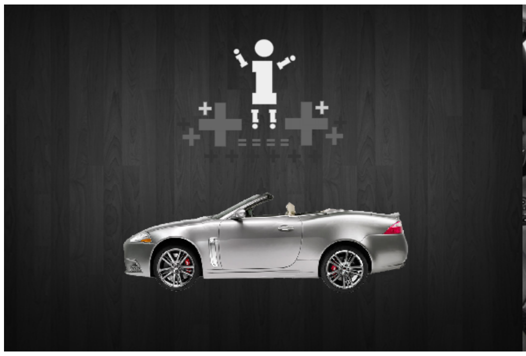
- More options for components, including more car-related components such as mufflers, headlights, trim, mud flaps, etc.
- Ability to choose more car specs, such as engine size, number of seats, seat material, etc.
- More car model options.

The existing implementation provides an excellent framework from which one could improve the app in any of these, or in any of a number of other, different directions.



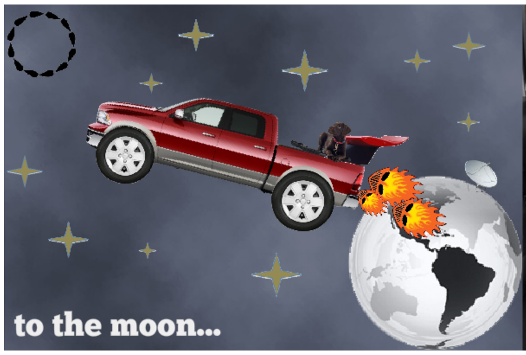
(a) "Flower Power" exterior

(b) interior

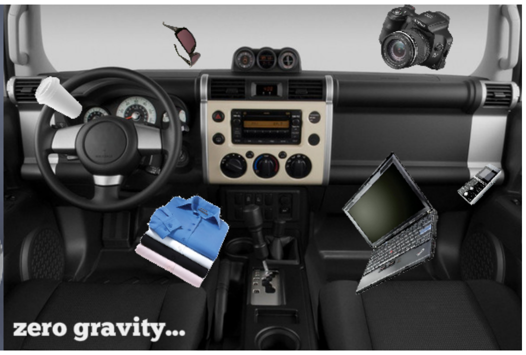


(c) "i" exterior

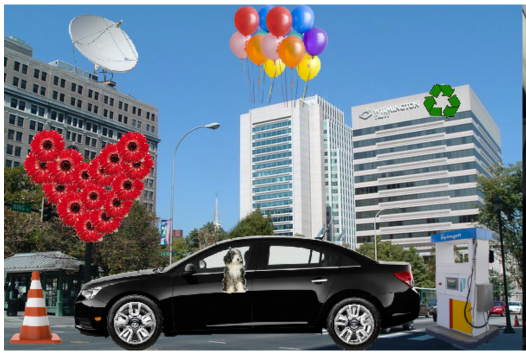
(d) interior



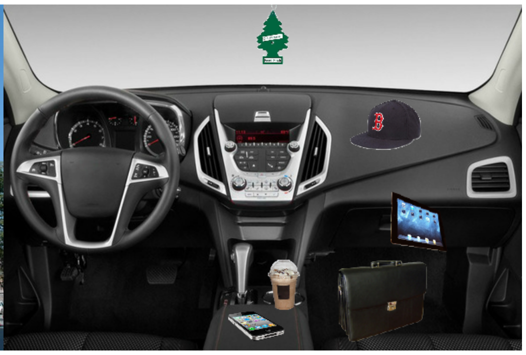
(e) "to the moon!" exterior



(f) interior



(g) "Evan's ride" exterior



(h) interior

Figure 9-1: A few of the many cars created by DreamCar test users.

No need for gas!

(i) "UP" exterior

(j) interior

(k) "All in One" exterior

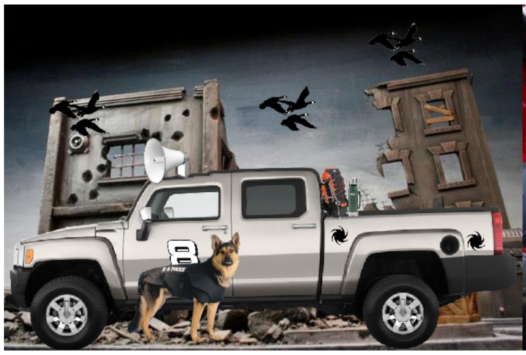
Everything I need

(l) interior

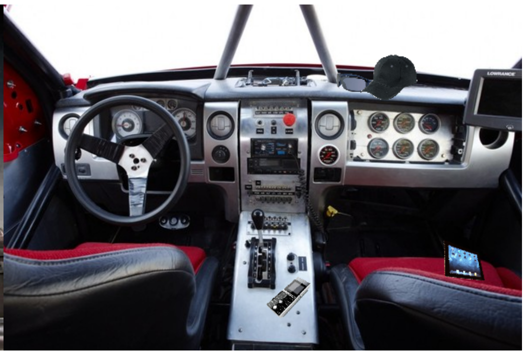
A Winter
Wonderland-- In
BLACK!

(m) "to the moon!" exterior

(n) interior

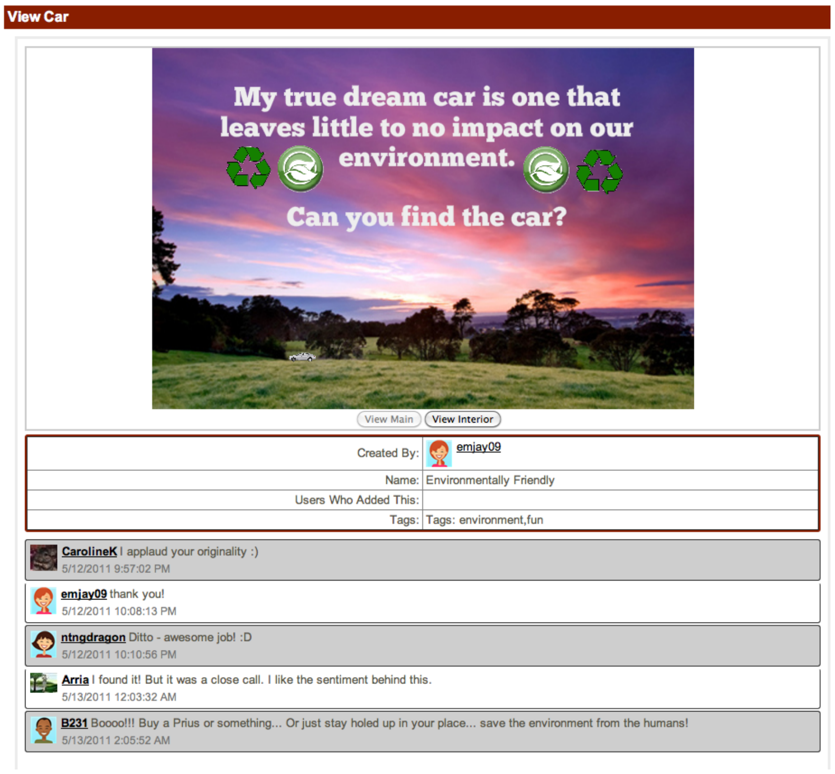


(o) "After the attack" exterior



(p) interior

Figure 9-1: A few of the many cars created by DreamCar test users (continued).


View Car
My true dream car is one that leaves little to no impact on our environment.
Can you find the car?
View Main
View Interior

Created By:	emjay09
Name:	Environmentally Friendly
Users Who Added This:	
Tags:	Tags: environment,fun

CarolineK I applaud your originality :)
5/12/2011 9:57:02 PM

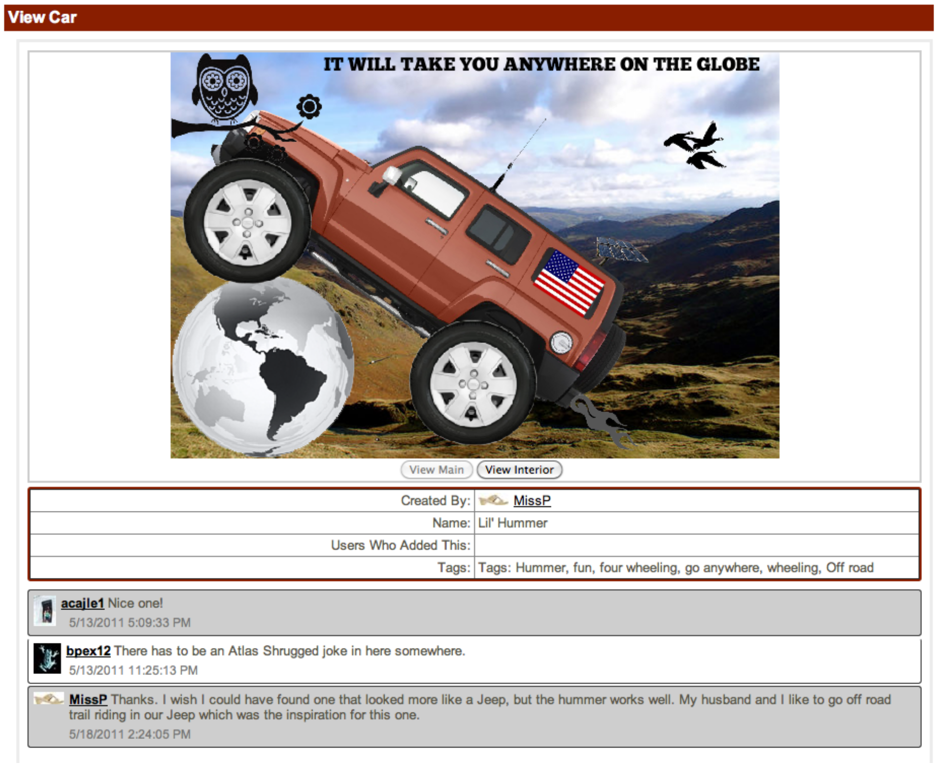
emjay09 thank you!
5/12/2011 10:08:13 PM

ntngdragon Ditto - awesome job! :D
5/12/2011 10:10:56 PM

Arria I found it! But it was a close call. I like the sentiment behind this.
5/13/2011 12:03:32 AM

B231 Boooo!!! Buy a Prius or something... Or just stay holed up in your place... save the environment from the humans!
5/13/2011 2:05:52 AM

(a) User comments on “Environmentally friendly”



Created By:	MissP
Name:	Lil' Hummer
Users Who Added This:	
Tags:	Tags: Hummer, fun, four wheeling, go anywhere, wheeling, Off road

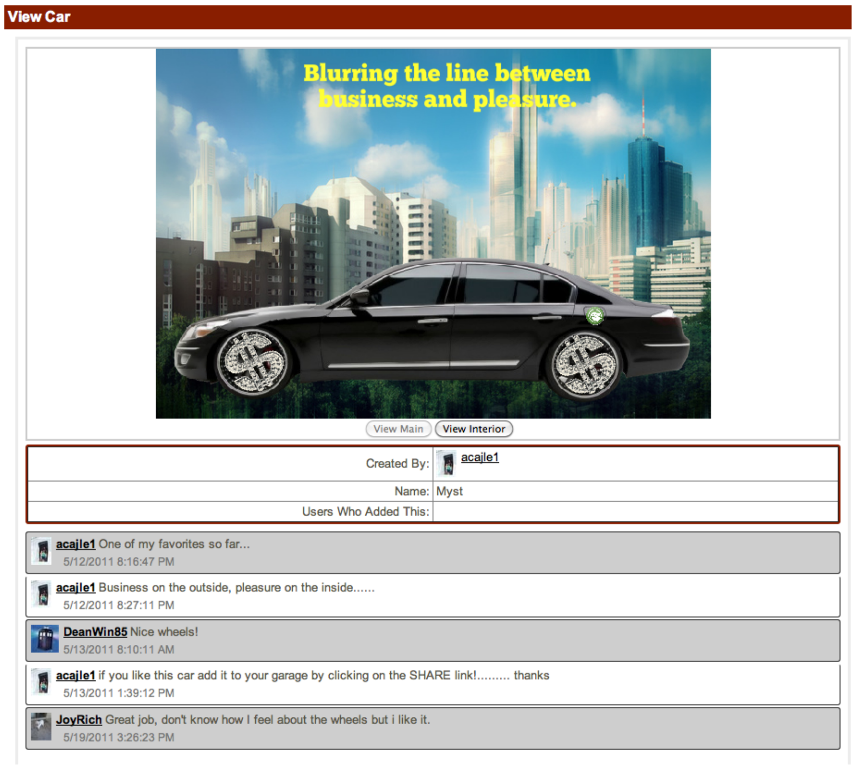
acajle1 Nice one!
5/13/2011 5:09:33 PM

bpex12 There has to be an Atlas Shrugged joke in here somewhere.
5/13/2011 11:25:13 PM

MissP Thanks. I wish I could have found one that looked more like a Jeep, but the hummer works well. My husband and I like to go off road trail riding in our Jeep which was the inspiration for this one.
5/18/2011 2:24:05 PM

(b) User comments on “Lil’ hummer”

Figure 9-2: A few cars shown in the DreamCar gallery with comments.



Created By:	acajle1
Name:	Myst
Users Who Added This:	

acajle1 One of my favorites so far...
5/12/2011 8:16:47 PM

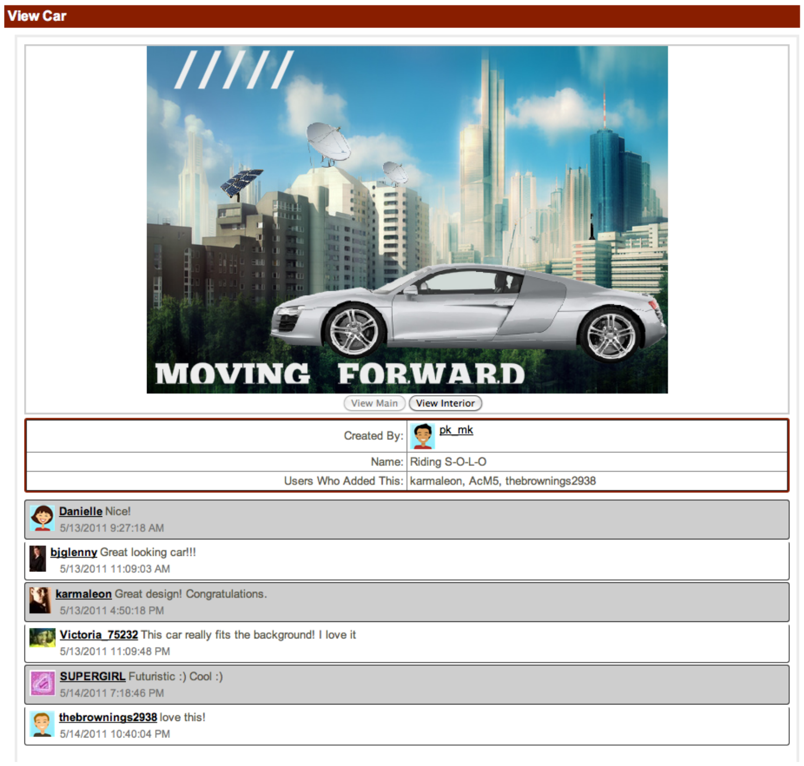
acajle1 Business on the outside, pleasure on the inside......
5/12/2011 8:27:11 PM

DeanWin85 Nice wheels!
5/13/2011 8:10:11 AM

acajle1 if you like this car add it to your garage by clicking on the SHARE link!......... thanks
5/13/2011 1:39:12 PM

JoyRich Great job, don't know how I feel about the wheels but i like it.
5/19/2011 3:26:23 PM

(c) User comments on "Myst"



Created By:	pk_mk
Name:	Riding S-O-L-O
Users Who Added This:	karmaleon, AcM5, thebrownings2938

Danielle Nice!
5/13/2011 9:27:18 AM

bjglenny Great looking car!!!
5/13/2011 11:09:03 AM

karmaleon Great design! Congratulations.
5/13/2011 4:50:18 PM

Victoria_75232 This car really fits the background! I love it
5/13/2011 11:09:48 PM

SUPERGIRL Futuristic :) Cool :)
5/14/2011 7:18:46 PM

thebrownings2938 love this!
5/14/2011 10:40:04 PM

(d) User comments on "Riding S-O-L-O"

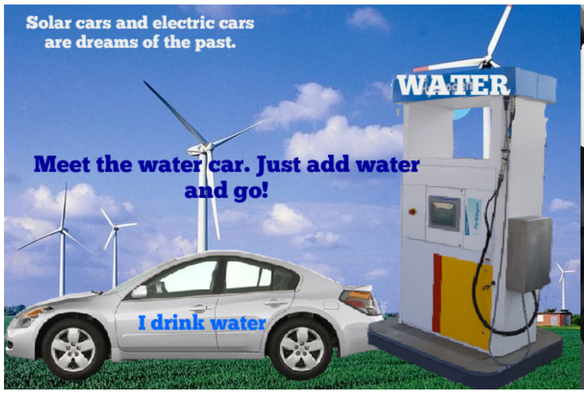
Figure 9-2: A few cars shown in the DreamCar gallery with comments (continued).

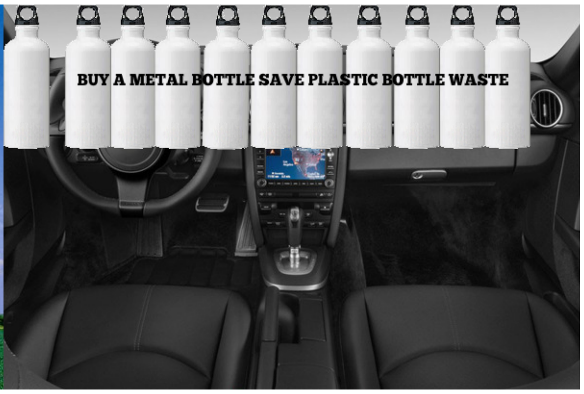
Dog of Doooooom
K9 POLICE
(a) "Dog of Doom" exterior

(b) "Sacrifice" exterior

(c) "Camper" exterior

Round 1:
FIGHT!
(d) "Fight" exterior


Solar cars and electric cars
are dreams of the past.
WATER
Meet the water car. Just add water
and go!
I drink water
(e) "WaterCar" exterior



(f) "WaterCar" interior

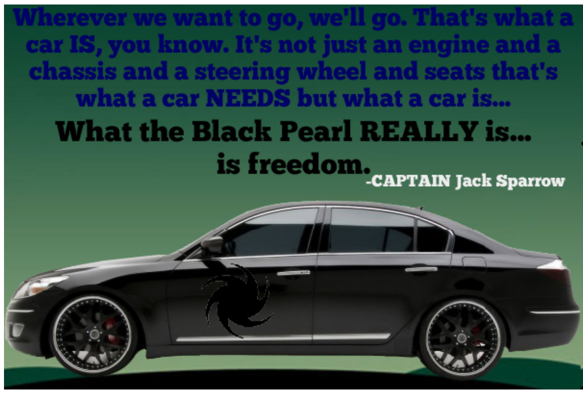
Figure 9-3: Users found many creative ways to use text and components.

The Proper Disposal Method for Cars with Flame Decals
(Stop it. You have been warned.)

(g) "Public Service Announcement" exterior

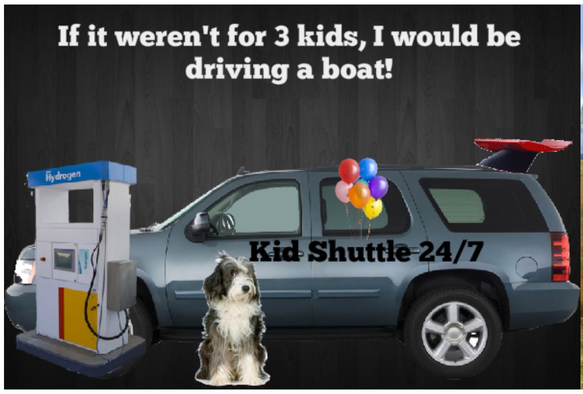
Demon Cat Continues Rampage

(h) "Apocalypse Meow" exterior

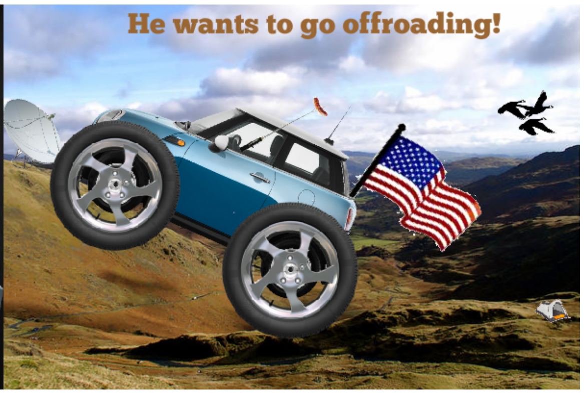

Wherever we want to go, we'll go. That's what a car IS, you know. It's not just an engine and a chassis and a steering wheel and seats that's what a car NEEDS but what a car is...
What the Black Pearl REALLY is... is freedom.
-CAPTAIN Jack Sparrow

(i) "The Black Pearl" exterior

(j) "Sky Car" exterior



(k) "Mom Car" exterior



(l) "VEXED" exterior

Figure 9-3: Users found many creative ways to use text and components.

Chapter 10

Conclusion

10.1 Academic Contributions

In this thesis, I have described the concept of trust-based marketing applications and the implementation of two such applications—one for the iPhone and one for the web via Adobe Flash.

Trust-based marketing apps provide tremendous opportunities for new marketing strategies. They provide a means of establishing trust with consumers in an interactive fashion that is simply not possible with traditional media. Apps can be seamlessly integrated into users' daily lives, and they can provide real value to their users. Traditional trust-building strategies entail showing a consumer a commercial on trust, surely of limited efficacy compared to providing an excellent app that provides a useful service.

Moreover, trust-based apps are versatile. They might be used to engage younger consumers to being building a relationship early, as in the DreamCar study; but they have also been shown to be effective in general consumer populations as well, as in the DubbleWrap study.

Analysis of the DubbleWrap study demonstrates excellent results for trust-based marketing apps, with a favorable comparison against television ads that included a 60% greater increase in consideration and a 34% greater increase in preference for the sponsoring company (compared to television ads). Analysis of the DreamCar app is forthcoming, but the results should further disambiguate the effects of different app features on efficacy.

On the engineering side, a brief comment bears mention. Lessons learned from developing engines and scripts to support these apps through logging, parsing, data extraction, and data import, as documented in this thesis, can be used in followup studies. Like all of the work at the Center for Digital Business, the story here is about how technology can be used for tremendous leverage in tackling big problems. The technology-driven approach described here has the opportunity to greatly magnify companies' efforts to build trust with their consumers—efforts that certainly qualify as a big-picture problem.

Overall, these demonstrations represent a large step forward for trust-based app-based marketing, and future work is immediately in order to begin refining the conclusions of this thesis.

10.2 Future Work

First and foremost, the ongoing DreamCar study described within should help address the question of how different dimensions of app design (degrees of social network embeddedness, customizability, and information discovery) affect the efficacy of a trust-based app.

Beyond this, we can extend this approach in several directions. Future work is needed to identify the cost of driving consumers to a free mobile app; with over 350,000 apps in the iPhone app store, for example, the app may not gain the initial momentum needed to put it in the top downloads list without some advertising dollars.

Additionally, a trust-based app should be implemented in the field with sales and clickthrough metrics used to gauge efficacy. This work will feature its own set of challenges—measuring marketing efficacy outside of market research is notoriously difficult—but many of the strategies here will carry over. The logical next step for the DreamCar app is releasing it as a Facebook application with the improvements discussed in Section 9.2, and sponsoring a similar contest for users.

More generally, the methodology used in this thesis—pre/post, test/control, differences in differences—can be repeated in other studies, for other companies or using other app designs, to target more specific questions. What kinds of apps are most effective at building trust? Is there a difference in consumer behavior from platform to platform? Does frequency of usage affect brand measures?

10.3 Prospective Meta-Analysis

This last point bears elaboration. One of the biggest challenges in marketing science today is the fragmentation of methodologies which limit generalization across studies. The lack of a single standard for analyzing problems in the space of old vs. new media or new product development mean that every study has a slightly different design, with slightly different controls and slightly different test procedures.

To illustrate the problem better, consider the number of choices made even in the simpler of the two experiments described in this thesis, the DubbleWrap experiment. Here are just a few:

- How many TV ads to have TV controls watch.
- How to enforce website engagement: i.e., how to make sure users visiting `youcovered.com` had actually looked at the website.
- Which brand measures to use, and how to phrase those questions. For example, should the agreement scales be 1-5 or 1-10? How should we measure preferences?
- Whether users who do not have an iPhone can be assigned to a non-iPhone control group.
- And so on...

An important development in marketing science would be the development of a **Prospective Meta-Analysis** (PMA) group for establishing standardized protocols and aggregating study data for large-scale studies across multiple stimuli, companies, and countries. This pattern is evidenced in the field of medicine, where the Cochrane

Collaboration has organized a PMA methods group for evaluating protocols and developing meta-analyses from standardized protocols.[1]

In marketing, this work could be approached starting in a few research specialties. As such, the DubbleWrap and DreamCar experiments are well-suited as precursors to a larger study of old versus new media. Moreover, the environment in which DubbleWrap and DreamCar were developed—the MIT Center for Digital Business, chaired by Prof. Urban—is well-suited to developing this initiative. I expect that in the coming years this will be a focal initiative for the Center.

[1] `http://pma.cochrane.org/`

Appendix A

DubbleWrap Study Surveys

Here are selections from the surveys given in the DubbleWrap study.

A.1 Pre-Stimulus Survey

1. How familiar are you with each of the following insurance companies?

   ◦ 1 Have never heard of the company
   ◦ 2 Have just heard the name
   ◦ 3 Somewhat familiar
   ◦ 4 Very familiar

   - Allstate
   - Farmers
   - GEICO
   - Liberty Mutual
   - Nationwide
   - Progressive
   - State Farm

2. Please indicate how likely you would be to consider each of the following insurance companies if you or your household had a need for homeowners/renters insurance. You can base this rating on anything you know, or have heard, about each company. (1 = Not at all likely to consider, 10 = Extremely likely to consider)

- Allstate
- Farmers
- GEICO
- Liberty Mutual
- Nationwide
- Progressive
- State Farm

3. Now we would like to understand your opinion of one of the insurance companies you selected in a previous question. Some statements that may or may not describe the company will be presented. Please read each statement and using the scale below as a guide, tell us if you agree or disagree with it. (1 = Strongly Disagree, 5 = Strongly Agree)

   - Overall, I trust Liberty Mutual
   - Liberty Mutual offers insurance coverage at a better price
   - Liberty Mutual offers more flexible insurance policies
   - Liberty Mutual offers better customer service in case of an insurance claim
   - Liberty Mutual is open, honest, and transparent
   - Liberty Mutual works hard to meet my changing needs
   - Liberty Mutual is willing to assist and support me
   - I would recommend Liberty Mutual to a friend
   - Liberty Mutual appears to be more responsible than other insurance companies
   - Liberty Mutual is an insurance company that will deliver on promises made
   - Liberty Mutual provides believable information
   - I have confidence in the information Liberty Mutual provides

4. Now we would like to understand your attitude towards protection and risk. Please read each statement and tell us if you agree or disagree with it. (1 = Strongly Disagree, 5 = Strongly Agree)

   - I try to deal with insurance-related topics as rarely as possible
   - I plan most of my activities ahead of time to avoid unexpected events
   - I engage only in safe sports where I have control
   - I follow activities which my friends and colleagues would rate as responsible
   - When moving it is important to me to be secured against all kind of risks
   - I am afraid my belongings might get lost or damaged when moving

- It is important to me that all of my belongings are covered by my homeowners / renters insurance
- It is important to me that my most valuable belongings are covered by my homeowners / renters insurance

5. Please read each statement below and indicate how likely you are to do that the next time you have a need for insurance. (1 = Very Unlikely, 5 = Very Likely)

    - Get a quote for homeowners/renters insurance from Liberty Mutual
    - Get a quote for auto insurance from Liberty Mutual
    - Get information about additional insurance products from Liberty Mutual

6. Which of the following companies would you prefer the next time you had a need for home owners/renters insurance?

    Please assign 100 points across the companies listed to indicate your preferences. You may assign all your points to one company or allocate them across any or all of them. The more points you assign to a company, the more likely you would be to consider purchasing homeowners/renters insurance from them.

    - Allstate
    - Farmers
    - GEICO
    - Liberty Mutual
    - Nationwide
    - Progressive
    - State Farm
    - Other

A.2 Mobile App Group

A.2.1 Pre-App Download

Congratulations! Youve been selected to test the new iPhone app!

To continue with the survey, you should have your iPhone in front of you so you can complete the test. If you dont have your iPhone with you now or cant download the app now, please close the survey until you have your iPhone available and can download the app.

If you are able to download the app now, click on Continue.

Otherwise, dont click continue, but instead close the survey. To continue with the survey later, when you are able to download the app, click on the original survey link in the email invitation you received and it will bring you back to this page.

Today youll test the DubbleWrap iPhone app.

Go to the App Store on your iPhone and use the Search tab to search for "dubble-wrap". Select the app called "Dubble-Wrap"; it should say it is published by Keone Hon and it should be free. Tap the blue "FREE" button, then tap the green "INSTALL" button that appears, to install the application on your phone.

Once the app is installed, go ahead and launch it; after the loading screen disappears, you'll be presented with three options: "Liberty Safe", "Box X-Ray", and "Settings." Start by tapping on "Settings" and then, on the screen it takes you to, enter your survey code, given below, and press "Done".

Survey code: [SURVEY CODE]

Please enter below the survey code you received after completing the test of the iPhone app.

A.2.2 Post App Usage

1. Based on your experience with the DubbleWrap app, please read each statement below and tell us how strongly you agree or disagree with it. (1 = Strongly Disagree, 5 = Strongly Agree)
   - The application is informative
   - The application is meaningful
   - The application is likeable
   - The application is believable
   - The application is relevant
2. How has the iPhone DubbleWrap app affected your impression of Liberty Mutual, if at all? (1 = Not as good as before, 5 = Better than before)
3. How has the iPhone DubbleWrap app affected your likelihood of considering Liberty Mutual for auto or homeowners/renters insurance, if at all? (1 = Less likely to consider, 5= More likely to consider)
4. How strongly do you agree or disagree that the iPhone DubbleWrap app portrays a company that? (1 = Strongly Disagree , 5 = Strongly Agree)
   - Follows through on its responsibilities to its customers
   - You want to be associated with
5. How strongly do you agree or disagree with these statements about the iPhone [PIPE Test Group A: DubbleWrap] [PIPE Test Group B: MoveAssist] app? (1 = Strongly Disagree, 5 = Strongly Agree)
   - The app is intuitive and easy to use

- The app is helpful in the moving process
- This is an app I would use again during my next move
- This is an app I would recommend to friends and colleagues
- The app is interesting
- The app led me to think about protecting my possessions
- I found the app to be engaging
- The app led me to consider getting homeowners/renters insurance
- The app is focused on my needs
- The app helps me to structure the process of moving
- I would uninstall the app
- The app is innovative
- The app is trustworthy
- I have concerns that my data might be used for other purposes than moving
- I have concerns that the insurance company that sponsored the app has its own interests in mind with this app
- I would mention the insurance company that sponsored this app when talking to friends and colleagues about the app

6. What, if any, positive emotions did you have when using the iPhone [PIPE Test Group A: DubbleWrap] [PIPE Test Group B: MoveAssist] app? (Please be as specific as you can. If you cant think of anything, please type N/A.)

7. What, if any, negative emotions did you have when using the iPhone [PIPE Test Group A: DubbleWrap] [PIPE Test Group B: MoveAssist] app? (Please be as specific as you can. If you cant think of anything, please type N/A.)

8. What, if any, features would you like to see in a future version of the iPhone [PIPE Test Group A: DubbleWrap] [PIPE Test Group B: MoveAssist] app? (Please be as specific as you can. If you cant think of anything, please type N/A.)

9. What would you tell a friend about the iPhone [PIPE Test Group A: DubbleWrap] [PIPE Test Group B: MoveAssist] app? (Please be as specific as you can. If you cant think of anything, please type N/A.)

10. Overall, what were the best things about the iPhone [PIPE Test Group A: DubbleWrap] [PIPE Test Group B: MoveAssist] app? (Please be as specific as you can. If you cant think of anything, please type N/A.)

11. Overall, what were the things you liked least? (Please be as specific as you can. If you cant think of anything, please type N/A.)

12. Application uninstall:

   Here are the instructions to use if youd like to remove the DubbleWrap app from your phone:

   From the home screen, identify the DubbleWrap app icon. Tap on the icon and hold it there for a few seconds. You will notice that the screen will start to wiggle and finally an X sign will appear on the application icon. Tap on the X sign and when asked whether to proceed, select Delete. The app is now removed from your phone.

Bibliography

[1] J.M. Kim et al. The zero touch experience: intent based contextual morphing on mobile devices using localized keyword distributions. 2010.

[2] G. Urban. *Don't just relate-advocate!: a blueprint for profit in the era of customer power*. Wharton School Publishing, 2005.

[3] G.L. Urban, C. Amyx, and A. Lorenzon. Online trust: state of the art, new frontiers, and research potential. *Journal of Interactive Marketing*, 23(2):179–190, 2009.

[4] K.Y. Wang. Morphing content in mobile applications. 2009.

[5] Y. Wang. An analysis of different data base structures and management systems on clickstream data collected for advocacy based marketing strategies experiments for intel and gm. 2005.

[6] M. Zhang and G.L. Urban. *An example of trust-based marketing and customer advocacy in e-commerce*. PhD thesis, Massachusetts Institute of Technology, 2006.