MIT ESD

Massachusetts Institute of Technology
Engineering Systems Division

ESD Working Paper Series

Feature Performance Metrics for Software as a Service Offering: The Case of HubSpot

Avi Latner
MIT, System Design and Management
Cambridge, MA, USA

Ricardo Valerdi
MIT, Lean Advancement Initiative
Cambridge, MA, US

ESD-WP-2011-04 March 2011 esd.mit.edu/wps

Feature Performance Metrics for Software as a Service Offering: The Case of HubSpot

Avi Latner
MIT, System Design and Management
Cambridge, MA, USA

Ricardo Valerdi
MIT, Lean Advancement Initiative
Cambridge, MA, USA

Abstract— **this paper provides an industry case study for measuring the performance of software as a service (SaaS) product features in order to prioritize development efforts. The case is based on empirical data from HubSpot and it is generalized to provide a framework applicable to other companies with large scale software offerings and distributed development. Firstly, relative value is measured by the impact that each feature has on customer acquisition and retention. Secondly, feature value is compared to feature cost and specifically development investment to determine feature profitability. Thirdly, feature sensitivity is measured. Feature sensitivity is defined as the effect a fixed amount of development investment has on value in a given time. Fourthly, features are segmented according to their location relative to the value to cost trend line into: most valuable features, outperforming, underperforming and fledglings. Finally, results are analyzed to determine future action. Maintenance and bug fixes are prioritized according to feature value. Product enhancements are prioritized according to sensitivity with special attention to fledglings. Underperforming features are either put on "life-support", terminated or overhauled.**

Keywords- Product development; performance metrics; software as a service; customer value

I. INTRODUCTION

In the past decade, with the increase of internet bandwidth, decrease in hardware costs, and a paradigm shift in business models, software delivery has been increasingly done by software as a service (SaaS) model [1]. The software application is hosted at the vendor, licensed for a reoccurring subscription fee and accessed through a web browser. SaaS is not only a change in the business model but a change in the product development paradigm [2].

Since software typically resides on vendor's servers, it is easier for them to release updates at more frequent intervals. Coupled with agile development practices, applications are updated almost continuously without traditional version control. This also allows the vendor to collect valuable information about customer usage patterns. The information available is unprecedented in scope and immediate in availability. With a continuous deployment model and immediate customer response the feedback loop between development and customers has never been faster.

However, in order to fully leverage the fast feedback loop, companies must use the right performance metrics. Most software suites are a collection of several features or applications. It is important to focus development efforts on the features where the investment will make the most impact on software usage and company profitability [3, 4]. For startups, often with limited cash flow and high uncertainty, this focus is all the more important.

In a SaaS model, much like other subscription based models, vendors use relationship marketing where customers are viewed as having a long term relationship instead of a series of discrete transactions [5]. Hence, the marketers' objective is to acquire and keep customers [6].

Sharing a common evaluation metric, focusing development efforts and communicating a shared vision is important in any software project. The emphasis is even greater in distributed development where multiple locations exacerbate communication problems [7]. Former research conducted [8, 9] proposed integrating requirement engineering techniques into a single framework fitted for distributed development. This research proposes an enhancement to this framework that enables requirement prioritization.

II. METHODOLOGY

This research was done using data collected at HubSpot.com, an online inbound marketing service. The service includes several features that help websites get found online by more qualified visitors, show customers how to convert more visitors into leads, give customers tools to close those leads efficiently, and provide analytics to help make smart marketing investments. HubSpot offers several products that differ in level of support. All products offer the same 17 features. HubSpot is an interesting case study as it has a diversified service with many features, a rapidly expanding customer base, open management and it implements the latest development methodologies such as lean startup [10]. HubSpot development and other business functions are located at one site. This made HubSpot an ideal case to research since people could be interviewed in person. However, the framework is extendable to other distributed development activities.

Usage information was collected from HubSpot's 3,000 customers over a period of four months. For each feature the percentage of users that used the feature was calculated. If a user accessed a feature at least once in the week prior to the measure than it counted as an active user. In order to eliminate seasonal volatility the usage was based on the average of four measures taken at the end of four consecutive months.

Development cost was collected from evaluating sprint logs (short development activities) from the earliest point available. In total 27 sprints were taken into account. Each sprint stands for a calendar month. The logs had indication of the storylines that were done in a given sprint and their story points which are the agile development measure for effort [11][11]. The research involved the analysis of over a thousand storylines and assigned each storyline to a feature. Some storylines were assigned to more than one feature and others representing back-end work or cross functional work were not assigned. Story points' scale changed over time and in between scrum teams. Also team sizes changed over time. In order to calculate the total amount of effort per feature for the entire period the sprint efforts were adjusted according to relative team size and the number of story points per team per sprint.

For feature sensitivity the study looked at two measurements six months apart, from sprint 21 and 27. Value was calculated based on usage and early usage data without expert opinion. Usage was also calculated based on one month data and without an average of several months mainly because some features go back only as far as sprint 21. This also makes the value measurement more sensitive to changes over time.

III. Feature Performance Framework

This paper proposes a framework by which to prioritize product development efforts. By following the proposed steps a company can gain awareness of relative importance of product features and examine how well their past decision making is aligned with feature importance. A step beyond that would be for a company to make future prioritization decisions in light of these findings.

Two hypotheses are examined as a base for the framework. The first is connection between customer feature usages is a predictor of customer retention. This premise is essential for capturing feature value. The second hypothesis explores the correlation between feature investment and feature value.

Feature performance changes over time for external reasons such as emergent customer preferences and internal strategic priorities. If this methodology is followed a company's focus may shift causing a change in performance assessment. Features that where once under invested may receive more investment and, as a result, may increase in value. Another feature may mature and exert its full potential suggesting a shift in investment. Therefore it is suggested that a company repeat this framework and re-evaluate the situation every time there is a significant change in the business environment (new competitor entering the market, new uses by customers, new capability available). Feature value, which mainly captures external shifts in the way the product is used, should be calculated as frequently as once a month. If the process is fully automated this may even be done once a week. Measures that are also dependent on internal shifts in investment such as feature profitability and sensitivity should be calculated less frequently, perhaps once a quarter. Since investment is based on cumulative data a few months have to pass before significant change may be observed.

A. *Usage As a Predictor of Customer Retention*

If customers that frequently use a given product are less likely to discontinue service we can use usage data of a given feature as a proxy for the impact that the feature has on customer retention. This connection is tested by the following hypothesis:

- H_0: No difference exists between retention of customers who use a feature and those who do not
- H_1: Customers who use a feature are more likely to retain service subscription

B. *Correlation Between Development Effort and Value*

It is assumed a company, even without having implemented this framework, would have some understanding of feature value and would therefore strive to align development effort to feature value. If this assertion is true we should see a correlation between development effort and feature value. This assumption is formulized as the following hypothesis:

- H_0: No correlation between development effort and feature value
- H_1: A positive correlation exists between development effort and feature value

C. *Measuring Feature Value*

SaaS product revenue is a function of the customer base and the product price. Customer base is a function of the number of new subscribers and the attrition rate. Therefore a measure of a feature value should actually be a measure of the impact that a feature has on these key parameters: customer acquisition and attrition rate.

Studies suggest that retaining existing customers is more important than acquiring new ones [12, 13]. One reason for that is that service discontinuers create a negative word of mouth that is more powerful than the positive word of mouth of continuing customers. Another reason is that new customers come at high adoption cost of sales and marketing [14]. Hence our research gives more value to retention rate impact than to acquisition impact using a weighted average of 70% to 30% respectfully. The exact weight should be a managerial decision and is a way to focus a company's priorities. A company more concerned with high attrition should choose a weighting such as this one with high preference towards retention. Whereas a company concerned with slow adoption should choose a more evenly weighted measure closer to 50% to each parameter.

This choice of parameter is validated through the hypothesis articulated in section III A. Our other measure of feature retention value is expert opinion survey done amongst business development and support staff within the company. Surveyed employees were asked to rate the top five most valuable features in the product to the customers. With no preference to either measure they were given equal importance.

Feature effect on customer acquisition is computed using two equally weighted measures. The first is the customer usage data in the first 30 days after subscription to the service started, since the usage in this early period reflects the features that made the customer subscribe to the service. The second is the

expert opinion of the sales representatives. The salespeople were asked to rank the five most important features in closing a sale. The equations below summarize the measure calculations.

$$value_i = 0.7 \cdot retention\ score_i + 0.3 \cdot adoption\ score_i \quad (1)$$

$$retention\ score_i = 0.5 \cdot usage_i + 0.5 \cdot support\ poll_i \quad (2)$$

$$adoption\ score_i = 0.5 \cdot early\ usage_i + 0.5 \cdot sale\ poll_i \quad (3)$$

Where i denotes a feature out of n features in a given SaaS offering.

Analyzing usage data in this way is valid in cases where all the features are client-facing, meaning that customers utilize the features by actively accessing them. When a SaaS product contains back office features, such as a product for network maintenance that has an automatic remotely triggered fix feature, a different measure must be used. Another example is user-automated reports that run without user interference. For example, Salesforce.com found a strong connection between user adoption of automated reports and retention. In their case a feature's value formula should also measure the amount of user automation. If value is calculated in weekly or monthly basis it is advised to repeat the value measures without surveying expert opinion. Expert opinion could be used to determine value of individual features but it should not be the primary basis for this analysis.

D. Measuring Feature Profitability

Capturing cost is in principle straight forward. Many of the costs are feature specific whereas the value of individual feature is more difficult to quantify since the customer pays for the service as a whole. The lion part of the cost of a feature is development costs. Most of the development effort is spent on building, enhancing or debugging a given feature. Other reoccurring monthly costs are also feature-specific. These costs of goods sold could be servers and storage. Cost is the accumulation of investment on a feature over the product history.

One way to have value and cost in comparable units is to allocate all the revenue according to value and to do the same for costs. With costs that are not feature specific a pro-ratio allocation of costs could be used. If that is the case than difference between total feature value and total cost value should be the gross margin. However doing that would be a time consuming effort. Since this paper aims at giving a practical measure it is important that the measure be as simple as possible without compromising accuracy. Hence, it is advised to keep value and cost at relative terms by dividing 100% of value and cost amongst the features.

$$profitablity_i = value_i - breakeven\ value_i \quad (4)$$

$$breakeven\ value_i = cost_i \cdot (1 - Gross\ Margin) \quad (5)$$

Where i denotes a feature out of n features in a given SaaS offering.

IV. Measuring Feature Sensitivity

Feature sensitivity is defined as the effect a fixed amount of development investment has on value in a given time. It is a measure of how effective recent development investments have been in improving features. Sensitivity is a dynamic measure that captures change between two time periods. One should use two measures of value and cost that are significantly apart, perhaps four or six months. Since feature value and cost described in sections II A and B are measured in relative terms the average sensitivity would be zero and many of the features will have negative sensitivity. It is our experience that overall zero sensitivity can be counter-intuitive to some business managers. To prevent that, the value in a given time can be multiplied by the growth in customer base. This way than the average sensitivity score would be equal to the customer base growth rate. The overall sensitivity score will than reflect the product performance as a whole. The equation set below summarizes the sensitivity measure.

$$sensitivity_i = \frac{(N)_t}{(N)_{t-1}} \cdot \frac{(value_i)_t - (value_i)_{t-1}}{(cost_i)_t - (cost_i)_{t-1}} \quad (6)$$

V. Results

The hypothesis testing connection between customer usages to retention was applied to five features separately and on the product as a whole. The total number of customers sampled in each test was 2,843. In all cases, but one, the null hypothesis was rejected at a 90% confidence level. For the product as a total the null hypothesis was rejected at a 99% confidence level. Meaning, users who are not using a feature are much more likely to discontinue service than active users. The only feature were the null hypothesis could not be rejected at a significant level is list manager. That is due to a very low population of customers that use list manager. Although features where tested separately they are not necessarily independent variables. In fact that data implies a correlation between feature usages.

TABLE I. Usage Connection to Retention

Feature	Attrition Rate Active Users	Attrition Rate Non-Active Users	% Usage	P Value
Blog	0.47x	2.89x	30.53%	~0
Content Management	1.3x	2.51x	39.71%	0.0376
Leads	2.03x	3.53x	44.50%	~0
Landing Page	0.76x	2.9x	23.41%	~0
Lead Nurturing	0.77x	2.22x	10.15%	0.054
List Manager	0.97x	2.58x	2.29%	0.356
Product	x	2.69x	82.35%	0.0003

Table I provides the results of the statistical tests of the hypotheses. To protect sensitive business information retention is stated in relative term to 'x' the attrition rate for active users of the product as a whole.

Value was computed using the formulas presented in section III C. Around 60 support and sale personal were polled to compute the scores. The components of the value score and the derived value are presented in table II. The table also shows value and development effort for each feature. The value and

development effort scores in table II are used to test the second hypothesis of correlation between development effort and value. The relation between value and cost is *value=0.64*development effort + 0.021;* R^2*=0.44*. The null hypothesis is rejected at a 90% confidence level showing that there is a positive relationship between value and development effort.

TABLE II. VALUE AND COST SCORE FOR 17 FEATURES

Feature	Usage Score	Support Poll Score	Early Usage Score	Sale Poll Score	Value Score	Cost Score
Leads	10.82	21.09	6.81	15.36	14.49	17.08
Sources	9.06	17.68	12.05	21.82	14.44	8.72
Content Management	9.96	9.41	17.14	2.54	9.73	8.63
Landing Page	5.91	14.78	5.58	9.25	9.47	2.22
Keyword Grader	6.80	9.19	9.67	14.87	9.28	7.96
Blog	8.17	5.37	9.89	8.91	7.56	4.21
Social Media	6.15	2.21	11.09	3.63	5.14	8.07
Competitors	6.72	2.50	1.14	9.41	4.81	10.03
Page Grader	4.52	1.98	7.23	3.66	3.91	6.89
Lead Nurturing	2.93	4.92	3.18	2.61	3.62	9.75
Blog Grader	6.45	0.92	4.21	0.59	3.30	3.66
Prospects	4.56	2.94	2.22	0.93	3.10	1.86
Link Grader	5.84	0.96	2.26	1.11	2.88	1.96
Visits by Page	5.71	0.39	3.29	1.01	2.78	1.36
Reach	4.36	0.40	3.37	0.40	2.23	1.60
Email	0.96	3.44	0.42	2.48	1.98	3.74
List Manager	1.06	1.80	0.45	1.41	1.28	2.26
Total	100	100	100	100	100	100

Finally we compute feature profitability and feature sensitivity in table III. As you can see the total profitability is 60% and is equal to the company's gross margin at the time. The sum of all sensitivity scores is 0.24 which is the growth of customer base between the two sensitivity measurements.

TABLE III. FEATURE PROFITABILITY AND SENSITIVITY

Feature	Profitably	Sensitivity
Sources	11%	0.29
Landing Page	9%	0.80
Leads	8%	0.25
Content Management	6%	0.21
Keyword Grader	6%	0.03
Blog	6%	1.11
Prospects	2%	(0.06)
Visits by Page	2%	0.28
Link Grader	2%	0.48
Social Media	2%	(0.05)
Blog Grader	2%	0.21
Reach	2%	0.18
Page Grader	1%	(0.06)
Competitors	1%	0.09
Email	0%	0.14
List Manager	0%	0.24
Lead Nurturing	0%	(0.13)
Total	60%	0.24

VI. CONCLUSIONS

Based on the results we segment the features. This is most easily done by looking at a scatter plot, as seen in figure I, of cost on the horizontal axis and value on the vertical axis. The scatter plot should also have a line representing the gross margin and the linear regression line.

The most valuable features are a segment of features that are high in value and investment. These features are recognized as important by the company. In our case these are leads and sources. This group should be the highest in priority for bug fixes and regular up-keep. As long as the sensitivity is positive they should also be considered for enhancements.

Outperformers are a segment of features that are doing very well relative to the investment in them. In a scatter plot described above they will they appear closest to the top left corner. In this case they are blog, content management and landing page. By contrast underperforming is a segment containing features that in retrospect do not justify their investment. In this research they are lead nurturing, social media, competitors and page grader. Out of this group the features with zero or negative profitability need re-examination. If value covers the cost of goods sold and there is little maintenance development anticipated the feature could be kept on 'life support'; that is kept alive while avoiding investment as much as possible. Otherwise the feature should either be terminated or overhauled.

In the fourth segment, fledglings are features that have had little investment and provide little value. They include link grader, landing page, blog grader, visits per page, prospects and reach. It is interesting to note the existence of a long tail effect; most of the value derives from a few of the features and the fledgling group is the largest one. However this group also holds the most potential as it may include promising features that are yet to mature. For example, link grader and visits by

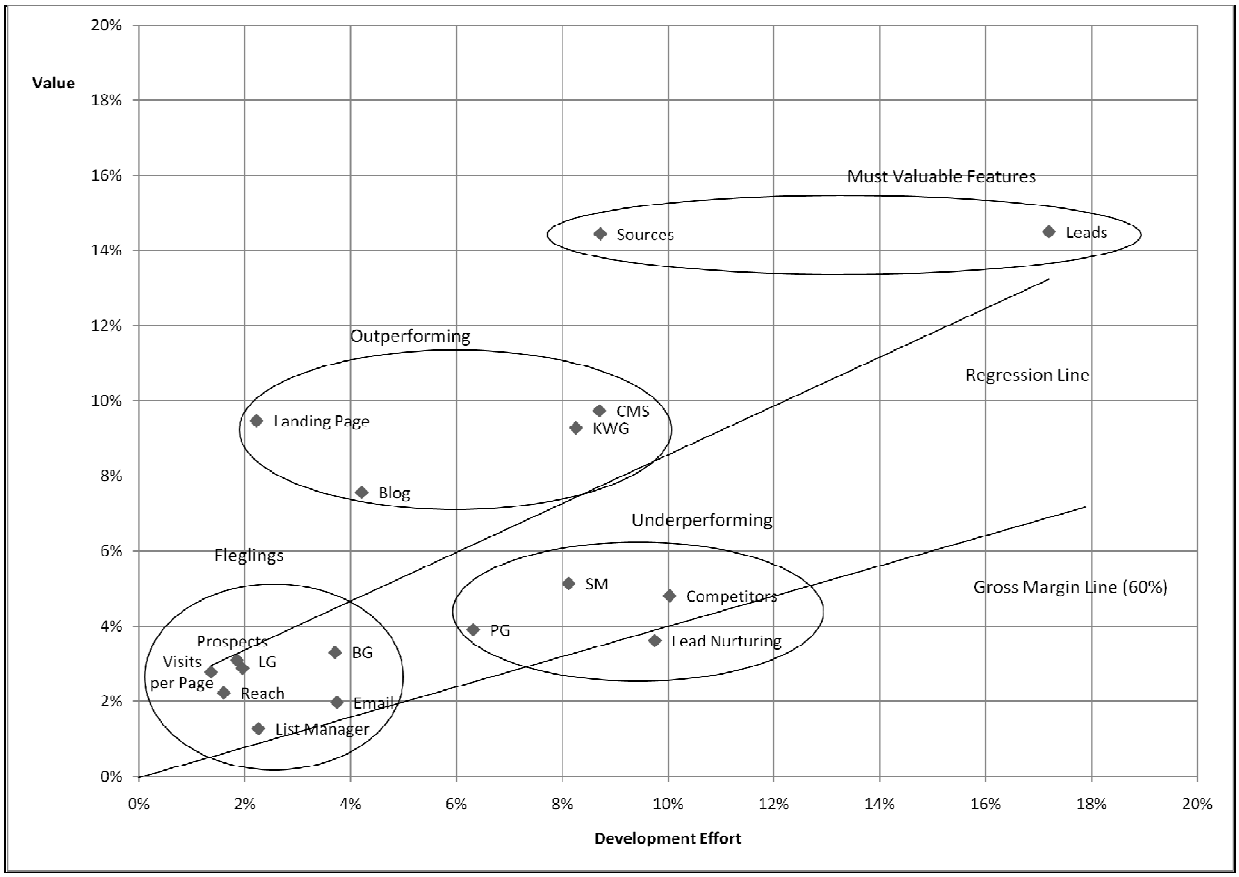

Figure 1. Scatter Plot of Value and Development

page have an above average sensitivity and therefore they are ideal candidates for future investment.

The framework provided in this paper is applicaple to other industry cases. It could benefit a software product if these conditions apply: (1) the prodcut is provided as SaaS, (2) the product has a defined set of features and (3) have a large enough number of features to justify a quantitative measurement system.

ACKNOWLEDGMENT

The authors gratefully acknowledge contributions made by collaborators at HubSpot: Yoav Shapira, Bradford Coffee, Dan Dun and Joss Poulton.

REFERENCES

[1] A. Dubey and D.Wagle, "Delivering Software as a Service", The McKinsey Quarterly, May, 2007.

[2] C. Vidyanand, "Software as a service: implications for investment in software development", Proceedings of the 40th Hawaii International Conference on System Sciences (HICSS), Waikoloa, Hawaii, January, 2007.

[3] D. Hubbard, How to measure anything: finding the value of intangibles in business, Hoboken New-Jersey: John Wiley and Sons, Inc., 1962, pp. 119-136.

[4] J. McGarry, D. Card, B. Layman, E. Clarck, J. Dean et al., Practical software management: objective information for decision makers, Indianapolis Indiana, Addison-Wesley, 2001, pp. 13-26.

[5] D. Berger and N. Nasr, "Customer lifetime value: marketing models and applications," Journal of Interactive Marketing, vol. 12, pp. 17-30, 1998.

[6] P. Kotler and G. Armstrong, Principles of marketing, 7th ed., Englewood Cliffs New-Jersey: Prenlice-Hall, 1996.

[7] A. Taweel, B. Delaney, T. Arvanitis and L. Zhao, "Communication, knowledge and co-ordination management in globally distributed software development: informed by a scientific software engineering case study", IEEE ICSGE, pp. 370-375, July 2009.

[8] B. Berenbach and M. Gall, "Toward a unified model for requirements engineering", IEEE ICSGE, pp. 237-238, October 2006.

[9] B. Berenbach and T. Wolf, "A unified requirements model; Integrating features, use cases, requirements, requirements analysis and hazard analysis", IEEE ICSGE, pp. 197-203, August 2007.

[10] E. Ries, Lessons learned: what is lean about the lean startup? available at: < http://www.startuplessonslearned.com/2009/12/what-is-lean-about-lean-startup.html>, accessed on: Feb 16, 2011.

[11] M. Cohn, User stories applied for Agile software development, Boston MA. Addison Wesley, 2004, pp. 87-96.

[12] L. Oliver, Satisfaction: a behavioral perspective on the consumer, New York. McGraw-Hill, 1997.

[13] M. Mahajan, E. Muller and R. Kerin, "Introduction strategies for new products with positive and negative word-of-mouth", Management Science, vol.30, December 1984.

[14] P. Madhavan and A. Bhattacherjee, "Understanding post-adoption behavior in the context of online services", Information System Research, vol. 9, no. 4, December 1998.