Smarty Project:

Designing and Implementing an A/B testing platform

Master's Degree in Software Engineering, Formal Methods and Information Systems

Student: **Marco Alacot Torres**

Director: **Mª Carmen Penadés**

September 2013

To my fathers, without them this work would never have been possible

Aknowledgements

I want to thank M. Carmen Penadés for their attention and the guidance she gave me from the very beginning until the end. Also to all my peerTransfer teammates for the effort they put every day.

Also I want give a special thank you to my girlfriend Alba for her support and her immense patience with me during the writing of this work.

Figures reference

Index

1. Introduction

“I know that half of my advertising doesn’t work. The problem is I don’t know which half.” – John Wanamaker

Building a product that people loves needs many time and resources. In a startup environment, time and resources are a scarce commodity, so we need to find ways to optimize the use of them.

We need to learn, not only to build fast, but also to decide what to build, and when. We can hire the best product team in the world and trust his judgement, but without the right data and tools, we will fail most of the times.

With data-driven product development, we gather data from our website, and use that data to make decisions and to experiment what, and when to build our new features, with A/B and multivariate testing we can test multiple versions of some feature and test the impact in our metrics such conversion, retention or acquisition. That way we can guide the new features of our product, backing the decisions we make with the proof of the data.

In order to guide the new features through metrics and data we need to gather the most amount of information we can about our customers and how they use our product, even we need to measure the users that still aren´t customers.

We gather all this information through Web analytics, which is the measurement, collection, analysis and reporting of internet data for purposes of understanding and optimizing web usage.

When we gather enough data about our users and the way they interact with our platform, we are able to start experimenting with new features and see the impact of that features in our metrics, if we are able to improve some metrics, the change is good, if not is better to not do the change.

There are many ways to do experiments, but in this document I'm going to focus the attention on A/B testing and multivariate testing, both are very powerful techniques for use experiments.

All this ingredients mixed conform data-driven development process, a very effective way to drive a product to a successful company.

The present work is in the context of the **Smarty Project,** which is a project focused on designing and developing an A/B testing platform and the main reason for this work. This project was done in my company, which is **peerTransfer.**

1.1. peerTransfer, who we are? what we do?

Founded in 2009, peerTransfer provides an innovative new way for international students to transfer funds to pay for their school tuition and related fees.

Designed with the international student in mind, peerTransfer ensures a simple and secure payment solution that not only provides students with peace of mind that their payments safely and quickly post to their student accounts, but also saves them a significant amount of money over traditional banking options.

peerTransfer offers a secure, seamless, and easy-to-navigate system for students that enables them to pay in their home currency, and also saves them a significant amount of money when doing so.

By collecting funds in the students home currency, bundling transactions to secure higher volume purchases and more favorable foreign exchange rates, and eliminating unnecessary intermediary transaction fees, peerTransfer provides significant cost savings as compared to traditional banking solutions – up to thousands of dollars for a single year s tuition. In addition to the student savings, peerTransfer helps schools to reduce administrative overhead by ensuring the fast and accurate delivery and posting of funds.

And, unlike large financial institutions, peerTransfer provides international 24x7 support across multiple channels such as voice, Skype, chat or email to guide students through the process.

peerTransfer is completely free for schools, can be turned on in a matter of hours and has been proven to save overhead, by ensuring all student information is included in the transfer.

Headquartered in Boston, Mass., peerTransfer also maintains offices in Valencia, Spain. It is backed by international investors (including Spark Capital, a $1Billion Venture Capital fund).

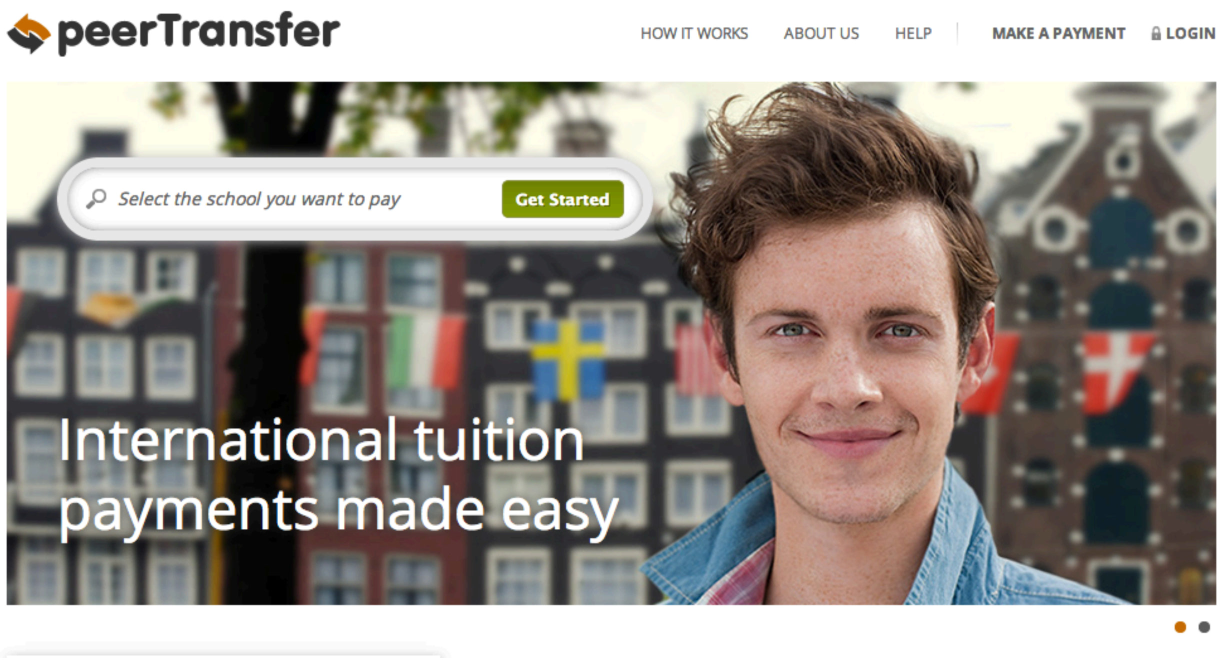

peerTransfer
HOW IT WORKS
ABOUT US
HELP
MAKE A PAYMENT
LOGIN
Select the school you want to pay
Get Started
International tuition payments made easy

Figure 1,1, peerTransfer Website

1.2. Motivation

On the early times of peerTransfer all was rush, and reducing time to market, we spent many time building many new features without thinking too much if we need it or not, and without measuring the impact of those features, we were simply building and building.

The turning point was after a very long day, when we realized that our Kanban board [1] was too much full of new stories, and most of that new stories had nothing in common between them, that was a smell that something in our process was wrong. We can see on the Figure 1.2, how is a Kanban board:

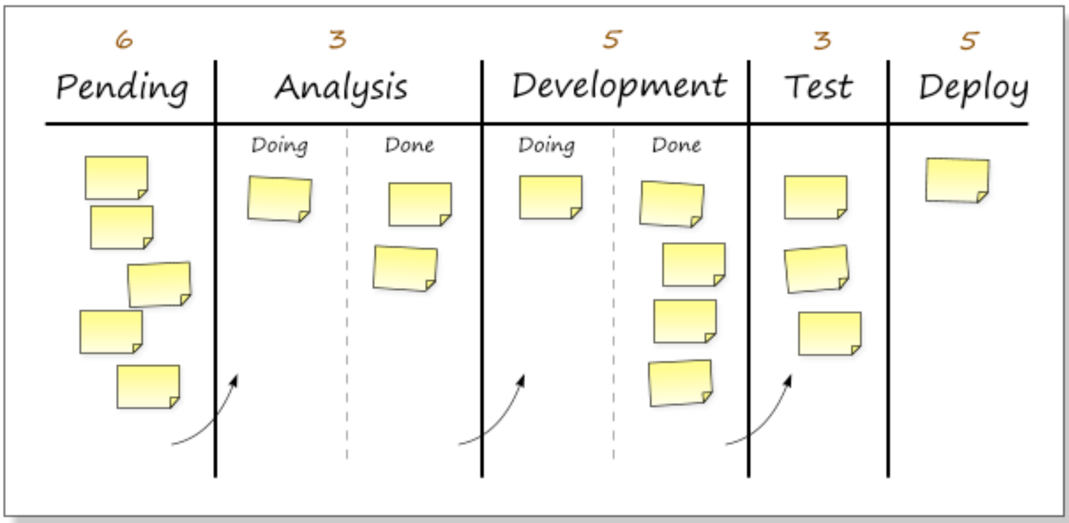


Figure 1.2, A Kanban board [KB13]

[1] Kanban "*is a method for managing the creation of products with an emphasis on continual delivery while not overburdening the development team. Like scrum, Kanban is a process designed to help teams work together more effectively*" [VO13] a Kan-Ban board "*consists of a big board on the wall with cards or sticky notes placed in columns with numbers at the top, The cards represent work items as they flow through the development process represented by the columns*." [KB13]

Hopefully we have a very good product and we succeed the first two years, but we realized that we need to make a change if we want to continue succeeding. We started thinking about it and realized that we need to make a change in our development process, we needed to add some light in our new features and to measure the impact of those.

We were not tracking our users in our site, so we hadn't got enough data about them, we didn't know what our customers wanted or what are they feelings about our product, we didn't know if our new features, and also the old ones were driving us to the success or to the failure.

In order to fix that, first we started gathering metrics about our customers, we were thinking about to use one third party solution for that, but we didn't found any that suits our needs, so we started to build a new system that gathers most of the actions that our customers make in our platform.

After we had more data and metrics that we can process, we needed a way to visualize and to analyze the data, we used Tableau for this purpose. So, in this situation, with the data, and the tools to analyze that data we started to feel the power of this way of working, so the **Smarty project** started to have more and more importance in the company.

1.3. Objectives

The main objectives of this work is to demonstrate that experiments and a data driven product development can improve the key metrics inside a web environment, and how we designed and implemented the Smarty project and the results we obtained.

We can break down this objectives on the following sub-objectives:

1. Demonstrate the importance of Web Analytics.
2. Demonstrate the effectiveness of experimentation in web environments
3. Demonstrate how a data-driven development process can improve the key metrics of a website.
4. Compare the two main experimentation methods, which are multivariate testing and A/B testing.
5. Explain the design and implementation decisions behind the different components of the Smarty project
6. Explain the architecture of the Smarty project platform and how it works.
7. Show the results we obtained with the Smarty project and the problems we had

1.4 Organization of this document

The second chapter is a brief introduction about Web Analytics, the state of the art and why we should use web analytics in our site.

The third chapter is about Data-driven product development, it includes a introduction about this development methodology and also some tips on how to implement it, on the second part of the chapter presents two testing technique which are A/B testing and multivariate testing, both are introduced and compared to each other.

The fourth chapter is about the Smarty project, which is the main motivation of writing this document. This chapter is about the requirements we had for this project but also the implementation and design decisions we took.

The fifth chapter shows the conclusions and results we accomplished, but also the problems we had and the future steps on the platform.

2. Web analytics

”Web analytics is the measurement, collection, analysis and reporting of internet data for purposes of understanding and optimizing web usage.” [Wik]

With Web analytics we can measure web traffic, but also the key metrics of our site, this is very important in order to know how our product is performing and how we can improve the the effectiveness of our site.

Web analytics are widely used on marketing campaigns, as for example it’s possible to measure the impact of a certain email campaign, and predict with this metrics how a new campaign can impact in our metrics.

Also it provides information about unique visitors, page views, and data related to the users of our site, this is very useful for personalizing the campaign or even the user experience.

There are two main categories of web analytics:

- **Off-site web analytics** refers to web measurement and analysis regardless of whether you own or maintain a website. It includes the measurement of a website's potential audience (opportunity), share of voice (visibility), and buzz (comments) that is happening on the Internet as a whole.

- **On-site web analytics** measure a visitor's behavior once on your website. This includes its drivers and conversions; for example, the degree to which different landing pages are associated with online purchases. On-site web analytics measures the performance of your website in a commercial context. This data is typically compared against key performance indicator, and used to improve a web site or marketing campaign's audience response.

2.1 Web analytics state of the art

In this section we are going to focus our attention on the state-of-the-art with respect to Web analytics, the questions we are going to address are these:

1. **What data is collected in Web analytics?**

The most typical data gathered by web analytics tools are:

Hits	Refers to each element of a Web page downloaded to a viewer's Web browser. Hits do not correspond in any direct fashion to the number of pages viewed or number of visitors to a site.
Unique visitors	The number of visitors to the site that came from a unique IP address
New - return visitors	Number of users that visit the site for the first time.
Page views	The absolute number of times that a web page has been viewed
Page views per user	The number of page vies divided by the number of visitors
IP address	IP address of the visitor
Visitor location and language	The local preferences stored in the user's browser, and also the location geo-located from the ip address.
Referring page/sites	Indicates how the user came to the site.
Browser type	Browser, version and operating system.
Visitor path / navigation	The path that the visitor follows through the website.
Bounce rate	The percentage of visitors who leave the site after the first page; calculated by the number of visitors who visit only a single page divided by the number of total visits. The bounce rate is sometimes used as another indicator of "stickiness."

2. How is it obtained?

The data is obtained essentially through event tracking systems, an event tracking system monitors all the interactions between the user and the site, such as clicks, pages viewed, etc. Any of that interactions, or even combination of multiple elements can be considered as an event, and cause of it, can be tracked.

For example we can define an event consisting of a user entering to the site and browsing to a specific category, that two events (a new user coming to the site, and after that looking to that category) can be integrated in one event in order to measure how many time that two actions happens successively. That way we can know hoy many of our new visitors browses to that category.

3. Who uses the data and for what purposes?

This depends on the organization and the type of business, but in most cases the data is used by the marketing or product teams in order to take better decisions or add new features to the product.

in our case the data is used for all the people involved on the development process, who is practically, the entire company.

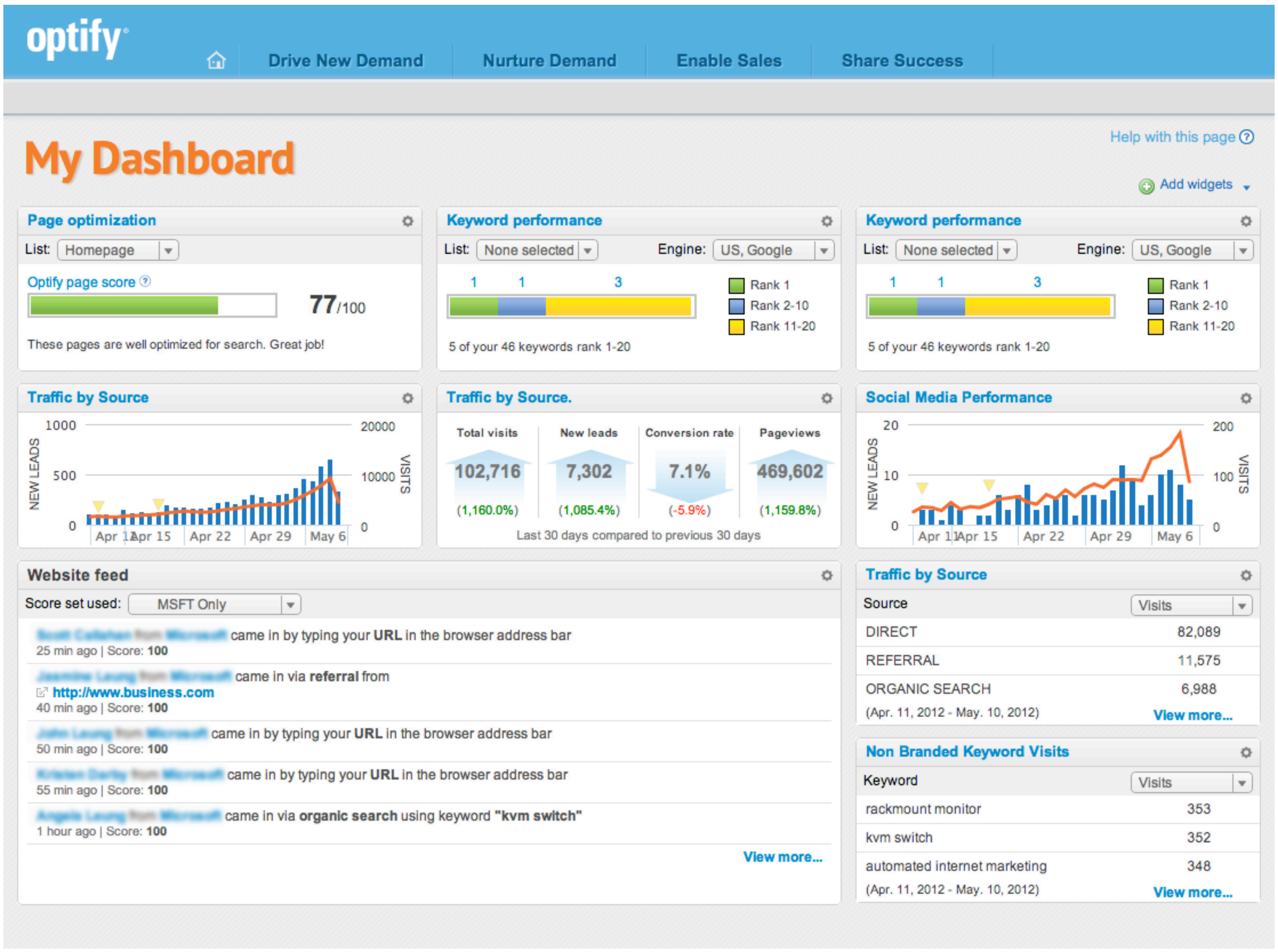
2.2 Why use Web analytics?

Web analytics are used for many purposes, but there are three main uses, which are **canned reports, data mining, and dynamic content presentation**:

- **Canned reports:** Specific reports can be generated (typically on the fly or at specified intervals) to report on a number of activities. The set of reports does not typically change. They normally report simple statistics about the number of visitors, popular entry pages, and distributions of traffic by time of day, but also many other metrics. Some offer clickstream analysis—showing the popular sequences of clicks and pages that users take to navigate your site. The efficiency of your sales funnel can also be measured by examining the conversion rate of every step in your sales or checkout funnel (Figure 2.1 shows an example of canned report inside Optify's website). [OP13]

- **Data mining:** Some systems have flexible reporting and scripting languages that allow you to construct your own specialized reports based on historical data. This supports open-ended discovery and ongoing questioning. Do our search engine visitors buy more often than our banner ad visitors? What is our repeat order rate within a six-month window of an initial purchase?

- **Dynamic content presentation:** Many Web analytics systems have begun to overlap with Web content management systems and support the ability to change content on the fly. They encode business rules within your Web pages that can change specific portions of your content based on the actions of a particular user. In addition to testing landing pages, you can also introduce up-sells or cross-sells to your buyers, or display special offers to repeat visitors.



My Dashboard

Page optimization

List: Homepage

Optify page score 77/100

These pages are well optimized for search. Great job!

Keyword performance

List: None selected Engine: US, Google

1 1 3 — 🟩 Rank 1, 🟦 Rank 2-10, 🟨 Rank 11-20

5 of your 46 keywords rank 1-20

Keyword performance

List: None selected Engine: US, Google

1 1 3 — 🟩 Rank 1, 🟦 Rank 2-10, 🟨 Rank 11-20

5 of your 46 keywords rank 1-20

Traffic by Source

Traffic by Source.

Total visits	New leads	Conversion rate	Pageviews
102,716	7,302	7.1%	469,602
(1,160.0%)	(1,085.4%)	(-5.9%)	(1,159.8%)

Last 30 days compared to previous 30 days

Social Media Performance

Website feed

Score set used: MSFT Only

[redacted] came in by typing your **URL** in the browser address bar
25 min ago | Score: **100**

[redacted] came in via **referral** from
http://www.business.com
40 min ago | Score: **100**

[redacted] came in by typing your **URL** in the browser address bar
50 min ago | Score: **100**

[redacted] came in by typing your **URL** in the browser address bar
55 min ago | Score: **100**

[redacted] came in via **organic search** using keyword **"kvm switch"**
1 hour ago | Score: **100**

View more...

Traffic by Source

Source	Visits
DIRECT	82,089
REFERRAL	11,575
ORGANIC SEARCH	6,988

(Apr. 11, 2012 - May. 10, 2012) **View more...**

Non Branded Keyword Visits

Keyword	Visits
rackmount monitor	353
kvm switch	352
automated internet marketing	348

(Apr. 11, 2012 - May. 10, 2012) **View more...**

Figure. 2.1, Canned report example.

3. Data-driven product development

People naturally follow a similar mental trajectory when they make decisions. If we are trying to persuade them to take action, we also must have a firm grounding in this entire process.

Having many quantitative and qualitative data sources is probably only the half of the solution, the real challenge and opportunity is how to interpret that data into actionable insights for the optimization process, we also need to understand who our customers are, and also what they're trying to accomplish.

No change in the product is small enough to take it lightly, every change we do in the product could have a noticeable impact in our product, some elements may have more influence than others, but you will never know which one will affect more to your product before you **experiment** it.

The experiments are a key factor on data-driven product development, before we decide if a change is good or not for our product, we need to test it in a experiment, in this chapter we're going to introduce two kind of experiments, the A/B testing which is the kind of testing that we used in our Smarty project, and the multivariate testing.

Also we are going to see the other key factors involved in data-driven product development, what are the advantages and disadvantages, and why is so important for a successful product.

3.1 Developing an action plan

Data-driven product development is based on experimentation, and experiment is not a casual thing, experimentation requires planification, setting goals and hypothesis and after gathering conclusions.

In order to develop an action plan, we need to involve all the team, as we need to gather knowledge of all the areas of the company, if we don't do this, our goals and hypothesis can be biased in a particular point or view.

We are going to explain better this concepts in the following lines:

Goals

Before we run any experiment, we should think carefully what we're trying to achieve. A goal can be whatever we want to improve, a goal can be something simple and concise like growing the number of visitors on our site or increase the conversion. Or something more complex like increasing the orders on certain product a 5 percent.

Anyway the experiments usually works better with one simple goal, that will be the answer to wether the experiment is good, bad, or simply makes no difference on the product.

In peerTransfer we set many goals in our experiments inside project smarty, we expected to improve the use of certain currencies, to low the calls to customer service, I'll give a full example of this goals in the Project Smarty chapter.

Hypothesis

"An Hypothesis is a supposition or proposed explanation made on the basis of limited evidence as a starting point for further investigation" [CO12]

The hypothesis states what you think you will find in the experiment. When the hypothesis is build you'll be making a plan to look for evidence, for example "If we observe the variable X, then it will prove that there must be a relationship between this variable and the outcome.

3.2 Experiments

An experiment is the combination of a hypothesis and a goal. Is a means of gathering information to compare an idea against reality.

We can illustrate this concept with the following example, 37signals[2] wanted to test it's landing page for the Highrise product [Figure 3.1], they didn't knew what kind of design will improve the conversion rate[3] so they made a few AB testings[4] with different designs, with the purpose of test it against the original design, the best design has to be the one with the better conversion rate.

The hypothesis was that the users were getting confused on the home page as it was bloated with too much text and figures and it had a bad structure, so many of them were leaving the page before registering in order to try the product.

The goal was to improve the ratio of users entering the site for the first time and after registering in order to try the product.

[2] 37 signals, founded in 1999 is the company behind the succesful products Basecamp, Campfire, and the web development framework Ruby on Rails.

[3] In internet marketing, "the conversion rate is the proportion of visitors to a website who take action to go beyond a casual content view or website visit, as a result of subtle or direct requests from marketers, advertisers, and content creators" [WIK13]

[4] As we will see later, an AB testing is a way of test a new variation against the original in order to measure which performs better.

One of the experiments, was focused on improve the copy, and the structure of the page, with a better design that improves the legibility and more information about the product itself [37S13]:

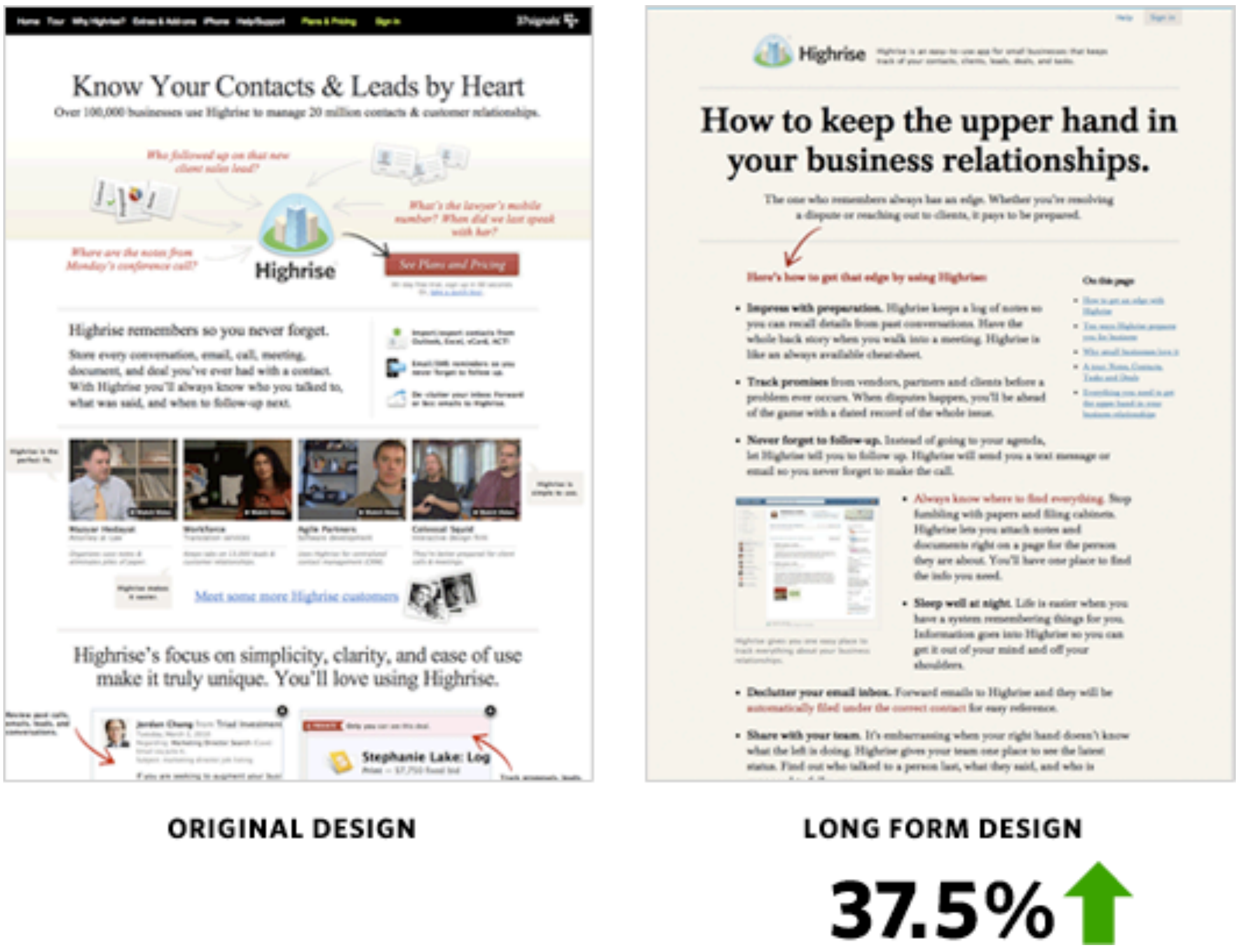

Know Your Contacts & Leads by Heart
Over 100,000 businesses use Highrise to manage 20 million contacts & customer relationships.
Highrise
Highrise remembers so you never forget.
Meet some more Highrise customers
Highrise's focus on simplicity, clarity, and ease of use make it truly unique. You'll love using Highrise.
How to keep the upper hand in your business relationships.
Here's how to get that edge by using Highrise:
ORIGINAL DESIGN
LONG FORM DESIGN
37.5%

Figure 3.1 First variation for the Highrise AB experiment

With this experiment, the conversion rate grown by a 37.5%, a pretty good improvement vs the original design. Another experiment consisted in a different kind of design, more clear and concise, the design consisted in a big picture of a girl and less text [Figure 3.2]

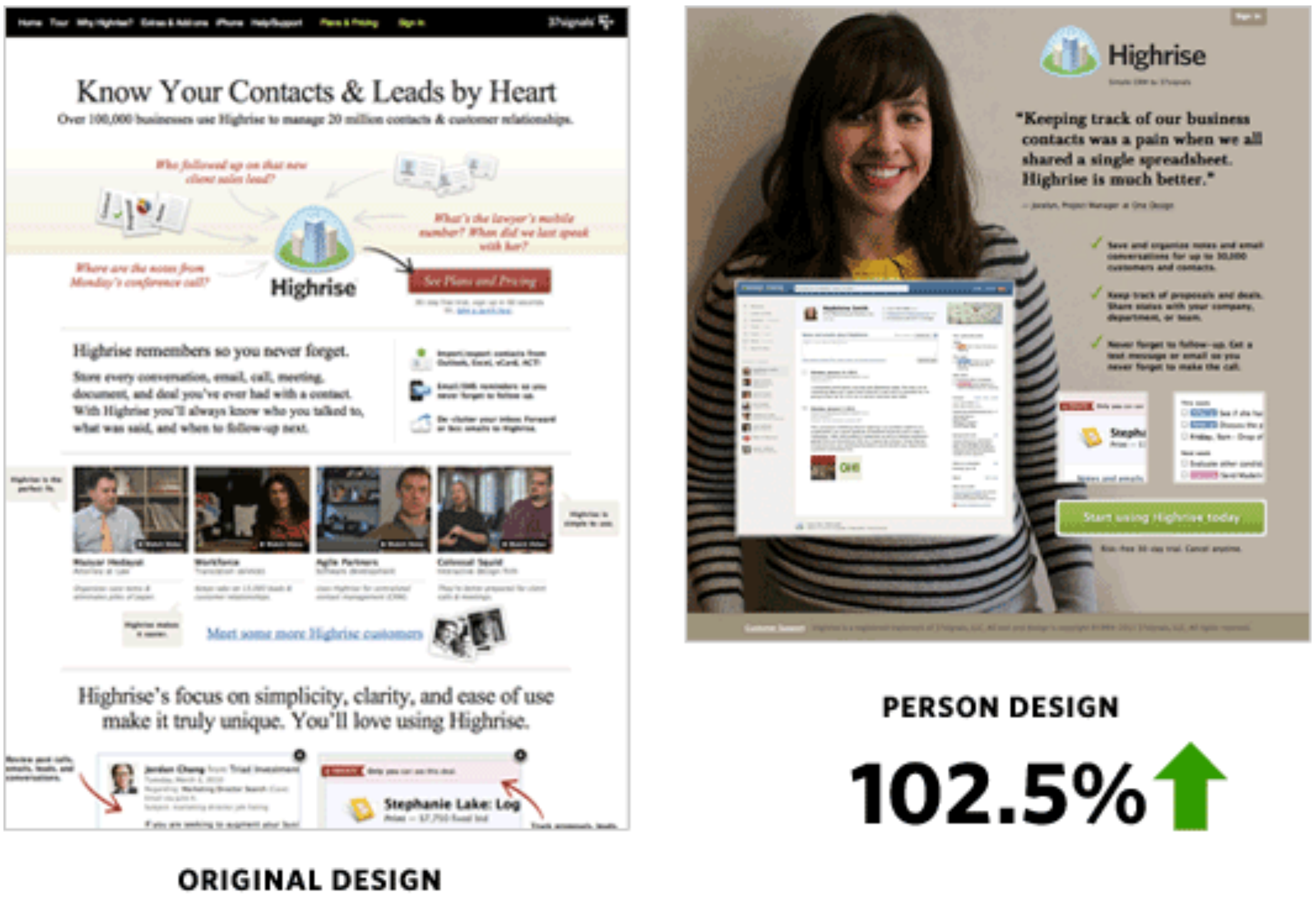
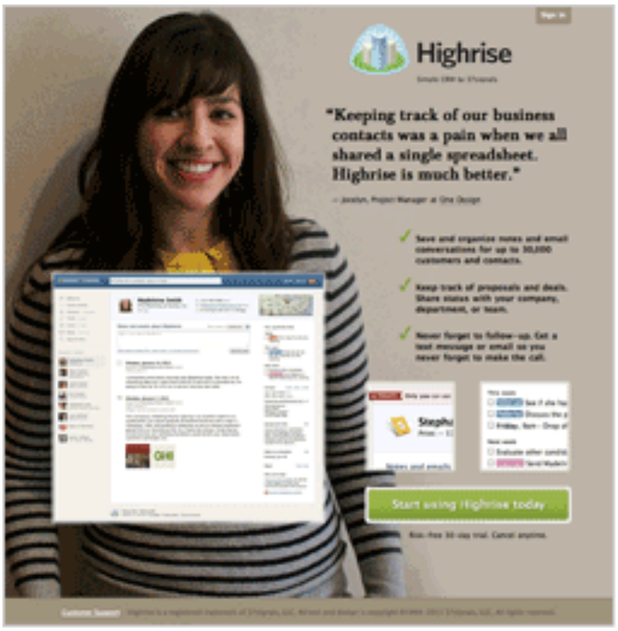

Know Your Contacts & Leads by Heart
Over 100,000 businesses use Highrise to manage 20 million contacts & customer relationships.
Who followed up on that new client sales lead?
What's the lawyer's mobile number? When did we last speak with her?
Where are the notes from Monday's conference call?
Highrise
See Plans and Pricing
Highrise remembers so you never forget.
Store every conversation, email, call, meeting, document, and deal you've ever had with a contact. With Highrise you'll always know who you talked to, what was said, and when to follow-up next.
Meet some more Highrise customers
Highrise's focus on simplicity, clarity, and ease of use make it truly unique. You'll love using Highrise.
ORIGINAL DESIGN
Highrise
"Keeping track of our business contacts was a pain when we all shared a single spreadsheet. Highrise is much better."
Start using Highrise today
PERSON DESIGN
102.5%

Figure 3.2, Second variation for the Highrise AB experiment

The results were surprising as even tough the new design had less information about the product, the version with the girl converted a **102.5%** more vs the original design, only with a change in the design and omitting information about the product.

They thought about the results, and decided to do a new experiment with a design with more information after the picture but keeping the photo of the girl, that design will combine the two designs that converted more than the original:



Highrise

"Keeping track of our business contacts was a pain when we all shared a single spreadsheet. Highrise is much better."

— Jocelyn, Project Manager at One Design

✓ Save and organize notes and email conversations for up to 30,000 customers and contacts.

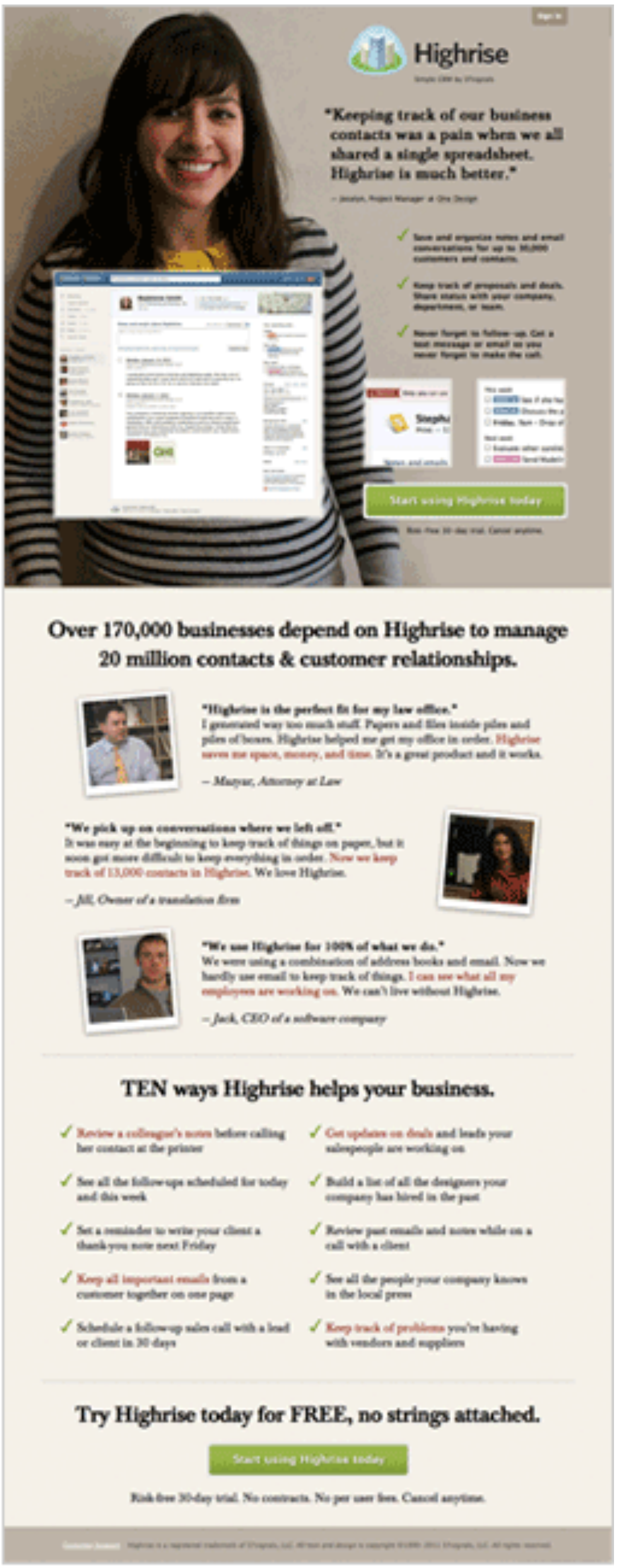
✓ Keep track of proposals and deals. Share status with your company, department, or team.

✓ Never forget to follow-up. Get a text message or email so you never forget to make the call.

Risk-free 30-day trial. Cancel anytime.

PERSON DESIGN

102.5% ⬆



Highrise

"Keeping track of our business contacts was a pain when we all shared a single spreadsheet. Highrise is much better."

— Jocelyn, Project Manager at One Design

✓ Save and organize notes and email conversations for up to 30,000 customers and contacts.

✓ Keep track of proposals and deals. Share status with your company, department, or team.

✓ Never forget to follow-up. Get a text message or email so you never forget to make the call.

Risk-free 30-day trial. Cancel anytime.

Over 170,000 businesses depend on Highrise to manage 20 million contacts & customer relationships.

"Highrise is the perfect fit for my law office."
I generated way too much stuff. Papers and files inside piles and piles of boxes. Highrise helped me get my office in order. Highrise saves me space, money, and time. It's a great product and it works.

– Mazyar, Attorney at Law

"We pick up on conversations where we left off."
It was easy at the beginning to keep track of things on paper, but it soon got more difficult to keep everything in order. Now we keep track of 13,000 contacts in Highrise. We love Highrise.

– Jill, Owner of a translation firm

"We use Highrise for 100% of what we do."
We were using a combination of address books and email. Now we hardly use email to keep track of things. I can see what all my employees are working on. We can't live without Highrise.

– Jack, CEO of a software company

TEN ways Highrise helps your business.

✓ Review a colleague's notes before calling her contact at the printer

✓ Get updates on deals and leads your salespeople are working on

✓ See all the follow-ups scheduled for today and this week

✓ Build a list of all the designers your company has hired in the past

✓ Set a reminder to write your client a thank-you note next Friday

✓ Review past emails and notes while on a call with a client

✓ Keep all important emails from a customer together on one page

✓ See all the people your company knows in the local press

✓ Schedule a follow-up sales call with a lead or client in 30 days

✓ Keep track of problems you're having with vendors and suppliers

Try Highrise today for FREE, no strings attached.

Risk-free 30-day trial. No contracts. No per user fees. Cancel anytime.

LONG FORM PERSON DESIGN

22.72% ⬇

Figure 3.3, Third variation for the Highrise AB experiment

Surprisingly the conversion rate descended a 22.72% with the combined design. The clear moral behind this results is that **it's impossible to know in advance if a new design will improve the metrics of our site or not, we need to experiment every new design in order to know it.**

A experiment is not only consisting on design changes, it could be a small refinement, a completely new feature or even you can experiment not adding features but eliminating them for the sake of simplicity of the application.

3.2.1 Designing experiments

Analytics are a great resource for finding areas of opportunity in your site, accessing analytics for your site's conversion indicators is relatively easy as almost every online business has some form of analytics, but also is important to know where you should find those opportunities to improve the conversion rate of your site.

It's important to set some filters when looking for elements to experiment in order to improve your conversion and other key metrics, as if not, the list of candidates can be too big and we can lost time and resources experimenting with elements that don´t have impact in our key metrics.

On the following points I´m going to explain the most important filters to limit the search on potential candidates to experiment:

Follow the Pareto principle

Usually a few of something is responsible for the vast majority of the results[5], If this principle is applied to site optimization, it follows that fixing the most fundamental problems will result in higher improvement of the metrics than fixing the other 80%.

Most important conversion actions

The importance of possible candidate to experiment, not only comes from the Pareto principle, the importance of the financial rewards should not be forgotten, as you also have to concentrate on improving the ones that have more impact on the revenue. If for example you have an offering of four different pricing plans, you have to concentrate your efforts on the most popular one, as in that case you stand to gain the most. If your least popular plan is only accounted for 1% of sales, the improvements you make on that plan will not have a great impact on the revenues.

Biggest possible audience

We always should give more priority to the pages with higher traffic level, that consequently result in the highest number of conversion. It's important to keep in mind that those pages

[5] The Pareto principle (also known as the 80–20 rule, the law of the vital few, and the principle of factor sparsity) states that, for many events, roughly 80% of the effects come from 20% of the causes.

with high traffic, and high conversion are not perfect, and you should try to improve them, so that kind of pages usually are an excellent candidate to experiment.

Also, many times companies have multiple pages for specific campaigns. It's important to examine which ones have the biggest audience levels and result in the highest number of conversions. That pages are generating the most revenue, so we should give top priority to them.

Most popular paths on the site.

Web analytics software shows you the most popular paths (flows of traffic) through your site. Some of these packages even show you the reverse goal paths—the common sequences of pages that led the visitor to the conversion action.

It´s easy to discover the most popular paths through your site with Web analytics software, even it´s possible to discover the **reverse goal paths**[6].

Analyzing paths it´s a complex job as it involves multiple interacting factors. In order to do this we must know where the traffic lands in our site, the traffic may land on different kind of pages of the site. For example, for a shop, you can have an important amount of traffic hitting the home page, the main categories and product detail pages, the combination will depend on the particular business.

[6] "*A reverse goal path is a way to find the path that was taken to reach a defined goal*." [LM12] i.e, if the goal page is www.ABC.com/site/page/goal the reverse goal path will be the reverse path that was taken to reach the goal, so a user may have started at ABC.com then ABC.com/site then ABC.com/site/page then finally reached the goal at ABC.com/site/page/goal

Usually, a big amount of traffic lands on pages different than the home page. This deep linking[7] is used to present the most relevant content possible. Deep linking is common on Pay Per Click campaigns, where the intent of the searchers can be inferred from their keywords. Who use generic keywords will be redirected to the home page, and the other ones that use more specific keywords can be redirected to more specific pages like a product details or a certain category.

This kind of redirects improves the conversion significantly for the users that use specific keywords, as this users land in a more advanced position inside the decision process. It's possible to combine all these factors in one metric, this metric has the purpose of estimate the quantity of the potential losses for each type of page on your site:

- **Potential lost revenue:** In order to calculate this metric, you must get a revenue estimation for the traffic source which is obtained my multiplying the revenue per visitor for a certain type of page by the traffic (number of visitors) which lands on that page. Now you can get the potential lost revenue by multiplying the revenue estimation by the bounce rate (the percentage of users that leaves the site without visiting any other page). With the potential lost revenue in mind is easy to rank our pages and focus our experiments on the ones with a bigger potential lost revenue.

Unfortunately, in the real world this is a little more complicated, as many times a page can be used not only as landing page but also as a link inside the conversion path from other

[7] In the context of the World Wide Web, "*deep linking consists of using a hyperlink that links to a specific, generally searchable or indexed, piece of web content on a website (i.e.http://example.com/path/page), rather than the home page (i.e. http://example.com/)*" [WIK13]

pages upstream on it. In this cases, the lost revenue calculation can be extended to include not only the bounce rate but also the drop-off rate for the traffic that is passing through the page.

We can illustrate this concept with an example, if we are running a site for generating car seller leads, and we get a certain amount for each visitor that we deliver to a car seller home page. We're a using a combination of general and specific keywords (i.e *"buying a car", "car in Boston", etc*)

Depending on how specific are the keywords, the traffic can land on the home page which is country wide, a state specific page, or a local page as the cars that are being sold on the site are organized by locality.

If the user lands in the home page, they must select a state page and then a local page in order to take the conversion action, which in this case is selecting a specific seller page. in the same way the users that land on the state pages must first click on a local page, and then click on the paid link.

So, in our example, the three different types of page have the following numbers:

- **Country wide page**:
  - CV = 1.000.000 visitors per month.
  - CB = 50% bounce rate.
  - CR = 0.20$ revenue per visitor.
- **State pages:**

  - SV = 500.000 visitors per month
  - SB = 40% bounce rate
  - SR = 0.30$ revenue per visitor
  - SA = 35% abandonment rate

- **Local pages:**
  - LV = 200,000 landing page visitors per month
  - LB = 35% bounce rate (for traffic landing on this page)
  - LR = $0.50 revenue per visitor
  - LA = 30% abandonment rate (for through traffic)

With this numbers in mind, we can calculate the potential lost revenues for the three different pages this way:

- **Country wide page:** CV × CR × CB = $100,000
- **State pages:** [(SV × SR × SB)+ CV] × CR × (1 - CB) × SA = $95,000
- **Local pages:** [(LV × LR × LB) + SV] × [SR × (1 - SB) × LA] + [(CV × CR) × (1 - CB) × (1 - SA) × LA] = $83,000

As we can see, the lost revenue of each page is surprisingly close, especially given the significant differences in direct traffic levels on each type of page. The reason of these is because the state and local pages act as conduits for upstream traffic, so their value is enhanced significantly.

Use face to face to get feedback

Not all the knowledge is content on the metrics, you should try to get as much information as you can directly from your users, this is probably the best way to collect feedback.

You can also use a software like Silverback[8] to record people that uses your site, and investigate what are the weak points on your user flows.

Most important parts of the page

Not all the elements on the page have the same value, depending on what task the user is trying to accomplish in our site, their behavior will be different. For example, in an e-commerce shop, if the user is looking for a particular item, they will skip all the non relevant items until they found the one that are looking for, also if we take the example of a user reading articles in a newspaper, the user / visitor will scan the articles starting from the upper-left side and focusing with decreasing attention to each new subheading or entry in the list.

Although the combinations can be almost infinite, there are some patterns that we can follow in order to optimize the most prominent parts of our pages. With Eye-tracking and conduct studies, the scientists have demonstrated the following patterns:

- The users pay more attention to the contents near the upper-left corner of a page when they are trying to get oriented in the page.

[8] http://silverbackapp.com/

- The users look for the content that they're looking for in the central portion of the page and usually ignore the contents located in the upper-right and lower-left corners.

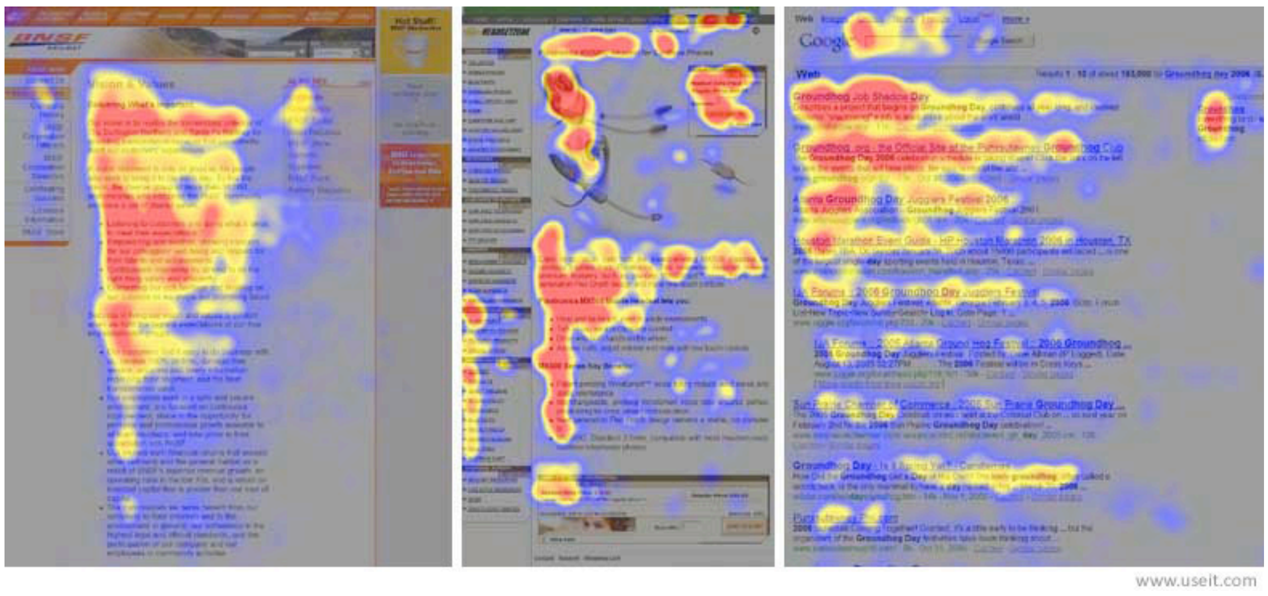


Figure 3.4 Eye tracking test on a website

Test the testing environment

Doesn't matter if you choose to implement your own testing platform, or you decide to use some third party solution, in order to be sure that your testing environment is working accurately, it's crucial to test the test platform itself.

In our project smarty we discovered that after a successful experiment we weren't increasing the revenue, so we suspected about our testing platform. In order be sure that the testing platform is accurate, it's crucial to run an **A/A/B test**[9] on the first run**,** in other words, test

[9] The most extended practice in this case is to run an A/A test, but altough it seems quick and easy, it can cause other problems. [GN12]

the control against the control and another version, if you see a significant difference in the results of both controls, it's clear that the testing environment is not working properly.

Granularity of the experiments

We refer to granularity as the level of detail at which the changes are made to the site. For example we can do specific and little variations to an element (like changing the font size or the color), or we can change the entire design and disposition of elements of a page, consequently this kind of changes are composed by many little variations.

Although this is very common, it can be extremely difficult to extract right conclusions on experiments that are too big, as many changes interact to each other. It's also difficult to guess the "perfect" size for a test, but this size is constrained by the traffic and the data rate[10]. Another advantage of fine granularity changes is that they are quick and easy to implement.

By continuously running fine-granularity tests, it's possible to make great improvements on conversion. As these kind of experiments which are small and incremental, can be done without great planification or resources, and suit perfectly in a continuous integration - testing methodology.

[10] In this case, we refer as "data rate" by the number of conversions per time unit.

Although we pointed the advantages o fine-granular test, sometimes we don't have another option that make bigger experiments, maybe because we don't have enough data rate or simply we don't have the time for doing multiple series of fine-granular experiments.

As we pointed, experiments with big redesigns are difficult to create, as require more time and effort. Since, we don't know in advance if the new design will perform better than the original, we're assuming some risk with this kind of experiments as we are "investing" more resources on it.

Even if the new design performs better than the original, the conclusions will be harder to extract, and we can lost some "learnings" about which individual elements contributed the most to the improved performance, and consequently we can't apply this knowledge for future experiments.

Playing with the granularity of the experiments, allow us not only to include all or the most important components but also fitting them into a reasonable experiment size, consequently we are reducing the search space for our conclusion.

Minimize Friction

Reducing friction is always a good idea, friction is any element on your site that slows down conversion, the most common items that cause friction are:

- **Form fields**: the more information we request, the slower will be the process, we should justify every single form field and try to reduce all the non-necessary.

- **Page length:** Usually is not a good idea to have very long pages, reducing the length of the pages will reduce the friction most of the times.

- **Process Steps**: The less steps a process haves, the more quicker will be, so we should try to control the number of steps for the different actions in our site.

Learn from pricing takeaways

It's very difficult to figure out what is the most profitable price for a product, so it's very important to build hypothesis with real data, and after that test that hypothesis with an A/B test.

But when testing a price, it's not all about the price, all the details surrounding the product like the value behind the price or the presentation are extremely important. It's also important to base the price on what the customer thinks your product cost, without considering how is your cost of operation or production.

Learn from Higher Prices

As it’s false that lower prices will always raise sales, it’s also false that raising prices will drop the sales, when we make A/B tests on prices, a good idea is to work with small increments and batches, this way you can make a better approximation on the ideal price.

Test All Best Practices and tips

Best practices and tips doesn’t work for every single site and situation, so don’t have blind faith on those best practices and tips (like the described here), and make tests for assure that work for you.

Conversion Trinity

The Conversion Trinity is a formula based on three steps:

- **Relevance.** Are you relevant to my wants/needs/desires (search query)? Have you maintained scent?

- **Value.** Do I know why you are the right solution for me? Have you explained your value proposition/offer well?

- **Call to action.** Is it obvious what I need to do next? Have you given me the confidence to take that action?

3.2.2 Looking for conversion opportunities

We can do a sort of analysis in the components of our site in order to find those potential candidates, this analyses combined with the filters presented in the last chapter will bring us the best candidates for experimenting:

- **Form analysis:** Forms are always tricky for customers. Try to identify any form fields that are giving customers problems as are too long, complex or have required fields that mustn't be required.

- **Funnel analysis:** A funnel is a well-defined flow on your website, i.e the checkout process, registration, lead generation or anything where users take a series of actions before reaching some sort of goal. So, the very first thing to do is find where these funnels occur. One example would be splash page -> demo -> sign up. This obviously varies depending on the business, but almost everyone can benefit from figuring out their funnels and how users flow through them. [Figure 3.4]

- **Social features:** An interesting study done by Empirica Research found that removing social media buttons (Facebook like and Twitter tweet) on a Clearasil skin product page, raised the sales by up to 25 percent. Some products are just too sensitive to think about sharing, so it's possible to lower the sales of that product if we place social media buttons.

- **Bounce rates:** represents the percentage of visitors who enter the site and "bounce" (leave the site) rather than continue viewing other pages within the same site. So you have to analyze the pages where your customers aren't engaged and are leaving the site and consider ways to improve those pages.

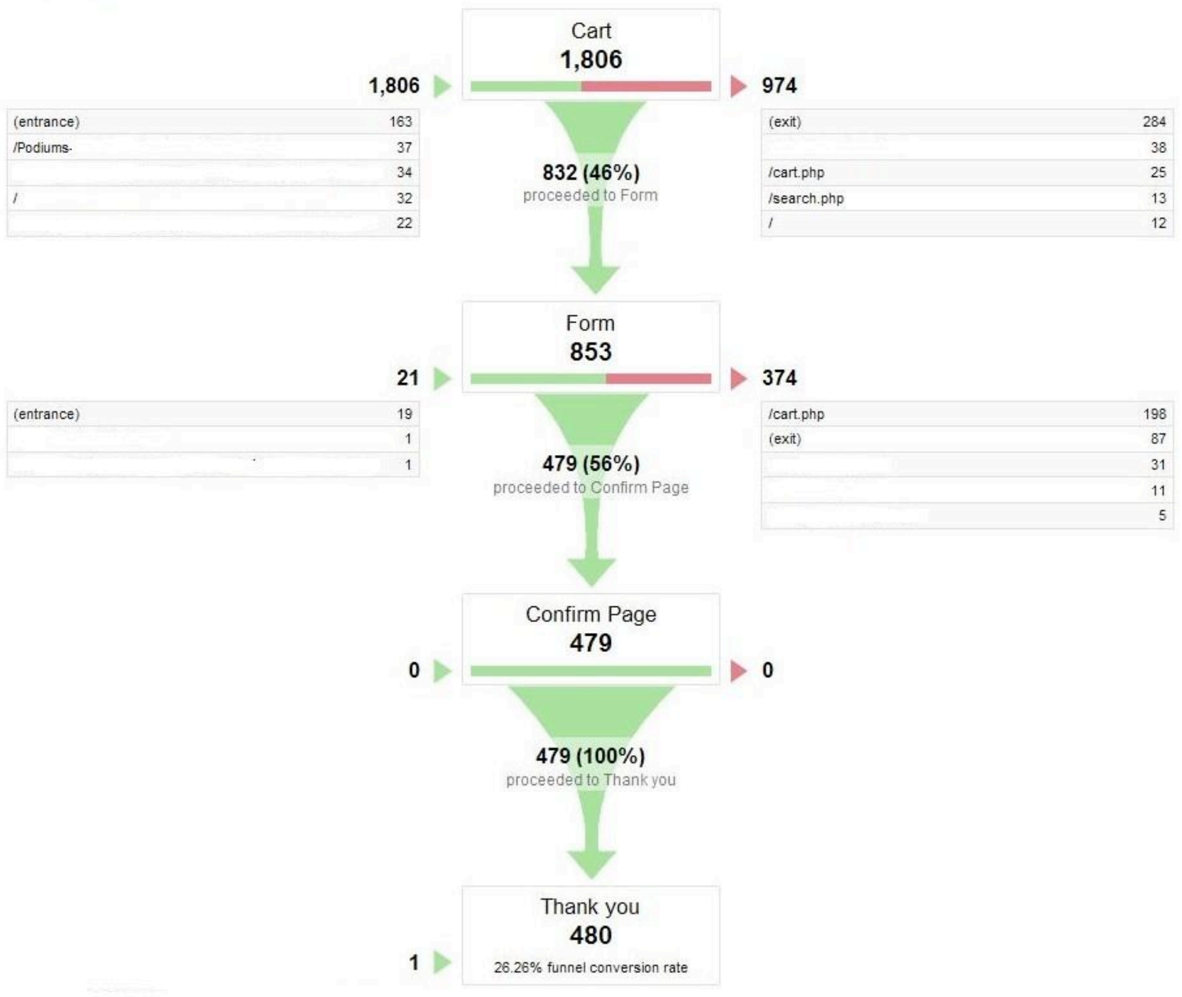


Figure 3.4 Funnel analysis

- **Page conversion:** Compare low and high-converting pages understanding the differences that could be driving differing performance by page.

- **Path analysis:** Identify where customers go next from your key pages or steps. If it's not where you want them to be, figure out why they are distracted and why they choose a "wrong" path.

- **Page structure:** defines how we organize and structure the components on our site. With only changing the structure of the page we can have a great impact in our metrics. Typical page structure testing elements include:
  - Size and contents of page header and page footer
  - Size and location of page navigation
  - Placement of trust symbols and credibility logos
  - Separation of page shell and navigation from page content
  - Size and location of forms or other calls-to-action
  - Mirror images (swapping) of key page sections (e.g., a form located to the left of the text or to the right)
  - Vertical stacking versus horizontal arrays of page sections
  - Single versus multiple columns

- **Information Architecture**: it defines the way that information is organized on the site. The most common information architecture–related test elements include:
  - Self-selecting by role or by task
  - Clear and distinct descriptive link text and choices

  - Sensible and prominent page titles
  - Breadcrumbs or other context inside the structure of the site
  - Consistent placement of all page elements
  - Navigation (organization of menu options)
  - Number of available choices presented
  - Alternative navigation methods
  - Cross-linking to other key information
  - Availability and format of on-site search
  - Ability to avoid, minimize, reverse, or easily correct mistakes

- **Presentation**: defines how we want to present our contents in a aesthetic way. Typical presentation testing elements include:
  - Degree of detail (e.g., full text, or links to supporting information)
  - Writing format
  - Choice of input elements (e.g., radio buttons or pulldown lists)
  - Action format (e.g., buttons, text links, or both)
  - Use of alternative formats and modalities (e.g., charts, figures, audio clips, video, presentations, demos)

- **Emphasis**: Emphasis is about the relative importance that we place on an element. But, as all the elements on the page interact to each other, we can't simply place the "highest" emphasis to every element in our site, so we need to find some balance. The most common emphasis elements are:
  - Amount of screen portion designated to the component.

- Use of relevant images (e.g., specific product or believable people)
- Image captions
- Font sizes and font families (e.g., headline sizes)
- Font emphasis
- Background color blocks or background images
- Visual separators (e.g., horizontal rules)
- Use of whitespace and visual isolation to focus on important items
- Removal of distracting secondary information

3.3 A/B testing and multivariate testing

A/B testing and multivariate testing are two different ways to probe hypothesis. In this chapter I'm going to define them, and to compare both, also I'm going to point some tips about when and where to use this techniques.

Perhaps in our Smarty project we used only A/B testing, it's important to introduce multivariate testing, as although is a more complex tool, it's very powerful.

3.3.1 A/B testing

We introduced the idea of AB testing in the chapter about experiments, in this chapter I'm going to deepen on the idea of AB testing, and also to introduce some examples.

A/B testing, also known as split testing, *"is a method of testing through which marketing variables are compared to each other to identify the one that brings a better response rate."* [HB12] [Figure 3.5]

In this context, the element that is being testing is called "control" and the element that is argued to give a better result is called "treatment."

When an A/B testing is running on the site, the traffic is split on two halves, one half will run the A part of the experiment, and the other will run the B part, a system will store the required user interactions in order to get the results after the experiment is finished.

When a user enters the site, is randomly addressed to one of the versions, we can limit this randomness by limiting if a user can see the other version after seeing one of the versions.

For example if a user entered the site and saw the A part of the experiment, we can set a cookie on the user in order to force this version of the site every time the user enters the site, that way we minimize the possible confusion that a user can feel if he see different versions every time he enters the site.

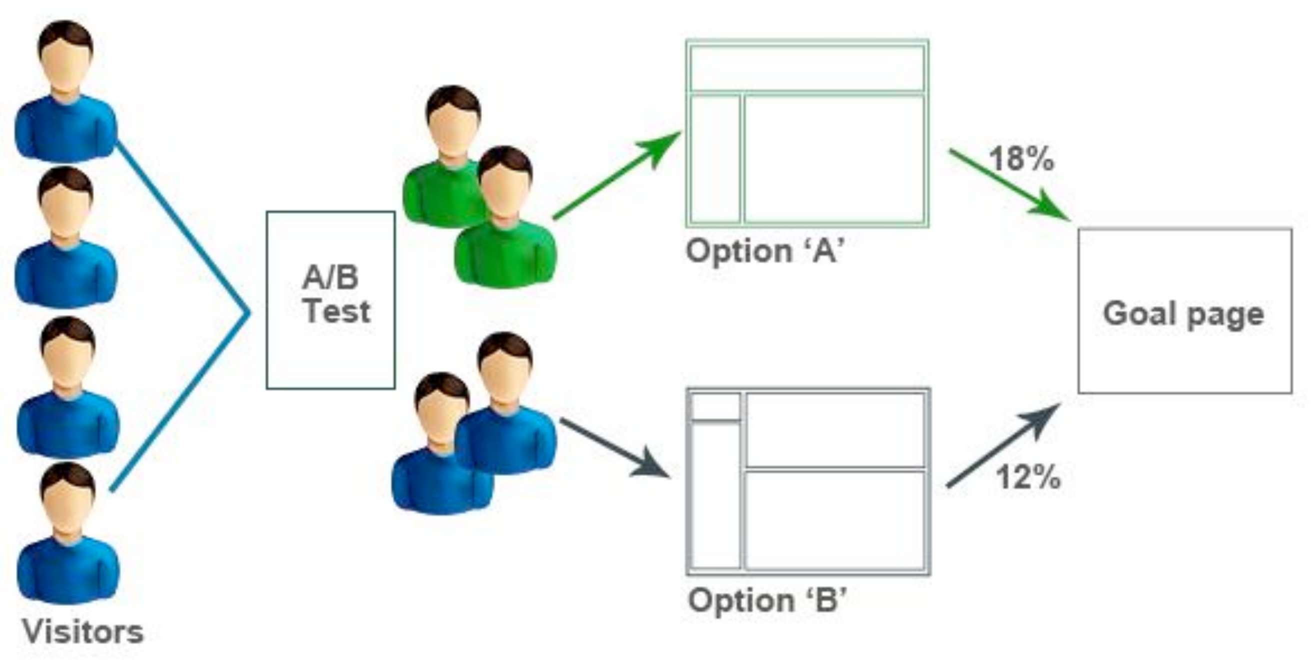


Figure 3.5, AB Testing

3.3.2 A/B testing, advantages and disadvantages

The A/B testing method has many advantages but also have some disadvantages, in this chapter we are going to point the most important ones:

A/B testing advantages

- **Easy implementation:** There are many solutions suited for doing A/B testing, most of them work pretty good with an affordable price. The solutions are easy to use, as many of them can be used without the support of the tech team, even it's possible to collect the data from your existing web analytics tools. Also if you have a very specific set of requirements it's possible to develop your own solution without wasting too much time.

- **The analysis of the results is easy:** As the winner of an A/B testing is very easy to determine, the analysis of the results are very easy as you only need to do very simple statistical tests, all you have to do is compare the baseline version to each challenger to see if you have reached a minimum confidence level.

- **Great flexibility on the definition of the variable values:** you have complete flexibility in how you the define the different alternatives for the test. For example, in one alternative you can simply choose to test a different font or size for the headline, and in another you can do a complete restructure of the site. This flexibility allows you to test a big range of different alternatives in one test, without the constraints of a more granular definition of variables like in a multivariate test.
- **Useful in low data rate tests:** For more advanced techniques you may need a big number of conversions or visits to your site, but is not the case for A/B tests. With the proper selection of the test variable and alternative values, you can still achieve significant results in a split test although your numbers are not so big.

A/B testing disadvantages

- **The variable interactions are not considered:** by it's definition, A/B testings only consider one variable each time, so the interactions between the variables are not considered in the further analysis, depending on how the variables interact with themselves you may find the best performing combination, for example, let's assume that you're testing two variables, the conversion rate for each combinations is as follows: *aa = 5%, ab = 2%, ba = 3%, bb = 7%,* with a multivariate test you can see that "*bb"* is the best performing combination. However, this is not possible if you had simply done two A/B tests (starting with V1 first). You would have tested recipe "*aa"* versus recipe "*ba"*. Since "*aa"* would perform the best, you would conclude that V1 should be locked in as V1a. You would then test V2 in this context by conducting another split test between "*aa*" and "*ab*". After this second test, you would come to the conclusion that V2 should be set to a. So the winning combination would be wrongly determined to be "*aa"*.

- **Limited number of combinations:** The number of combinations in a A/B test is usually small. If you review your site carefully, you probably came up with many potential issues, and also develop many alternative variations to test if perform better than the original one. However, because of the limited scope of A/B testing, you have to test each combination one at a time, also you have to guess which combinations you should test first, In other testing methodologies like multivariate testing, you can test many of your key ideas at once and find all of the combinations that improve your metrics in one test.

3.3.3 Multivariate testing

A/B tests are usually performed to determine the better of two content variations; multivariate testing can theoretically test the effectiveness of limitless combinations. The only limits on the number of combinations and the number of variables in a multivariate test are the amount of time it will take to get a statistically valid sample of visitors and computational power.

The main difference between A/B testing and multivariate testing, is that in multivariate testing it's possible to conduct tests where more than one component of the site may be tested in a live environment. It can be thought of in simple terms as numerous A/B tests performed on one page at the same time.

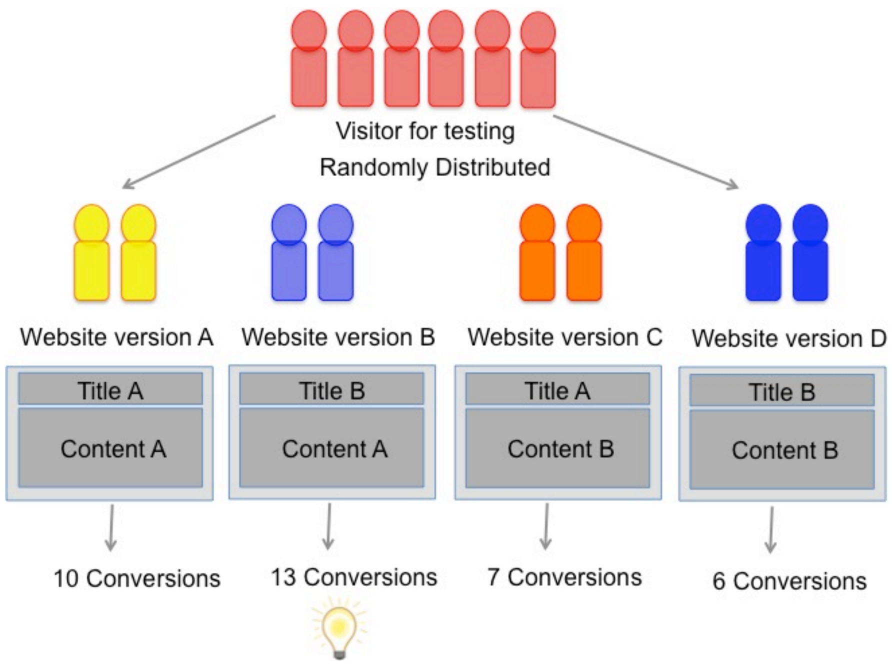


Figure 3.6, Multivariate testing

The purpose of multivariate testing is to simultaneously gather information about multiple variables, and then conduct an analysis of the data to determine which recipe results in the best performance.

Multivariate testing approaches differ on two important dimensions:

- **How the data is collected**: The data can be collected by two ways, full factorial or fractional factorial
- **How the data is analyzed**: The subsequent analysis can be either parametric or non-parametric. Within parametric analysis there are also significant differences. Some forms of parametric analysis take complex variable interactions into account, while others do not.

Multivariate testing data collection

There are two different kinds of data collection in a multivariate experiment:

- **Full factorial** experimental designs sample data across the whole search space. If this is done properly, the following analysis allows you to consider not only the main effects, but also all the interactions between the analyzed variables.
- **fractional factorial** experimental designs make simpler assumptions about the possible form of the parametric model for subsequent analysis. For example, they may simply assume that the values of the corresponding coefficients in the model are zero.

On the one hand full factorial parametric designs do not scale well, as that kind of design samples the data across all the search space, but also retrieves more complete information about the relationships among all main and interaction effects tested.

On the other hand fractional factorial designs can scale to larger search spaces, but make assumptions about the underlying process that may not be valid always, and may actually lead you to wrong conclusions.

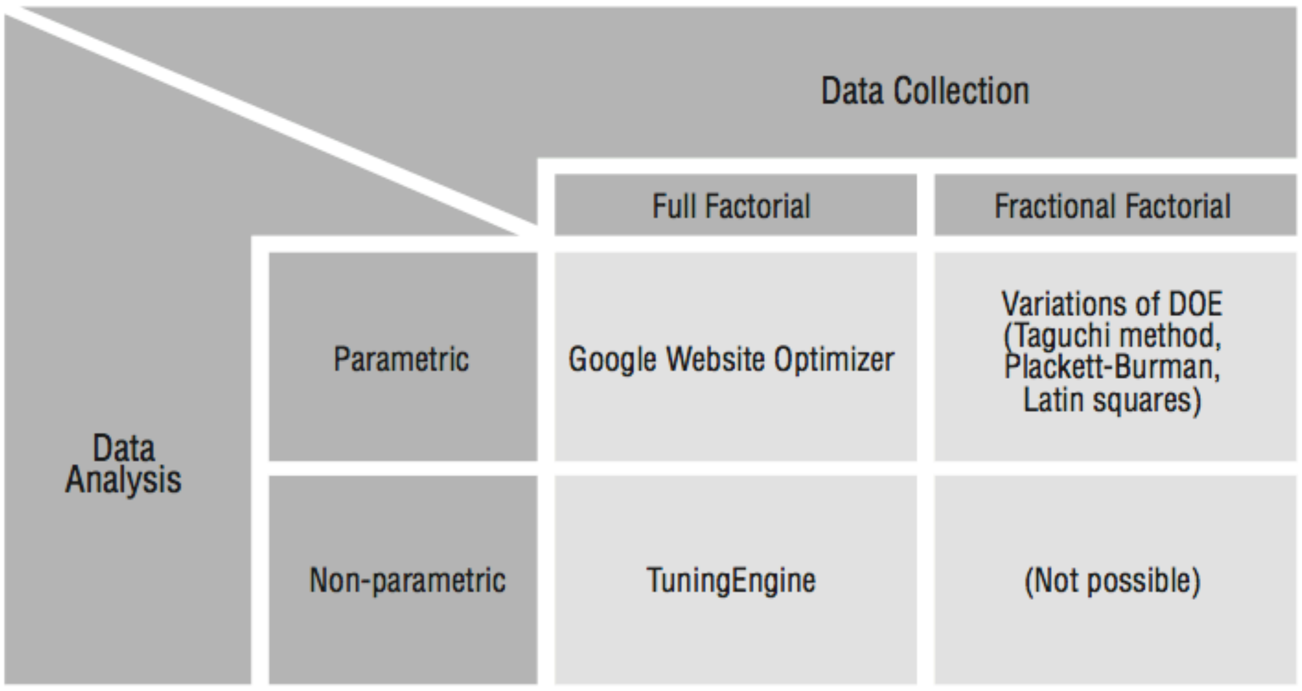
Data analysis

Also there are two kinds of analysis for a multivariate experiment:

- **Parametric** data analysis builds a model of how the variables tested impact on the conversion rate. For each combination in the search space, the model produces a prediction of the expected conversion rate, or other metric of interest.

- **Non Parametric** data analysis does not try to build a model based on the input variables. it tries to identify the best challenger combination, but without being able to note why is the best performing combination.

Combining the two data collection methods by the two data analysis methods we can compose the Figure 3.7



Data Analysis \ Data Collection		Full Factorial	Fractional Factorial
Data Analysis	Parametric	Google Website Optimizer	Variations of DOE (Taguchi method, Plackett-Burman, Latin squares)
	Non-parametric	TuningEngine	(Not possible)

Figure 3.7, Data collection and Data analysis table

3.3.4. Analyzing the results

One of the most important parts about experimenting, is to analyze the results. Understanding the results of the experiments can be tricky, specially if we are talking about an experiment that test deep and complex changes in our site.

In order to understand all the data that the experiments accumulate, we can make use of some simple methods:

Statistical significance

In a experiment, usually we attempt to hold all variables constant except the one that we want to investigate. The reality, of course, is that many variables are not constant. We're comparing similar groups, but those groups themselves are not equal, and consequently have differences that introduce noise. That's where statistical significance comes in [Figure 3.8]

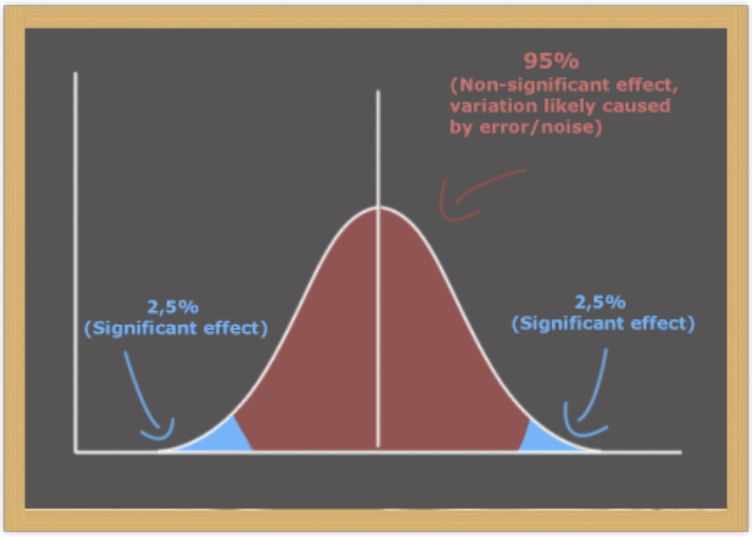


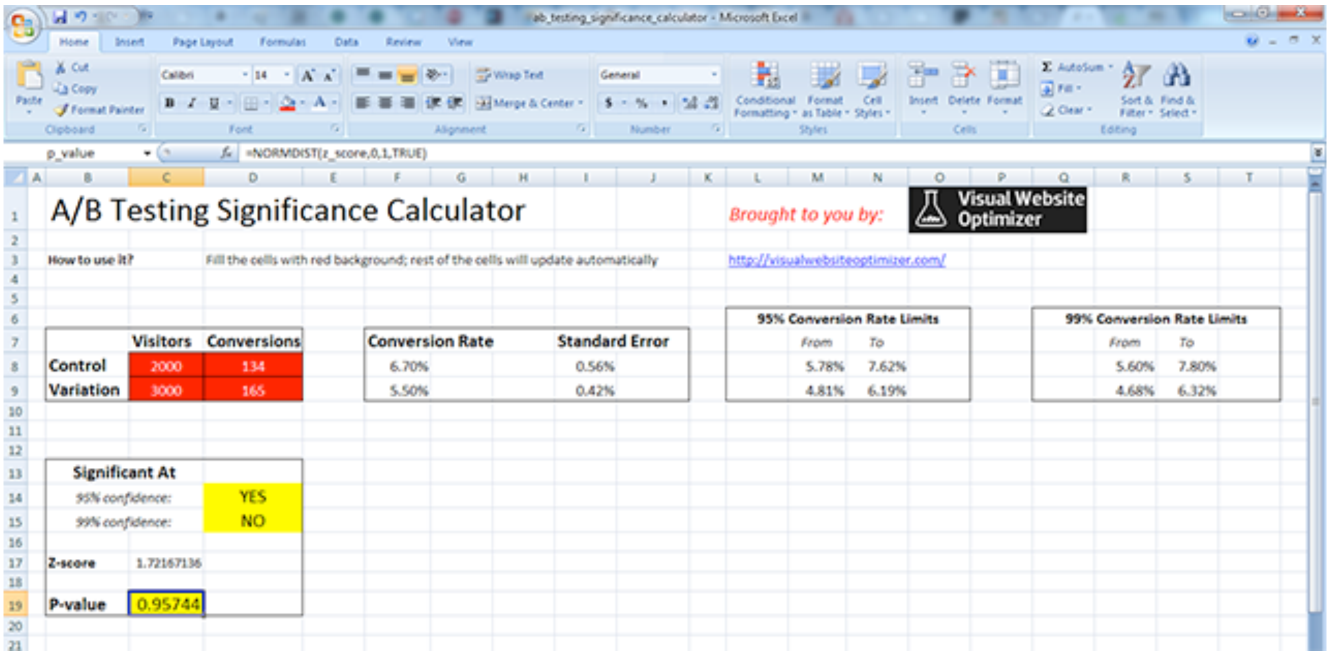
Figure 3.8, Statistical significance

Confidence doesn't gets higher always, usually it oscillates with the natural performance variances seen during the lifespan of the experiment, likely coming in and out of confidence many times.

It's important to not to simply extract conclusions when the experiment hits the 95 percent for the first time, as we just said, its probable that the confidence will oscillate, but instead let the experiment finish, and consider the confidence of the experiment for the entire duration.So **if the experiment has reached 95 percent confidence or above when finished, we can accept it as a winner.** There is still a 5 percent probability that the result was caused by chance, but it's feasible to accept this.

It's important to be careful with experiment results that never reach 95 percent confidence, as it's increasingly likely that you're reacting to mere noise in your experiment.

Vendor systems can calculate statistical significance for you as usually have that feature integrated in the product, but if we're using our own tool, we can make use of free tools like the one that Visual Website Optimizer[11][Figure. 3.9] offers, which is a simple calculator that helps to discover whether the experiment has reached an acceptable significance level.



p_value =NORMDIST(z_score,0,1,TRUE)

A/B Testing Significance Calculator

Brought to you by:

How to use it? Fill the cells with red background; rest of the cells will update automatically

http://visualwebsiteoptimizer.com/

	Visitors	Conversions
Control	2000	134
Variation	3000	165

Conversion Rate	Standard Error
6.70%	0.56%
5.50%	0.42%

95% Conversion Rate Limits	
From	To
5.78%	7.62%
4.81%	6.19%

99% Conversion Rate Limits	
From	To
5.60%	7.80%
4.68%	6.32%

Significant At	
95% confidence:	YES
99% confidence:	NO
Z-score	1.72167136
P-value	0.95744

Figure 3.9, Visual Website Optimizer tool

11 http://visualwebsiteoptimizer.com

Stabilization, standard deviation, and outliers

While statistical significance is the most reliable measure to understand the validity of the results, it isn't enough to choose a candidate as a winner. We should look for other parameters in order to make sure that our candidate is the winner:

- **Stabilization**: the experiment difference appears to be relatively stable over time
- **Standard deviation**: the measure of how much variation from the average there is in the data to determine how narrowly or widely the results are likely to vary
- **Outliers**: any large, atypical events that could impact the overall performance; i.e if four different people buy on the same day the most priced item in a store.

Segmentation

As we run our experiment and keep an eye on the dashboard, we'll notice that the users can be divided into groups by certain shared characteristics, such as the screen resolution they use or from what operating system they are browsing our site.

If we segment the results we can have a better understanding of why the new version performed better (or worse), and we can discover new opportunities for experimentation in the next wave of experiments.

In results that are flat or negative, perhaps we'll identify what area prevented the experiment from otherwise producing a winner. We can point some common ways to segment a experiment:

- **New users / repeating users:** New and repeating users, usually have different behaviors while browsing our site, as they have different expectations, experiences, and intentions when visiting the site. So we can find opportunities to customize the experience for the new or repeating users.

- **Device:** If you were observing the devices used by your user, for sure you seen that the use of mobile devices has grown dramatically in the last years, so, in the same way it's increasingly important to break down the results by the top devices your visitors use in order to detect flaws of design and opportunities for customization.

- **Channel Break down:** It's important to know where the traffic is coming to our site, and to divide the results by channel, this way we can understand how the experiment performed by paid (Google pay-per-click ads), natural (search engine listings), direct (visited your website by typing the URL into the browser), affiliate, or other (social media like Facebook or Twitter) traffic. We can find that our experiment is working for a certain channel, but not for the others.

- **Browser:** It's very common that a site could have compatibility issues with some browsers (even with certain versions of a browser). So if we break down our results by browser type and version we can discover that flaws.

- **Time/Day/Season:** The experiments always have different performances depending on the dat of the week, the month or even the season. For example, in our case we have a lot more traffic during the months were the tuitions are being paid, so in that months our experiments perform different than in the “normal” months.

- **Product:** A experiment can work for certain products but not for others, so it’s important too to break down the results by product in order to find those differences.

4. peerTransfer Smarty project

I consider this chapter the most important of this document as I'm going to explain the Smarty project, which was the main motivation to write this document.

I'm going to explain the context that conducted us to develop a system with the main purpose of give some light to our data, and to make easier the further decisions that we will take, and also for minimize the impact of our errors.

Also we're going to see the architecture of the solution and the flow of the data inside it, explaining the most important design decisions and also the technologies involved.

We followed most of the principles written in this document but also we learned by trial and error as most us were unexperienced in this area. So I´m going to explain the success but also the failures.

4.1 Project Smarty requirements

In this chapter we are going point the set of requirements that we defined for the Smarty project, some of them were defined before we started the project, but most of them rest arisen while we were developing the project and after the first version was shipped.

This is a very common situation in our day to day, we never spent many time defining requirements, as we all know that is impossible to define beforehand the entire set of requirements, and probably those requirements will be wrong.

We prefer to define a very basic set of requirements, and start the project with a **MVP**[12] guiding the product through real feedback. In our case, this was easier as the project Smarty is a internal project that will be used only by peerTransfer employees.

When we shipped the first MVP, the people were so excited about it, and gave us a lot of excellent ideas and invaluable feedback for improving the product.

[12] *“A Minimum Viable Product is that product which has just those features (and no more) that allows to ship a product that resonates with early adopters; some of whom will buy the product or give feedback in order to improve the product.”* [LS11]

Those ideas, and the requirements we set at the beginning, were converted in the features that conform the Smarty project**:**

- **Data storing and collecting**: First of all, we should collect all the data we need inside our applications, this data can be events of any kind, and also raw data, for example metadata from http requests. The data collected is stored inside a external service, which basically consists on a non relational database and a dashboard for configuration purposes.

- **Rollout of features**: Some types of test require to activate a feature, but only for a certain type of user, or maybe only for a certain period of time, for this kind of situations we need to conditionally activate and deactivate features.

- **Implement the logic for the A/B experiments:** The logic for the A/B experiments lives inside the code of our application, we used Split, which is a simple library in Ruby for implementing A/B tests.

- **Provide an interface for showing the experiments results:** The results of the experiments are stored on our databases, there is no problem for the technical people in the company for accessing those results, but we need a way to show the results for the non-technical people (i.e marketing or product departments), also the results have to be shown from the beginning of the experiment until the end.

- **Creating reports and publishing them (Analyze results):** Collecting thousands of data is not enough if we want to extract some conclusions about how our experiments are performing and the status of our key metrics. This case is similar to the interface for defining A/B experiments, the people in charge of analyzing the result are not used to technology as a tech department is, so we need a easy and powerful tool for manipulate, analyze and extract conclusions of our data.

- **Provide a set of configurable alerts:** We have to provide some sort of alerting system for the platform, for example we can set an email alarm if certain experiment is going extremely bad, or if some metric reach a specific level.

4.2 Technology and design decisions.

In this chapter I'm going to present the technology involved on each requirement, and the design decisions we took in order to build all the infrastructure needed for the Smarty project.

Data collecting and storing

We can divide the data collecting into two parts, one is in charge of collecting the defined events inside our applications and the other writes to a database for reporting purposes, this database has one table for each report, the data for this reports comes from different databases, in the next chapter we will explain how the synchronization works.

For the first part, on the first stages of the project our intention was to develop entirely our own event collecting system, as we underestimated the complexity of a system of this kind. This complexity comes, as in order to gather the data we need some mechanism to send the events that occurs in our systems and some place to store and classify them.

The problem is that if you have certain level of traffic and some experiments running at the same time, the computational requirements for storing and classifying the data can be pretty high, so if you want to own and maintain those system with high computational capabilities you need to invest time, money and many man hours just to keep the systems running.

This was not an option as we are a small team, and we prefer to focus our time developing new features rather than maintaining infrastructure. So we decided to use a third party solution for the storing and classification of the data, and develop our own system for collecting and sending all the events.

The peerTransfer platform is composed by many pieces in different applications that work together, as we want to gather live data from all the applications that we have running, we need a centralized way to do this. So we put this logic in our core application, that is between most of the interactions between the different applications.

Essentially the data that we want to collect are events, for example we want to capture which version of the experiment was presented to a participant, this kind of information is stored inside an event that we trigger when we present the particular version to the participant. We defined that an event has the following properties:

- **Name:** A unique name inside the experiment in order to differentiate it from the rest of events.

- **Properties:** Different attributes of the event, for example the name, address or country of the user.

- **Experiments:** A list of the experiments where the event takes part. This is important as an event can be part of more than one experiments. Which makes the events a reusable component.

Data storing and classifying

As we noted before, for this part we used a third party solution, at the beginning we tried **Mortar Data**[13], which is a very powerful service for providing data infrastructure in a very easy way.Mortar is built on **Hadoop**[14] and **Apache Pig**[15]**.** Pig is great because it is easy to learn (it has primitives that are similar to SQL), is highly expressive, and works well with other technologies.

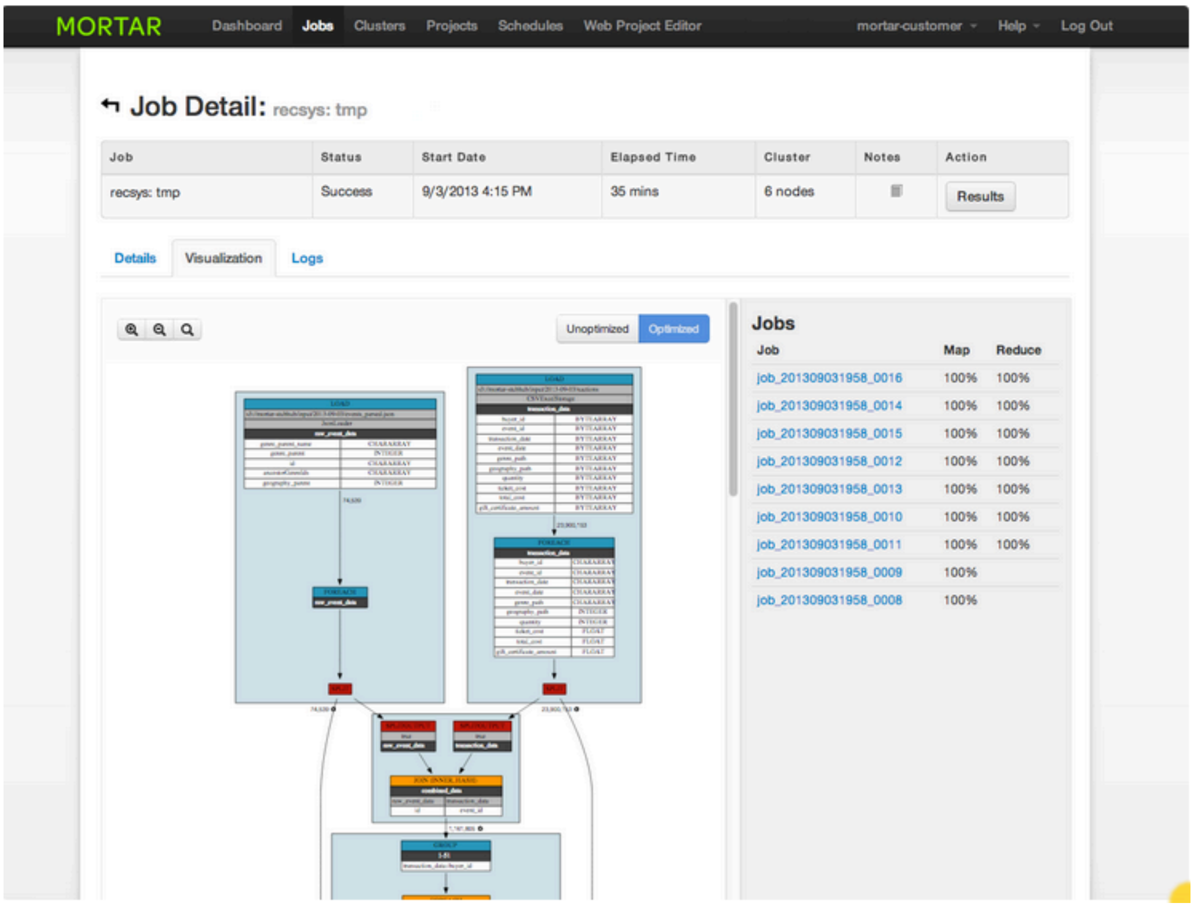
[13] http://www.mortardata.com/

[14] Apache **Hadoop** is *"an open source software project that enables the distributed processing of large data sets across clusters of commodity servers"* [IBM13]

[15] **Apache Pig** is "*a platform for analyzing large data sets that consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs.*" [AP13]

But more importantly, Pig is the standard for data processing at scale. It is used by hundreds of companies including LinkedIn, Netflix, and Yahoo. That means stability, maturity, and longevity.

Mortar is extremely powerful, but we felt that we were cracking nuts with a hammer as Mortar, although simplifies a lot the process of providing data infrastructure, was still too big and complex for our project, so we decided to abandon it and try a simpler solution.



Job Detail: recsys: tmp

Job	Status	Start Date	Elapsed Time	Cluster	Notes	Action
recsys: tmp	Success	9/3/2013 4:15 PM	35 mins	6 nodes		Results

Jobs

Job	Map	Reduce
job_201309031958_0016	100%	100%
job_201309031958_0014	100%	100%
job_201309031958_0015	100%	100%
job_201309031958_0012	100%	100%
job_201309031958_0013	100%	100%
job_201309031958_0010	100%	100%
job_201309031958_0011	100%	100%
job_201309031958_0009	100%	
job_201309031958_0008	100%	

Figure 4.1, Mortar Data

This simpler solution was **KeenIO**[16]**,** which is a software as a service specially suited for developing custom analytics & data science features. KeenIO essentially offers an analytics db, but also provides the infrastructure and APIs to collect the data and build the analytics, as we can see on the Figure 4.2.

The typical use case of KeenIO is exactly what we were looking for, as we wanted to make easier the process of storing data, but also we need an API in order to interconnect this database with the analytics tools, KeenIO provide this and more.

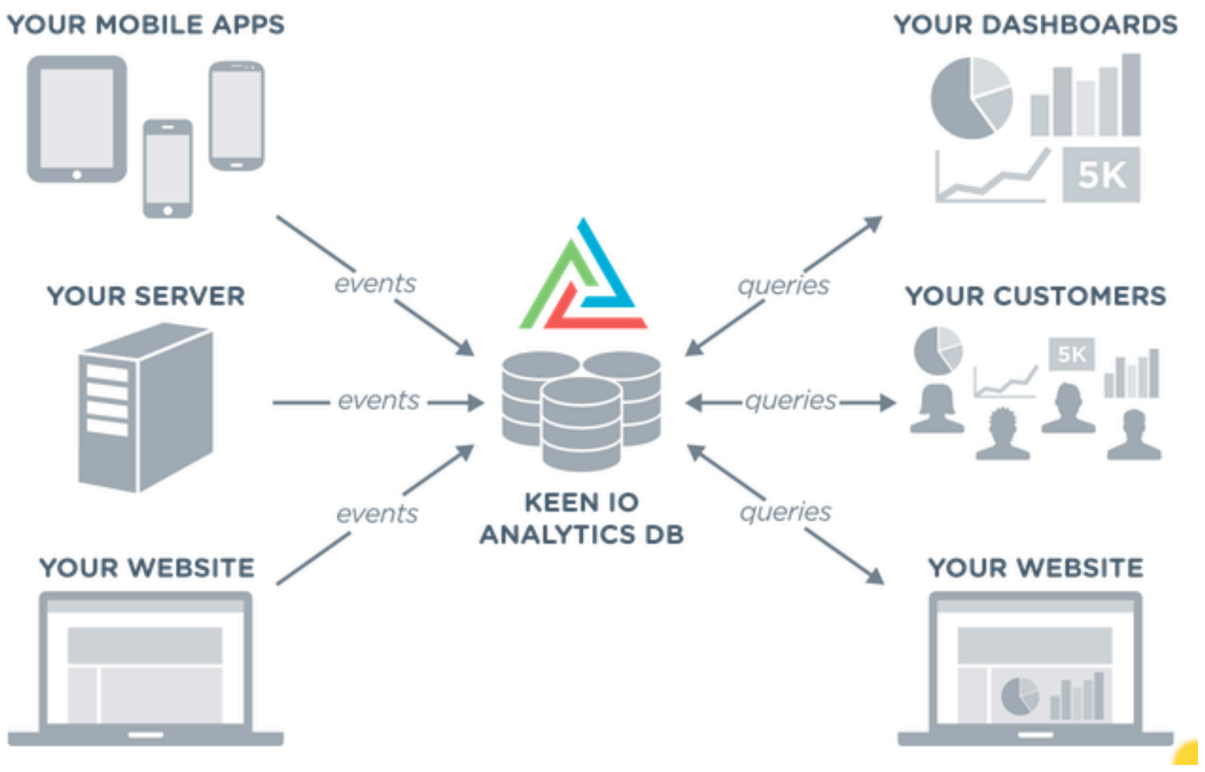
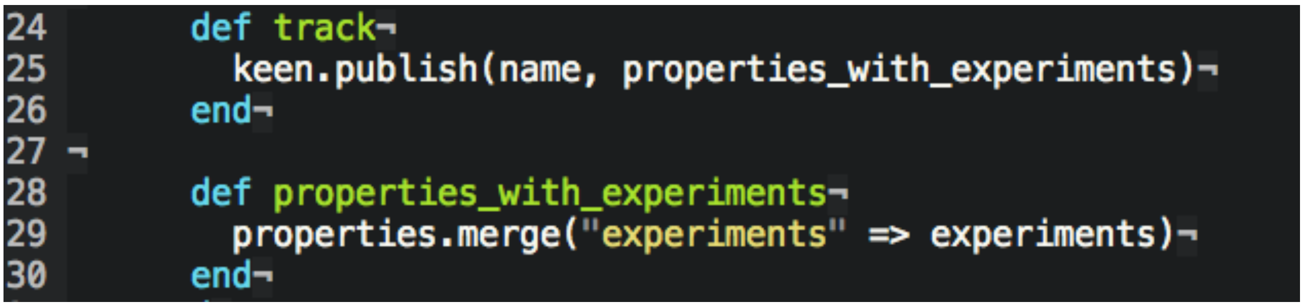


Figure 4.2, KeenIO

16 https://keen.io/home

With KeenIO, as we can see on Figure 4.3, it's very easy to send the events, the only thing we had to do, is use their client inside our code and that's it:

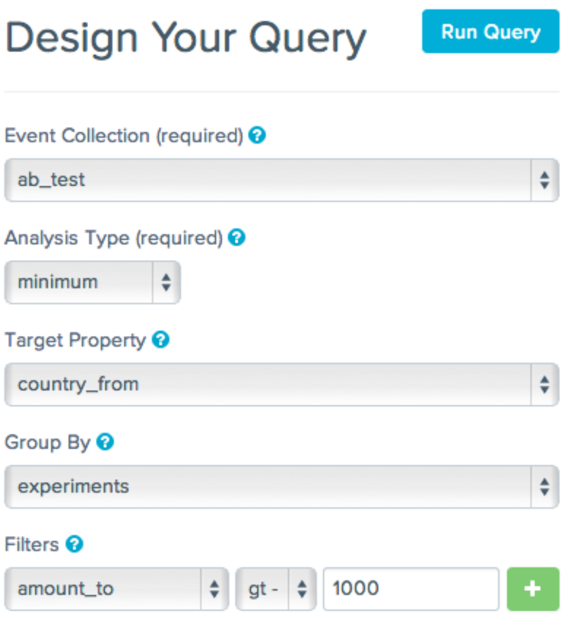


```
      def track
        keen.publish(name, properties_with_experiments)
      end

      def properties_with_experiments
        properties.merge("experiments" => experiments)
      end
```

Figure 4.3, Sending events to KeenIO

With only a few lines of code we are sending events across our applications. KeenIO stores the events, and make them available through their API, so we can access that classified data through analytics applications. We can see with more detail the implementation on the Appendix A.

The classification of the events is very easy to configure as we only have to define the name of the event, and this name acts as unique id. The interaction with KeenIO is mainly made through the API, but as we can see on the Figure 4.4 we can also use the workbench for run queries and save them:


Design Your Query

Run Query

Event Collection (required): ab_test

Analysis Type (required): minimum

Target Property: country_from

Group By: experiments

Filters: amount_to gt - 1000

Figure 4.4, Designing queries on KeenIO

With a service like this, we are saving a lot of time configuring, maintaining and developing systems, so, at the end although we have to pay KeenIO for their service, we are saving money.

Rollout of features

As we noted before, the rollout of features consist on conditionally activate or deactivate features, we implemented this feature with **Rollout**[17], which is a very simple Ruby library with this purpose that uses Redis[18] for persisting the configuration.

[17] https://github.com/bitlove/rollout

[18] Redis "*is an open source, BSD licensed, advanced **key-value store**. It is often referred to as a **data structure server** since keys can contain strings, hashes, lists, sets and sorted sets.*" [RD13]

For example, we can activate a feature or deactivate a feature for all the users like this:

```
$rollout.activate_group(:chat, :all)
```

```
$rollout.deactivate_group(:chat, :all)
```

In this case, ***:all*** refers to the group "All", which contains all the users, of course we can activate or deactivate those features not for all, but for a certain group of users that we configured, this can be for example a group consisting of early adopters of the applications or VIP users.

Also we can activate or deactivate a feature just for a single user like this:

```
$rollout.activate_user(:chat, @user)
```

Also we can use more advanced features, for example we if we're rolling out a new feature, we might want to test this new feature by slowly enabling it for a percentage of your users in order to minimize the possible negative impact of this new feature:

```
$rollout.activate_percentage(:chat, 20)
```

The algorithm that chooses if a user is contained or not in the set works like this:

```
user.id % 10 < percentage / 10
```

With this capabilities, we can conditionally enable or disable features automatically. We uses this capabilities for the following use cases:

- As an emergency brake, if an experiment goes really wrong we can deactivate the feature.
- For activating a experiment for a certain group of users.
- For gradually activate a feature, for example we can check how the feature is performing and grow or decrease the number of participants if the feature is performing well or not.

Logic for the A/B experiments

For this side of the project we tried some approaches, at the beginning we decided to look for a third party service in order to minimize the launch time of the project. We decided to give a try to **Optimizely**[19], Optimizely is a powerful AB testing tool meant to be used by non technical people. This tool is very powerful for making changes in the static elements of the page (i.e typographies, colors, sizes and positions of the elements)

Optimizely provides an integrated editor which is very easy to use, with this editor we can design the experiments in a visual manner, without editing the code, this feature makes possible the design of A/B experiments by non technical people.

[19] https://www.optimizely.com/

We can illustrate better this concept with an example, imagine that we want to do an experiment, this experiment consists on two versions, the original one and other with some changes in the position of elements and the background picture on the peerTransfer site, the goal is to improve the conversion rate.

If we want to do this with Optimizely, as we can see on Figure. 4.5, we simply have to enter to the editor, and add a new variation with the required changes:

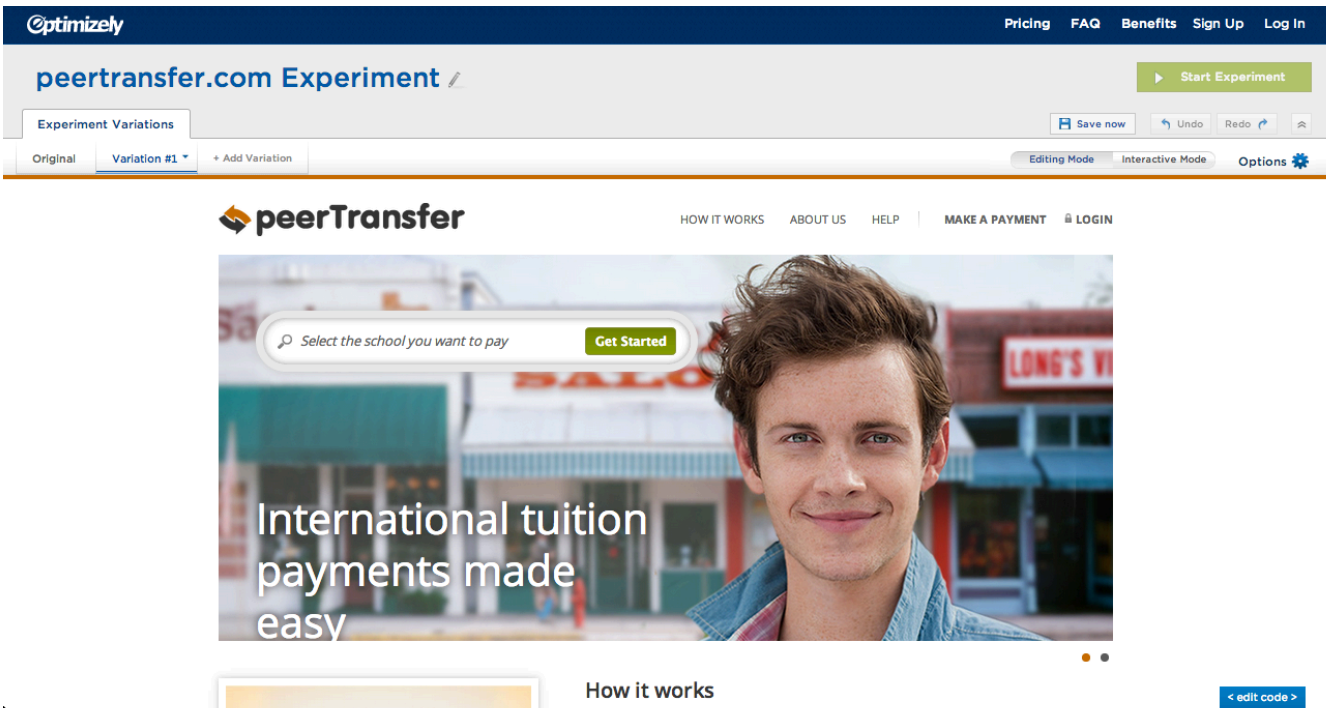


Figure 4.5, Adding a new variation on Optimizely

We select the elements that we want to change in size or position:

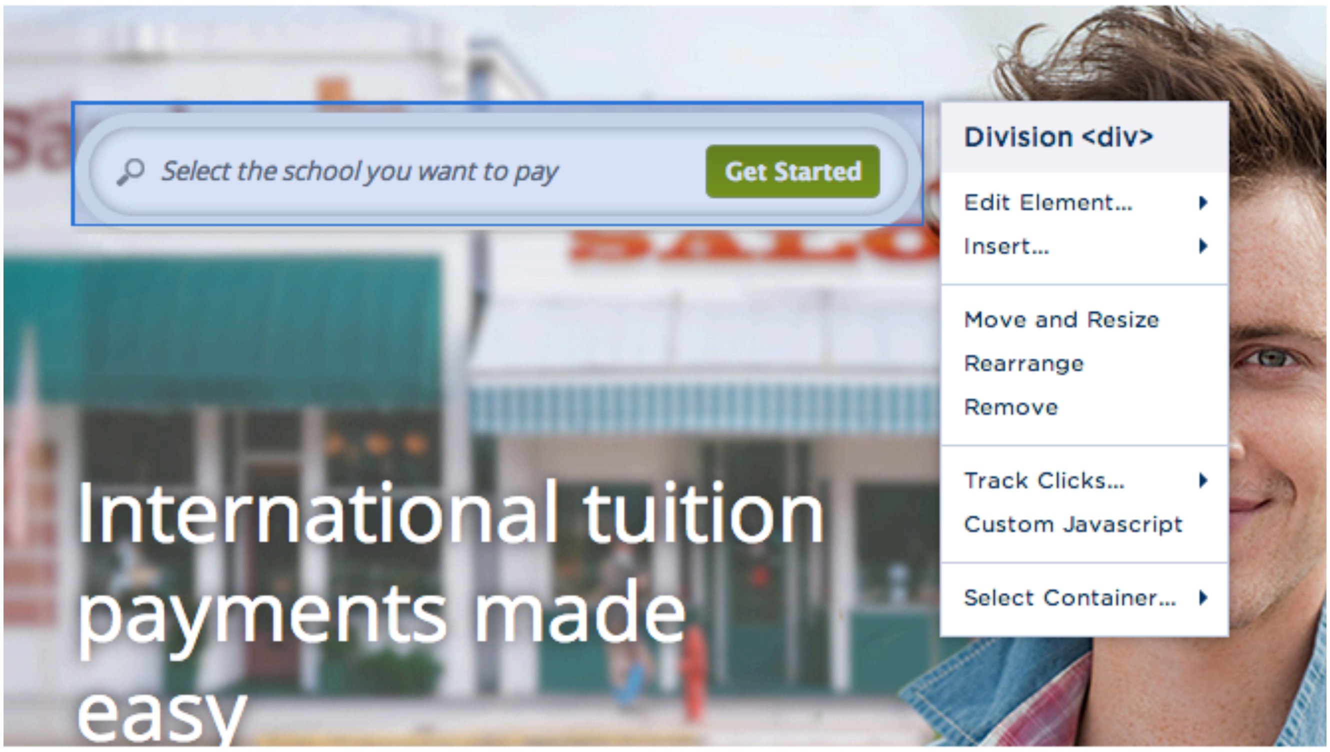


Figure 4.6, Modifying an element on Optimizely

After this, we simply have to drag and resize the elements to the desired position:

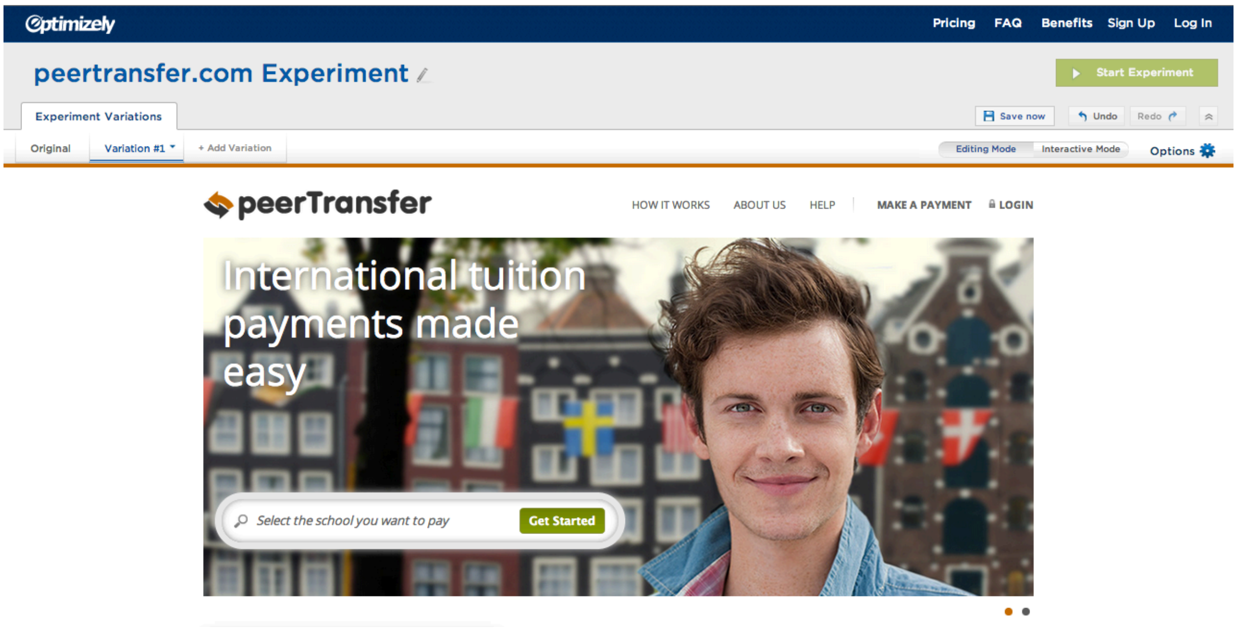


Figure 4.7, Moving an element on Optimizely

Finally we only have to change the background picture:

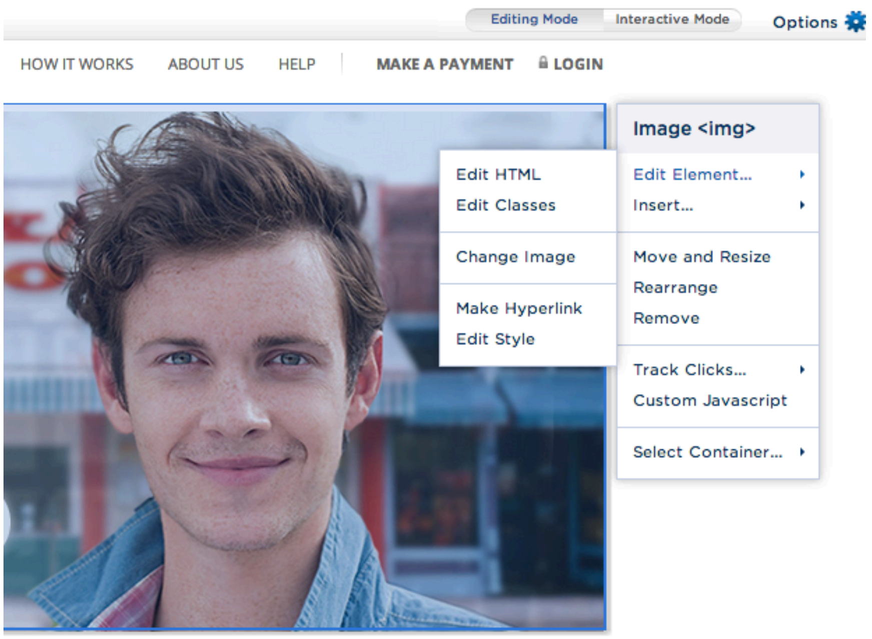


Figure 4.8, Changing the background on Optimizely

And with just a few clicks, we changed notably the design of our homepage:

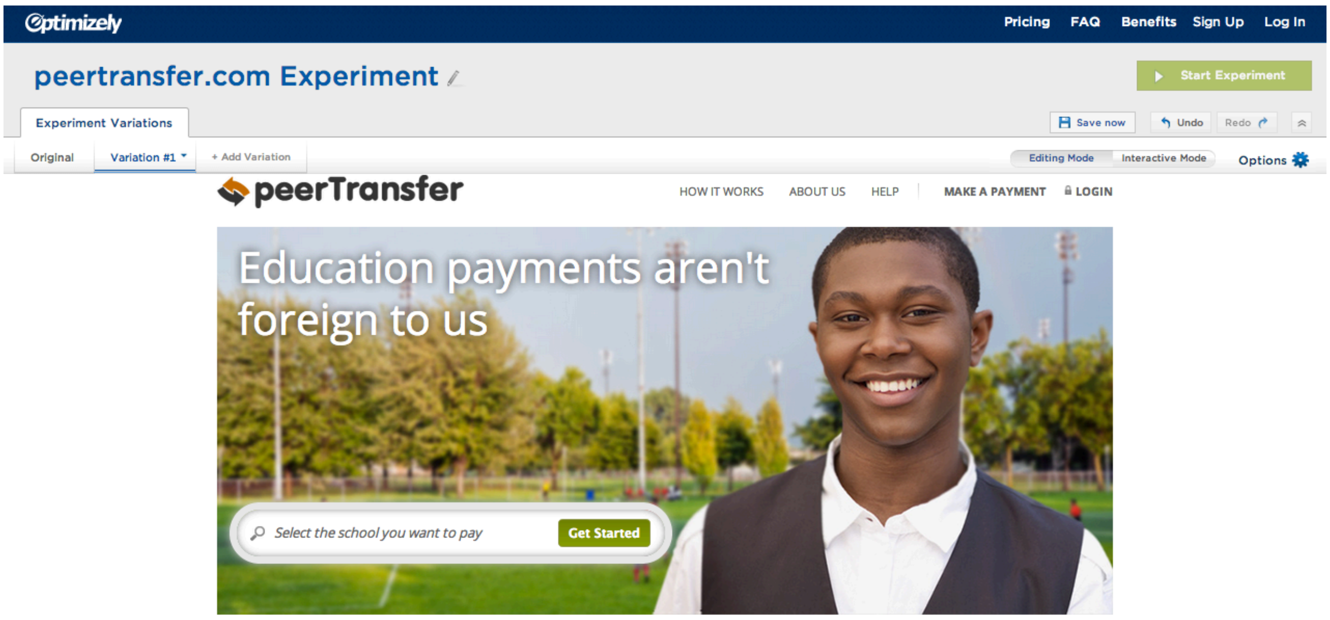


Figure 4.9, Final result of the changes made with Optimizely,

With the new variation created, we have to set a goal, Optimizely let's you use three different kind of goals, which are:

- **Click goals:** The participant clicks to a specific element of the site.

- **URL goals:** The participant reach a certain URL inside the site, for example we can set that the confirmation page is the goal as it is showed after the user completes the transaction.

- **Custom Event goals:** Also we can define custom events, for example, we can define that the participant submits an AJAX form on a page, which doesn't generate a new page view, the participant completes half of a form, the participant visits one of several different possible conversion pages, etc.

On the last example, if we want set that the participant clicks on the "*Get started*" button, we simply have to click on the element and select it as Click Goal, as we can see on Figure 4.10

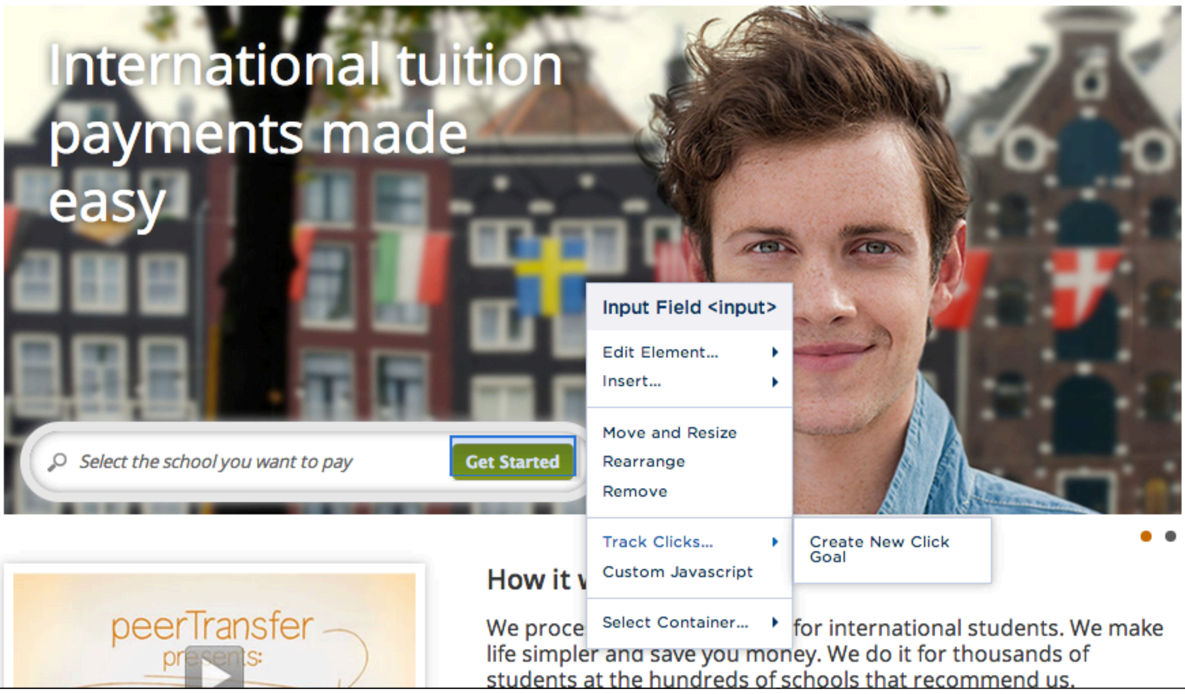
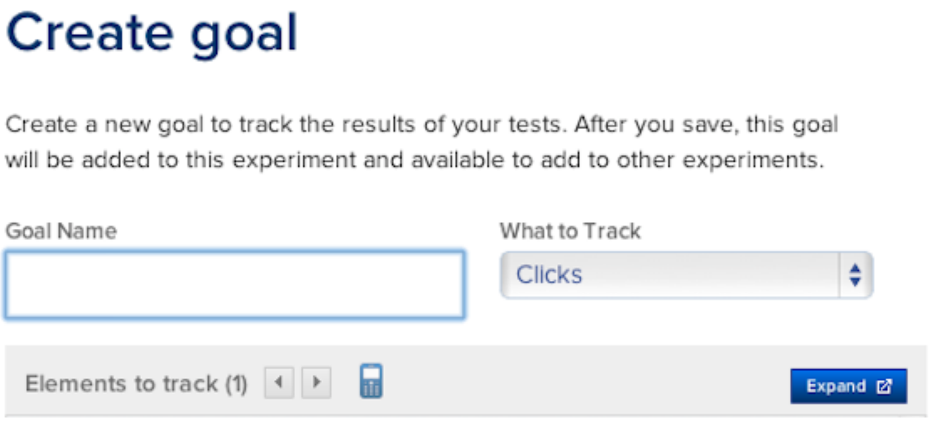

International tuition payments made easy
Select the school you want to pay
Get Started
Input Field <input>
Edit Element...
Insert...
Move and Resize
Rearrange
Remove
Track Clicks...
Custom Javascript
Select Container...
Create New Click Goal
peerTransfer presents:
How it w
We proce for international students. We make life simpler and save you money. We do it for thousands of students at the hundreds of schools that recommend us.

Figure 4.10, Setting a new click goal


Create goal

Create a new goal to track the results of your tests. After you save, this goal will be added to this experiment and available to add to other experiments.

Goal Name

What to Track: Clicks

Elements to track (1)

Expand

Figure 4.11, Editing the fields for the new click goal,

With this new defined goal, we are able to launch the experiment, Optimizely also provides the infrastructure for the A/B testing itself, so, when we create a new variation and the experiment is started, the traffic is splitted between the variations and the original without any change in our code, also it stores the results gives an interactive tool in order to analyze the results.

Although Optimizely is great, the problem for us was that we needed more advanced features, not just experiments based on appearance. For example, we wanted to test what pricing options improve more the conversion rate for certain countries, also we wanted to know how we have to price the currency conversion rate for the new currencies that we had in that moment.

This kind of experiments implies not only changes on the appearance, but also in the code that makes the conversion itself. With this two experiments we realized that Optimizely was not the tool we needed, so we decided to design our own A/B testing toolset.

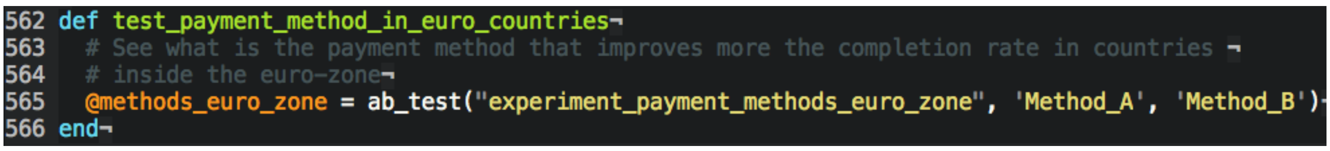
As peerTransfer's platform is coded on Ruby on Rails, we searched for A/B testing libraries and quickly we found **Split**[20]. Split is a very easy to use Ruby framework for doing A/B experiments, it's customizable and very powerful.

Split works on top of Rack, which is the standard web server interface for Ruby on Rails and also, like Rollout it uses Redis as datastore for the configuration, so was very easy to combine both libraries in our code.

So, after installing the library in our codebase, we can start doing experiments. We can see how Split works with the previous example:

20 https://github.com/andrew/split

If we want to experiment on which payment method we have to offer in countries inside the euro-zone, in order to improve the conversion rate, we can see how to do this in the Figure 4.12



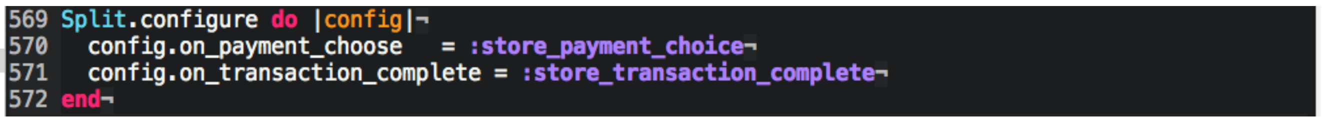
```
def test_payment_method_in_euro_countries
  # See what is the payment method that improves more the completion rate in countries
  # inside the euro-zone
  @methods_euro_zone = ab_test("experiment_payment_methods_euro_zone", 'Method_A', 'Method_B')
end
```

Figure 4.12, Test payment method in euro countries example,

So, Split will “split” the traffic inside our application, this is possible as Split works on top of Rack. It will assign one value of the values we provided, which are “*Method_A*”, and “*Method_B”* to the “*methods_euro_zone”* variable. Also it will store other data (user_id, ip, etc) about the participants.

So, as we have the data from the participants stored, **we have to define one event that will be the goal of the experiment**, for the last example, the goal is the transaction completion, in other words, our goal is that the user completes the transaction as we believe that this highly depends on the payment method, as each payment method have a different price and a different process, that can be easier or harder for the user, so this conditions can determine if the user completes or not the transaction, guiding his decision for example by the conversion rate that we apply, or by how simple is the process of payment.

In this case, as we said, we have to trigger the completion event when the user completes the transaction and also we have to define an event, that will be triggered when the participant starts the experiment, and it will store in which version the user participated. This is required as we also want to know how many participants doesn't finish the transaction. As we can see on Figure 4.3, we can configure those events like this:



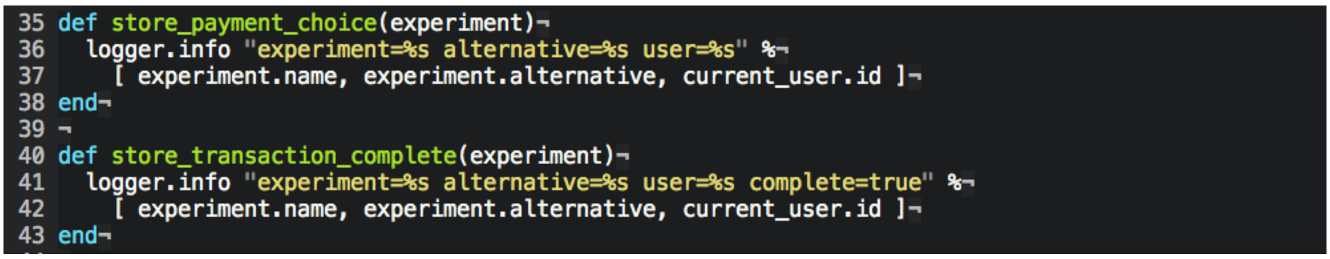
```ruby
Split.configure do |config|
  config.on_payment_choose    = :store_payment_choice
  config.on_transaction_complete = :store_transaction_complete
end
```

Figure 4.13, Events configuration on Split

The value for each config variable will be the method that will be triggered for each event, the "*store_payment_choose*" method corresponds with the event triggered when participant starts the experiment, and the "*store_transaction_complete*" method corresponds with the transaction completion event.

After this, we only have to define what method will be triggered on each event. In our case "*store_payment_choose*" will simply store the experiment version on the participant dataset, and "*store_transaction_complete*" will store a true value on the transaction completion variable inside the dataset of the participant.

We can see how this methods are implemented on the Figure 4.14



```ruby
def store_payment_choice(experiment)
  logger.info "experiment=%s alternative=%s user=%s" %
    [ experiment.name, experiment.alternative, current_user.id ]
end

def store_transaction_complete(experiment)
  logger.info "experiment=%s alternative=%s user=%s complete=true" %
    [ experiment.name, experiment.alternative, current_user.id ]
end
```

Figure 4.14, Triggered methods examples

So with this data stored, we know how much participants we had on each version, and how much participants completed the transaction, this way we can extract the completion rate for each version and declare a winner after the experiment finishes and the results are analyzed.

After this data is stored, we implemented a hook that sends to KeenIO the info related to the participant of the experiment, which is:

- **Event value:** this is always a binary variable, in this case is the transaction completion or not by the user.
- **Experiment:** The name of the experiment.
- **Data about the session:** This is used for the analytic tools and it's different on each experiment, for example in some experiments we can be interested on gather geographical data and in others we can be interested on the age and the type of degree of the user.

So with the data stored in KeenIO we are able to start analyzing the results with our analytic tools and making conclusions about which version is the winner of the experiment.

Analysis of the experiment results

With the results of how much participants of each version we had in an experiment, and how much of them completed the transaction, the only remaining thing is to know if there is a statistical difference on the results of each variation in order to finish the experiment.

This is a very important question, as we need to have an empiric way to know if there is such difference between the results of each version. The importance of this comes because many times the results of an experiment are not strictly different than the results of the original version, so there is no statistical difference, and we can't conclude which version performed better.

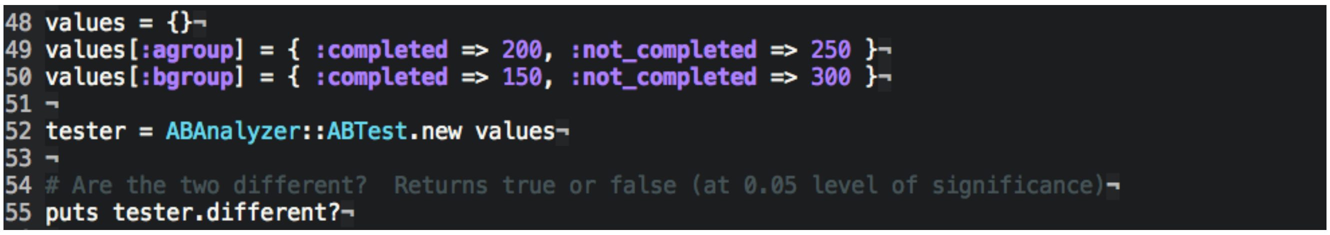
In this kind of cases, as we assume that the variations performed mostly the same than the original, we keep the original version and try another experiment with other variations in order to achieve better results.

In order to know such statistical difference between the results of each variation, we used **ABanalyzer**[21]**,** which is a Ruby on Rails library that is able to perform the needed calculations to determine this difference between each variation of an A/B test.

[21] https://github.com/livingsocial/abanalyzer

By default, it uses a **G-Test for independence**[22], but also a **Chi-Square test for independence**[23] can be used. We decide to use ABanalyzer, as it´s very simple and mature and it has been used in multiple projects with good results.

As we said, with ABanalyzer we can calculate the statistical difference very easily, as we can see on Figure 4.15, we only have to instantiate an ABTest object with the values we obtained in the A/B experiment, and call the "*different?"* method in order to get the results:



```
values = {}¬
values[:agroup] = { :completed => 200, :not_completed => 250 }¬
values[:bgroup] = { :completed => 150, :not_completed => 300 }¬
¬
tester = ABAnalyzer::ABTest.new values¬
¬
# Are the two different?  Returns true or false (at 0.05 level of significance)¬
puts tester.different?¬
```

Figure 4.15, Calculating statistical difference with ABanalyzer

This method returns true of false depending if the values are different or not, if the values are different and we have enough examples we can finish the experiment and declare a winner.

[22] A G-test for independence, "*compares frequencies of one attribute variable for different values of a second attribute variable*". [GI06]

[23] The Chi-square test "is *intended to test how likely it is that an observed distribution is due to chance, in other words it measures how well the observed distribution of data fits with the distribution that is expected if the variables are independent"* [PC08]

Showing the results

For this part of the project we simply added a new section for showing the provisional experiment results in our Lounge, which is a sort of internal social network, but also holds information about our key metrics, a section for proposing ideas for improving our company, etc. This internal application is available for every peerTransfer employee.

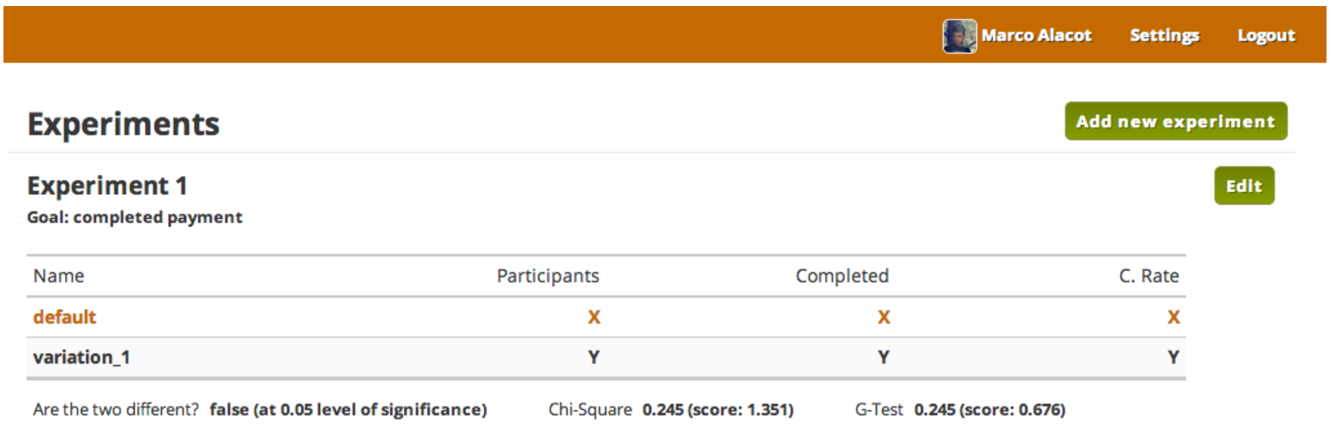
This is very important as we want to inform not only the people related with the experiments itself, but also all the company, as we believe that every employee can participate on the improvement of the company itself.

We inform on what experiments we're trying, and how this experiments are going, also we can edit the basic attributes of the experiments or even remove the experiment itself on the fly through the Lounge.

This is very useful, as many times we make mistakes when we are creating the experiments, so this way is very easy to change the wrong attributes, even by non technical people. Also sometimes we make changes just because the requirements changed.

The experiment results are crawled directly from ABanalyzer and published on the site in near real time.

The Figure 4.16 shows how the experiments results are presented on the Lounge:



Experiments

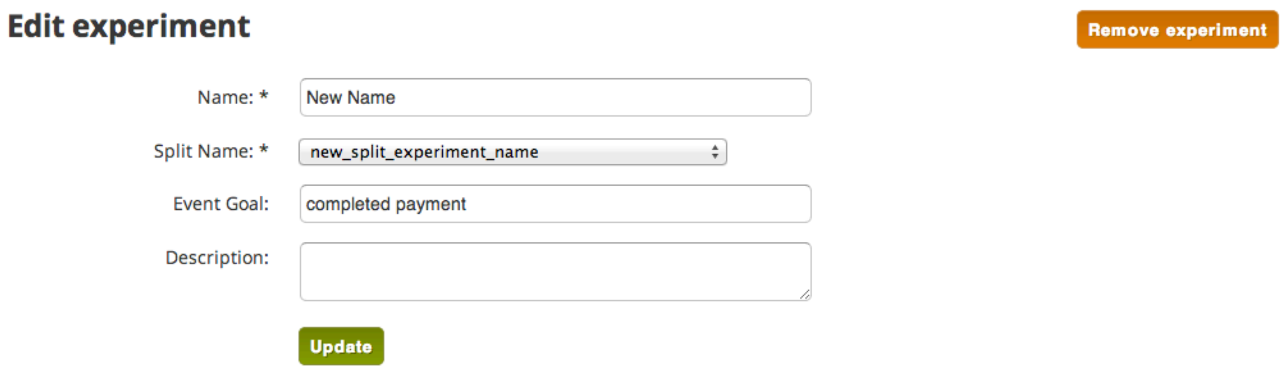
Experiment 1
Goal: completed payment

Name	Participants	Completed	C. Rate
default	X	X	X
variation_1	Y	Y	Y

Are the two different? **false (at 0.05 level of significance)** Chi-Square **0.245 (score: 1.351)** G-Test **0.245 (score: 0.676)**

Figure 4.16, Experiment stats in the Lounge application

The Figure 4.17 shows how we can edit the basic info of an experiment:



Edit experiment

Name: *	New Name
Split Name: *	new_split_experiment_name
Event Goal:	completed payment
Description:	

Figure 4.17, Editing information of an experiment

Alarms

As it’s impossible to know beforehand how the variations of an experiment will perform, sometimes we try experiments that perform much worse than the original version, also we can make a mistake on the design or the implementation of the variation, and the variation itself is not working properly.

In order to minimize the impact of those experiments in our key metrics, we check the provisional results of the experiment, and if the results are below a defined limit we send an email to the support team for inform about this. This alert is triggered when we gather a batch of results from KeenIO, and the mail is sent using the Ruby Mailer class.

The support team will evaluate the alert, and it will consult the people involved on the experiment in order to stop the experiment or fix the problem with the variation.

Web Analytics, reporting

The reports and the analysis of the data from the site, plays a central role on the definition phase of the experiments.

Without a good analysis we will be taking shoots in the dark when designing new experiments, as we don´t know how is the status of our metrics, and even which metrics we need to improve more. Also with the data analysis, we can see correlations between the metrics, and cause of this, we can define new experiments in order to improve them.

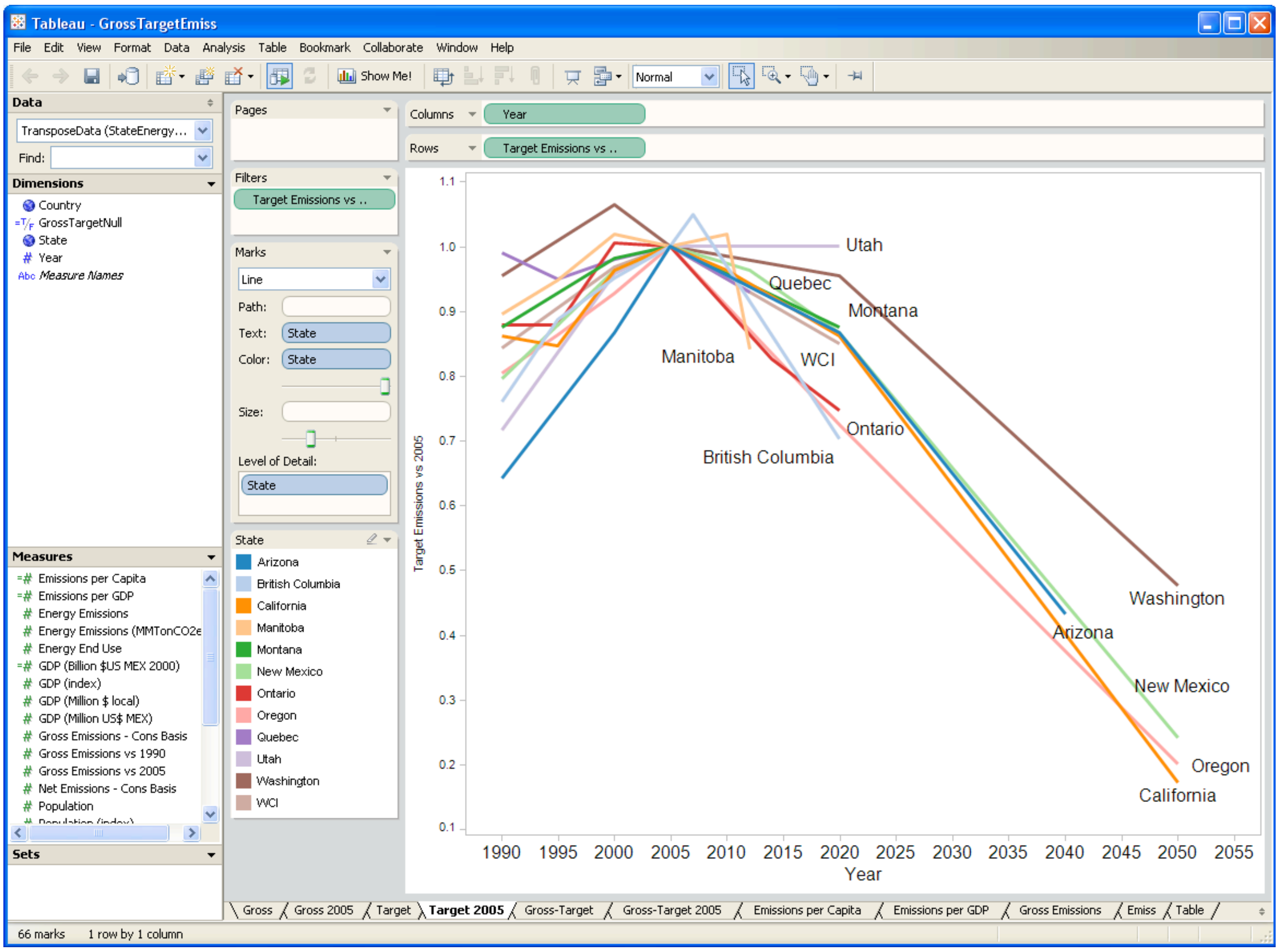


Figure 4.18, Example report with Tableau

This process is constant, we are always analyzing the data, and making conclusions about what features are not performing well, and what we should do in order to change that situation.

For this part of the project, we used a third party application called **Tableau**, which is an extremely powerful application, but despite of that, is easy to use as it can be used for anyone, whenever haves technical skills or not. In fact, in our company the people in charge of the data analysis don´t have technical skills. In the Figure 4.18 we can see and example report with fake data:

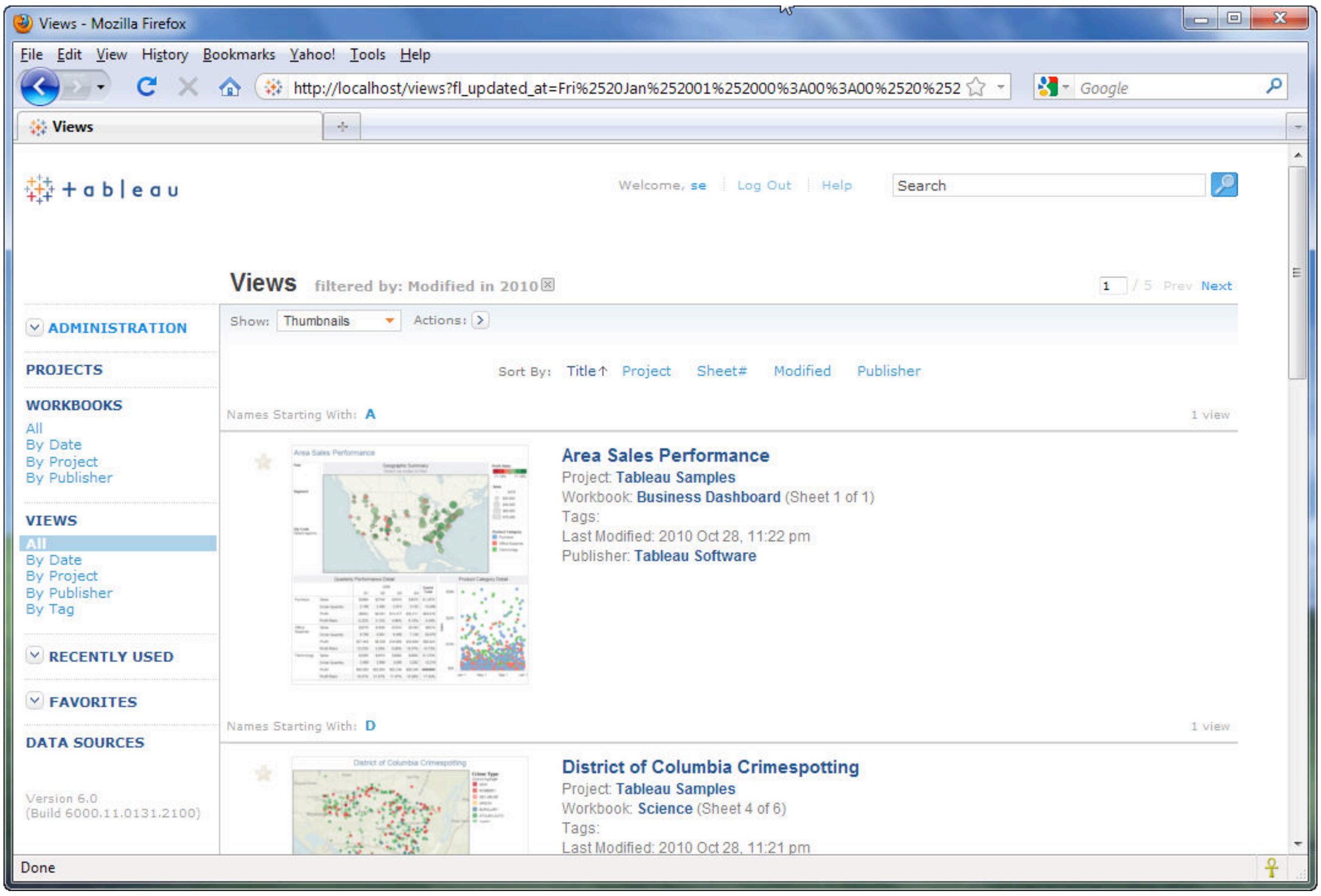
Tableau is deployed inside our infrastructure, and gathers the data directly from our databases, this way we don´t have to do any synchronization process or a special care about the consistency of the data as we are already controlling that in other parts of our infrastructure.

We use Tableau also for reporting purposes, we understand the reports as a snapshot of a specific query into our data, but also we need that reports have to be updated in realtime with the new data.

For instance we may have a report that calculates the revenue, this report have to be updated automatically without constantly executing the query manually in order to get the new data. The calculations in the report are instantly recalculated for every transaction we make automatically, that way we can show reports with real data in real time.

Tableau also acts as a server, and gives the possibility of publishing the reports in a website, so we can create and store the reports in Tableau, and then publish them in the Lounge[24]. In the Figure 4.19 we can see a Tableau server view inside a website:

[24] Unfortunately, for compliance and confidentiality reasons, I´m not able to show real reports.



Views filtered by: Modified in 2010 | 1 / 5 Prev Next

Show: Thumbnails Actions:

Sort By: Title↑ Project Sheet# Modified Publisher

Names Starting With: A — 1 view

Area Sales Performance
Project: Tableau Samples
Workbook: Business Dashboard (Sheet 1 of 1)
Tags:
Last Modified: 2010 Oct 28, 11:22 pm
Publisher: Tableau Software

Names Starting With: D — 1 view

District of Columbia Crimespotting
Project: Tableau Samples
Workbook: Science (Sheet 4 of 6)
Tags:
Last Modified: 2010 Oct 28, 11:21 pm

Figure 4.19, Tableau server view

4.3 Architecture, flow

In this chapter I want to give a briefly description of the flow inside the architecture of the project in order to explain better how all the pieces we described on the last chapter work together.

The architecture of peerTransfer is big and complex, so the requirements and needs of a project of the size and importance of Smarty has to be complex. As we noted on the last chapter, Smarty is formed by many different pieces, those pieces are a mix of external services, and applications developed by us.

For a better understanding on how that pieces work together, we can see the flow inside the Smarty platform, starting at a user entering the site, and finishing when the experiment ends, the Figure 4.20 shows how is the flow inside of the Smarty platform:

If we go through the diagram we can resume the main events that occurs inside the Smarty platform, those events are:

1. A user enters the site and selects a flow inside the application that implies an experiment.

2. Rollout evaluates if the user is a candidate for the experiment, as we noted before this is not necessary for all the experiments, but in many of them we restrict what users can enter the experiment in order to select a more suitable set of participants of the experiment. We make that classification by geographical data, currency used, etc.

3. Splits randomly presents to the user one of the versions that we defined for the experiment.

4. Splits captures the events and store them in Redis, those events are fundamentally two, one for indicate on which version the user participated, and another for indicate if the participant completed the goal or not.

5. With a cron job[25] we send the stored events in batches to KeenIO, we make it that way in order to improve the performance, and in order to minimize and control the load of our core application, if we have a good amount of traffic in the site, and we are sending all the events on real time, we may increase the load too much.

6. KeenIO stores the events and classify each of them inside the corresponding experiment.

7. Using a daemon, we query the API of KeenIO to gather the results of the experiments, the strategy for gathering the data is the same than for sending the events to KeenIO

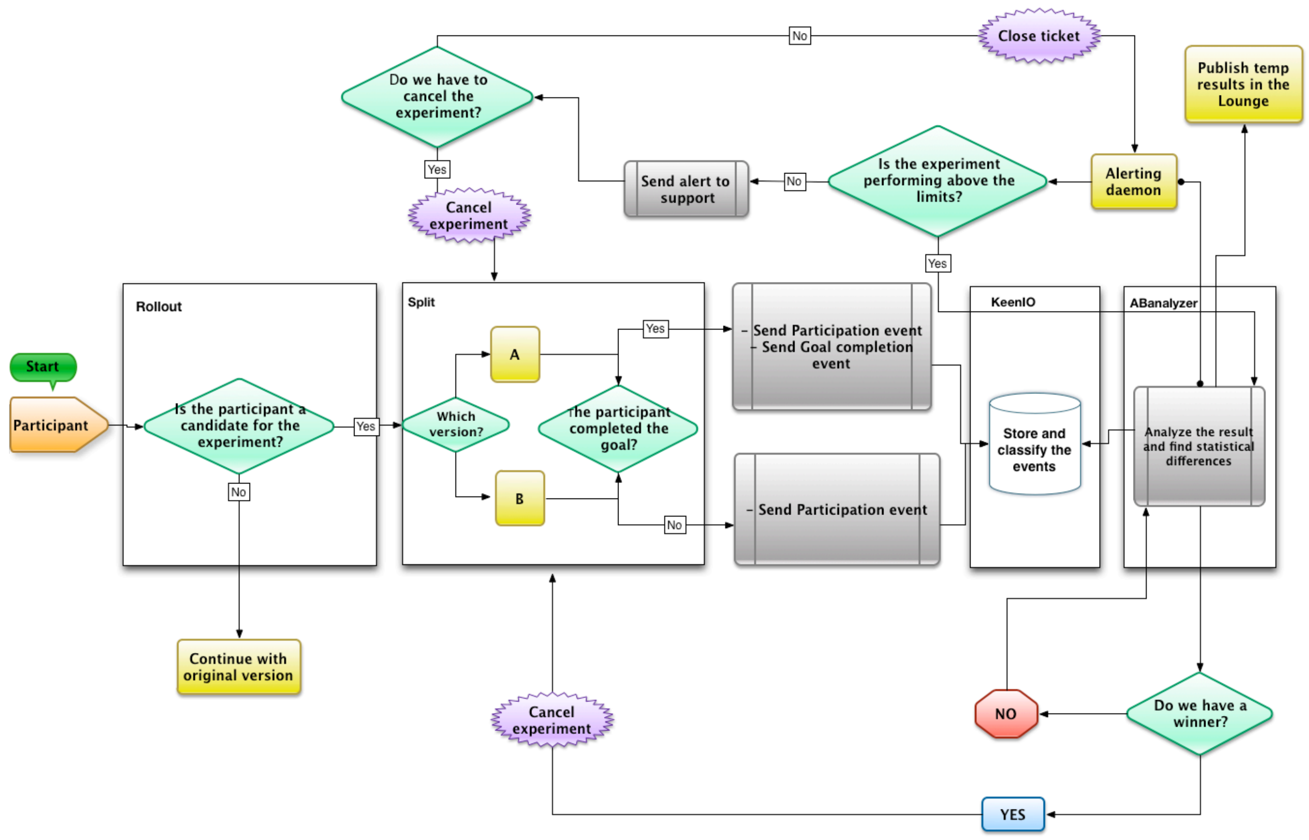
8. The gathered data is analyzed with ABanalyzer.

9. If the provisional results are performing much worse than the original version, we send an email alert to the support team, in order to check the results and even cancel the experiment if the results are really bad.

[25] A cron job is a long running process used for execute operations at certains periods of time.

10. The provisional experiment data is published in the Lounge

11. When we have enough participants we check the analysis of the result, and if there is a statistical difference we declare a winner, that can be the original or a new variation.



Close ticket
No
Do we have to cancel the experiment?
Yes
Cancel experiment
Send alert to support
No
Is the experiment performing above the limits?
Yes
Alerting daemon
Publish temp results in the Lounge
Rollout
Split
KeenIO
ABanalyzer
Start
Participant
Is the participant a candidate for the experiment?
Yes
No
Which version?
A
B
The participant completed the goal?
Yes
No
- Send Participation event
- Send Goal completion event
- Send Participation event
Store and classify the events
Analyze the result and find statistical differences
Continue with original version
Cancel experiment
NO
Do we have a winner?
YES

Figure 4.20, Flow inside the Smarty platform

5. Conclusion, results and future steps

After more than a year with the Smarty project up and running we can say that the results have been very positive and the general impact on the company have been very deep. We improved many of the key metrics in our site, from the conversion to the bounce rate, and we learnt great amounts of invaluable information about our customers. Now we are smarter and our features are more aligned with the needs of the customers.

The importance of using Web-analytics, and introducing the experiments in a data-driven development process have been demonstrated, as the results we had at the moment following this patters have been very positive.

We improved the flow of our payment methods thanks to the experiments we did on that field, that way our customers can process their transfers easier and quicker. We seen that **the cancellation rate for some payment methods was reduced by more than a 50%**, this is a great result, and personally I'm sure that without the Smarty project would never have been possible.

Also we did experiments with our back-office applications in order to improve the usability and to minimize the errors, the improvements were even better than expected, as in this case, we had a very direct feedback from the operators, and they helped us with the design of the experiments. Now our internal applications are easier to use and less prone to errors.

Another interesting result, is that since we started the project, we slowed our product development process, although this can be seen as an inconvenient, for us is a total improvement, as this means that now we are thinking more than acting, and we are backing our decisions with data.

At the moment we are making less changes to the product than before, but we have an empiric way of knowing if those changes are performing better or worse than the original ones without impacting the product.

As we gave total visibility to the results, all the company is getting involved on providing ideas for new experiments, and consequently in how to improve our product and the company. Also with the reports we all have a better sense of teamwork, as we can see everyday how our hard work improves our key indicators.

Problems

Although, the results have been very positive, we also encountered some problems, for example the alerting system for the features that perform below a limit was not present from the very first day, so we made a mistake on one experiment about a new pricing method, that mistake made impossible to pay with this method.

We noticed that the payment method wasn't running correctly and we stopped the experiment, but was too late as the experiment was running for about one day and nobody couldn’t pay with that method during that time. Cause of this we implemented the alerting daemon in order to avoid this kind of situations.

Also we made some implementation errors, one of them was with the logic concerning the events, this caused that many events were counted two times, giving us wrong conclusions about the result of some experiments.

Future steps

We have two main focus on improving the Smarty platform, one of them is concerning the automation of some processes in the creation of the events. We are working in a way of implementing the simpler experiments (the experiments that only test appearance changes) that doesn’t require changes in the codebase or the intervention of the Dev team.

That experiments will be designed and implemented by non technical people using a tool embedded in our Lounge application, this way we will reduce the friction with the Dev team, and make easier and less prone to errors the process of implementing them.

The other focus is about how we design the experiments, as we said on previous chapters, we design the experiments in a manual way, checking the reports and analyzing the data of our site, our key metrics and the relations between them.

The next step forward is to use predictive analysis tools in order to predict how a specific experiment can improve our key metrics. This is very important as this way we can rethink experiments if the prediction is really bad before we launch it to production, also as we are a small team, and we usually don't have enough time for implement all the defined experiments, many of them have to wait on the pipeline. So, this predictions also can be very useful as a guide to prioritize the experiments with a better "improvement prediction" and to leave on the pipeline or even rethink the experiments with the worse prediction.

6. Bibliography

- [AK09] *Web Analytics 2.0: The Art of Online Accountability and Science of Customer Centricity*, Avinash Kaushky, 2009, Sybex Publishing

- [AK07] *Web analytics, an hour a day*, Avinash Kaushky, 2007 Sybex Publishing

- [VO13] VersionOne website, VV. AA, available at http://www.versionone.com/what-is-kanban/, 2013

- [KB13] KanBan blog website, VV. AA, available at http://www.kanbanblog.com/explained/, 2013

- [OP13] Optify website, VV. AA, 2013, available at: http://www.optify.net/, 2013

- [MC12] *What Is Web Analytics And How To Get Started: An Introduction To The Web Analytics Process*, John M Cassidy, CreateSpace Independent Publishing Platform (March 20, 2012)

- [CO12] *Collins Cobuild Dictionaries of English*, VV. AA, Heinle ELT; 7 edition, August 3, 2012

- [WIK13] Wikipedia, VV. AA, 2013

- [37S11] 37 Signals blog, VV. AA, available at http://37signals.com/, August 2011

- [MT13] - MindTools blog, VV. AA, available at http://www.mindtools.com/pages/article/newTED_01.htm, 2013

- [LM13] - LunaMetrics blog, VV. AA, available at http://www.lunametrics.com/blog/2013/05/22/reverse-goal-path-underappreciated-google-analytics-report/, May 2013
- [GN12] - *Running A/A tests is a waste of time and money*, Gregory Nog, available at http://brooksbell.com/blog/running-aa-tests-is-a-waste-of-time-and-money, Dec 2012

- [HB12] - *A/B Testing in Action: 3 Real-Life Marketing Experiments*, Margarita Georgieva, available at http://blog.hubspot.com/blog/tabid/6307/bid/31634/A-B-Testing-in-Action-3-Real-Life-Marketing-Experiments.aspx, May 2012

- [LS11] - *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*, Eric Ries, Crown Business; First Edition edition (September 13, 2011)

- [IBM13] *What is Hadoop?,* VV. AA, available at http://www-01.ibm.com/software/data/infosphere/hadoop/

- [AP12] Apache PIG website, VV. AA, available at http://pig.apache.org/, May 2012

- [RD13] Redis website, VV.AA, available at http://redis.io/

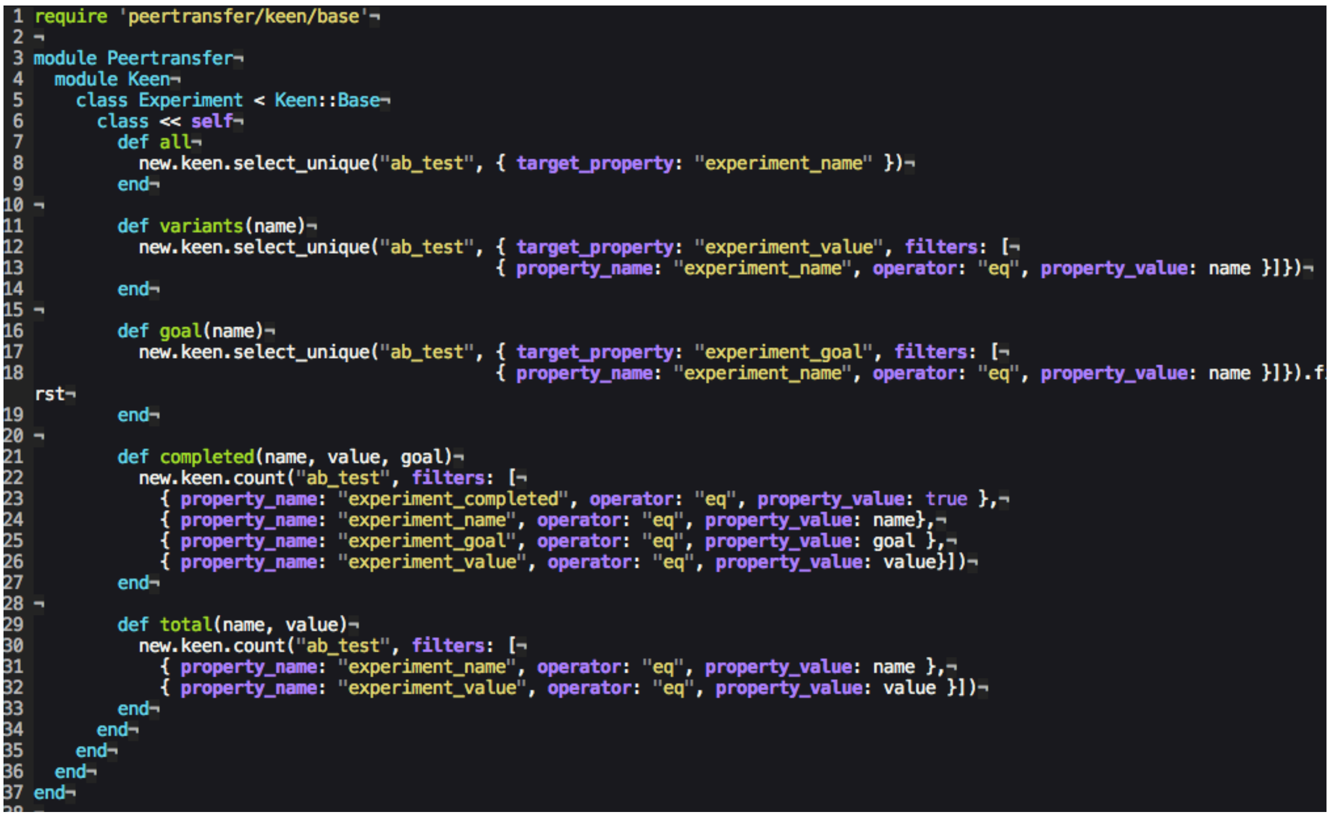
- [GI06] *G-test for independence*, John Mc Donalds, available at http://www.dmi.units.it, September 2006

- [PC08] *Pearson's Chi-square Test for Independence*, Catherin Light, available at http://www.ling.upenn.edu/~clight/chisquared.htm October 2008

Appendix A, main classes in the Smarty project.

Experiment class

The figure 6.1 is showing the Experiment class, which is the base class for the experiments, also is used for querying experiments, variants and goals in KeenIO.



```ruby
require 'peertransfer/keen/base'

module Peertransfer
  module Keen
    class Experiment < Keen::Base
      class << self
        def all
          new.keen.select_unique("ab_test", { target_property: "experiment_name" })
        end

        def variants(name)
          new.keen.select_unique("ab_test", { target_property: "experiment_value", filters: [
                                              { property_name: "experiment_name", operator: "eq", property_value: name }]})
        end

        def goal(name)
          new.keen.select_unique("ab_test", { target_property: "experiment_goal", filters: [
                                              { property_name: "experiment_name", operator: "eq", property_value: name }]}).f
rst
        end

        def completed(name, value, goal)
          new.keen.count("ab_test", filters: [
            { property_name: "experiment_completed", operator: "eq", property_value: true },
            { property_name: "experiment_name", operator: "eq", property_value: name},
            { property_name: "experiment_goal", operator: "eq", property_value: goal },
            { property_name: "experiment_value", operator: "eq", property_value: value}])
        end

        def total(name, value)
          new.keen.count("ab_test", filters: [
            { property_name: "experiment_name", operator: "eq", property_value: name },
            { property_name: "experiment_value", operator: "eq", property_value: value }])
        end
      end
    end
  end
end
```

Figure 6.1, Experiment class

Event class

The Figure 6.2 figure shows the Event class inside the KeenIO module, this class is the holder for the Event, and is the data structure (serialized to JSON) that we send to KeenIO.

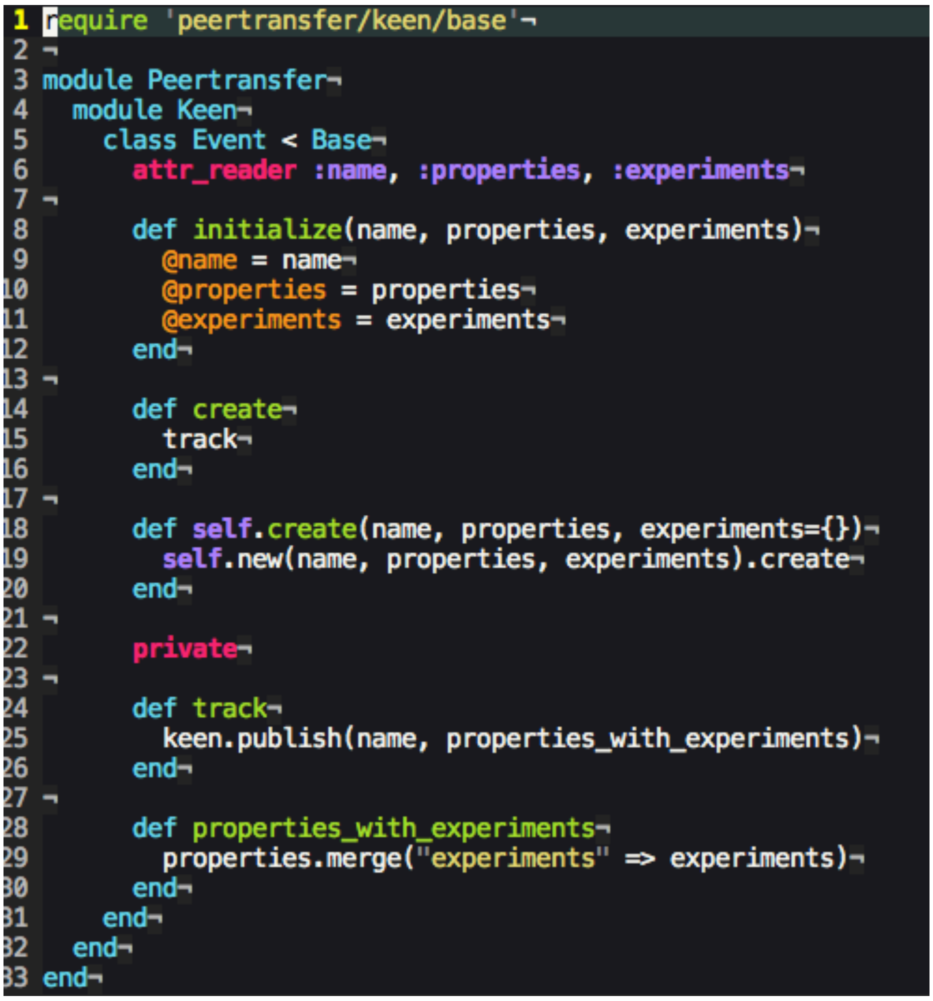
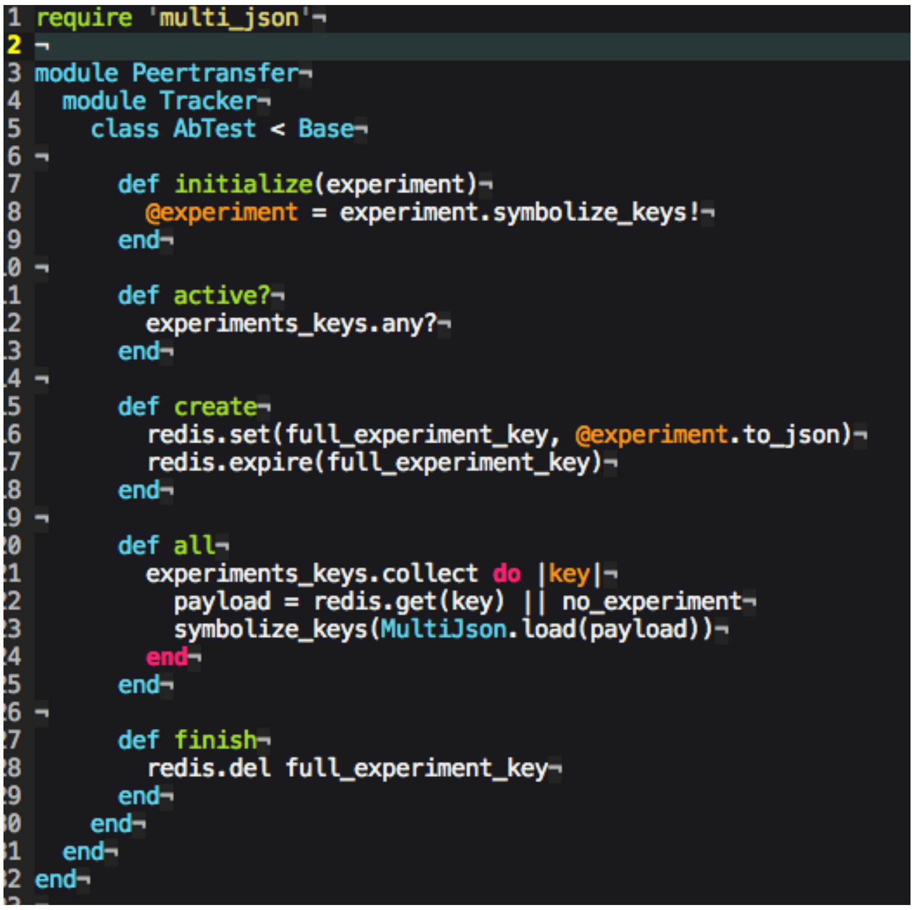

```ruby
require 'peertransfer/keen/base'

module Peertransfer
  module Keen
    class Event < Base
      attr_reader :name, :properties, :experiments

      def initialize(name, properties, experiments)
        @name = name
        @properties = properties
        @experiments = experiments
      end

      def create
        track
      end

      def self.create(name, properties, experiments={})
        self.new(name, properties, experiments).create
      end

      private

      def track
        keen.publish(name, properties_with_experiments)
      end

      def properties_with_experiments
        properties.merge("experiments" => experiments)
      end
    end
  end
end
```

Figure 6.2, Event class

AbTest class

The Figure 6.3 corresponds with the AbTest class, this class is used for track and control the experiments inside our platform.



```ruby
require 'multi_json'

module Peertransfer
  module Tracker
    class AbTest < Base

      def initialize(experiment)
        @experiment = experiment.symbolize_keys!
      end

      def active?
        experiments_keys.any?
      end

      def create
        redis.set(full_experiment_key, @experiment.to_json)
        redis.expire(full_experiment_key)
      end

      def all
        experiments_keys.collect do |key|
          payload = redis.get(key) || no_experiment
          symbolize_keys(MultiJson.load(payload))
        end
      end

      def finish
        redis.del full_experiment_key
      end
    end
  end
end
```

Figure 6.3, Abtest class