Predicting Field Reliability

Pete Rotella
Cisco Systems, Inc.
7200 Kit Creek Road
Res. Triangle Pk., NC, USA 27709
+1-919-392-3854
protella@cisco.com

Sunita Chulani
Cisco Systems, Inc.
125 W. Tasman Drive
San Jose, CA, USA 95134
+1-408-424-6802
schulani@cisco.com

Devesh Goyal
Cisco Systems, Inc.
125 W. Tasman Drive
San Jose, CA, USA 95134
+1-408-853-7214
devgoyal@cisco.com

ABSTRACT

The objective of the work described is to accurately predict, as early as possible in the software lifecycle, how reliably a new software release will behave in the field. The initiative is based on a set of innovative mathematical models that have consistently shown a high correlation between key in-process metrics and our primary customer experience metric, SWDPMH (Software Defects per Million Hours [usage] per Month). We have focused on the three primary dimensions of testing – incoming, fixed, and backlog bugs. All of the key predictive metrics described here are empirically-derived, and in specific quantitative terms have not previously been documented in the software engineering/quality literature. A key part of this work is the empirical determination of the precision of the measurements of the primary predictive variables, and the determination of the prediction (outcome) error. These error values enable teams to accurately gauge bug finding and fixing progress, week by week, during the primary test period.

Categories and Subject Descriptors

I.6.5 [**Simulation and Modeling**]: Model Development – *modeling methodologies.*

General Terms

Algorithms, Measurement, Reliability, Experimentation, Theory.

Keywords

Software release reliability, prediction, modeling, testing, error analysis, customer experience.

1. INTRODUCTION

For several years, Software Defects Per Million Hours (SWDPMH) has been the primary customer experience metric used at Cisco. This metric is goaled for product teams on a yearly basis, and this includes 120 product families in 2015. SWDPMH is considered to be a reasonable measure of "customer pain," since we count each time a bug is found by the customers. The metric A key reason SWDPMH is considered to be of critical importance is that we see a high correlation between SWDPMH and Software Customer Satisfaction (SW CSAT), as measured by our yearly customer survey, over a wide range of products and feature releases. Therefore, it is important to anticipate SWDPMH before the software is released to customers, for several reasons:

- Early warning that a feature release is likely to experience substantial field quality problems may allow for remediation during, or prior to, function and system testing on the "integration branch." (The integration branch is the software branch that results from the collapse of the development branches, in the case of a waterfall project, or the subsequent function and system testing on an agile development branch.)
- Prediction of SWDPMH enables better planning for rollout strategies and for maintenance releases. If we anticipate that SWDPMH will be too high, distribution of the release can be restricted until rebuilds can improve the reliability.
- Calculating the tradeoffs between SWDPMH and feature volume can provide guidance concerning acceptable feature content, test effort, release cycle timing, and other key parameters affecting future feature releases.

Our recent efforts have focused on enhancing our ability to predict SWDPMH in the field. Toward this end, we, and many others [1][2], have developed several predictive models, have tested the models with major feature releases for strategic products, and have provided guidance to development, test, and release teams on how to improve the chances of achieving best-in-class levels of SWDPMH and SW CSAT.

2. MODELING RELEASE QUALITY

Predicting field SWDPMH future experience is of paramount importance to the development, test, and technical support organizations. If the predictions can be made early enough in the development/test phases, steps can be taken to improve the release – steps such as adding testers, pulling out non-essential features, more time testing, push out the release date, etc.

2.1 Model General Characteristics

Our recent work has concentrated on models targeting the function and system test phases as typically conducted in our environment for waterfall and hybrid waterfall/agile development programs. Results for waterfall and hybrid waterfall/agile are consistently good. Results for pure agile projects are encouraging, but more testing is needed to check the minor adaptations needed for a pure agile environment.

ESEC/FSE'15, August 30 – September 4, 2015, Bergamo, Italy
© 2015 ACM. 978-1-4503-3675-8/15/08...$15.00
http://dx.doi.org/10.1145/2786805.2804428

The initial experiments focusing on the effect of test practices and processes on SWDPMH were done using three major software trains that are resident on 9 product families, and include 23 major feature releases. The dependent variables used were SWDPMH at 6, 9, and 12 months after release to the field. SWDPMH's numerator is the number of software 'incidents' encountered in the field, which is the number of times a field bug is seen by any customer. (For example, one bug seen by 10 customers would contribute a value of 10 to the numerator of SWDPMH. These bug 'incidents' are referred to here as 'BugSRs.')

A total of 340 independent variables were included in the initial modeling exercise; these include ~50 basic un-normalized variables, and about 100 normalized variables for each of the primary testing "dimensions", namely incoming, disposal, and backlog variables. Our aim was to find a highly predictive variable in each of these primary dimensions, if possible, since our organization normally measures many characteristics of these dimensions, and therefore additional data collection would not be needed. Also, incoming/disposal/backlog metrics are highly interrelated, and likely to be more fully characteristic of the testing and disposal functions. ("Disposal" means that the bug has reached a terminal state, including resolution, but also closed (i.e., a bug that is left unresolved), junked, duplicate, and unreproducible bugs.

Using JMP and Excel, we reduced the size of the independent variable array, and identified predictive (and useable) models. We found the most highly predictive variables are 'incoming' variables, followed by several disposal variables. All the backlog variables produce high (i.e., >0.10) P values and low levels of correlation (i.e., <40%) with the dependent variables. Table 1 is a summary of the best modeling results:

Table 1: Linear Regression, Best Modeling Results

Var.	Coeff.	S(x)	t	P	F	Adj. R2	S(y)	Inter.
x1	-47.0	4.5	-11	0.06				
x2	-12.1	1.4	-7.0	0.07				
y					0.09	0.92	0.26	42.4

Independent variable x1 we call Incoming Defect Level (IDL). It is the percentage of total bug content, as estimated using the asymptote of the Goel-Okumoto-Shaped (GOS) software cumulative incoming growth curve [3]. In other words, IDL is the actual percent of the total cumulative bug content of the release, estimated as the asymptote of the GOS S-shaped curve. Independent variable x2 we call Fix Rate Reduction (FRR). It is the percentage decline in the weekly bug disposal rate from the maximum level. Dependent variable y is the SWDPMH value for the platform/release in question, taken at 9 months after release to the field. Similar results are seen for SWDPMH observed at 12 months after release.

2.2 Model Results

To date, 423 analyses have been done for 43 platforms hosting 117 feature releases and 24 large maintenance releases. These platforms include router, switching, and datacenter hardware products, plus software-only applications and tools, such as security, collaboration, and network management.

Correlations between the full model equation scalar value and SWDPMH have been consistently high. For example, the adjusted R^2 for Product A, shown in the case study below, is at the high end (88%) of the range seen with the 43 products examined so far. The average correlation seen is 72% (range of 63%-88%).

3 ERROR ANALYSIS

A key requirement for any model used in a development/test environment is that the measurement and prediction errors be published and available to the users. Users need to know if the 'target' and 'actual' IDL and FRR curves are statistically distinct at each point in time during testing and fixing. If the target IDL curve is statistically higher than the actual curve, for example, the team will be asked to take action, such as adding resources, extending the testing timeline, reducing feature content, or taking another remediation step. Therefore, the error determinations for these curves constitutes an important practical step.

The IDL measurement error determination method is straightforward: The IDL measurement error is the standard error of the predictive variable, IDL, for the fleet of releases. In this analysis, we use the group of 31 releases, and the method used is:

- We calculate an empirically-determined measurement error (referred to here as 'residuals') for decile clusters of IDL from 10% to 80%. These residuals are calculated using the delta between the observed intermediate IDL values and the IDL value, for each release, derived using the end-point IDL asymptote value and its weekly cumulative values.
- Only releases that achieved an IDL level of 65% or higher were used in this analysis, since the standard error of the IDL variable in the model equation is added to the residuals-determined error, and the standard error of the IDL variable has only been determined above 65%.
- In other words, we use the final week's IDL to determine the most accurate asymptote, then derive 'actuals' for all previous weeks from that asymptote value. The residuals are rank ordered, and the forward-looking IDL calculations, week by week, are then compared to the 'actuals,' and the delta between the two is calculated. On each side of the 'actuals' curve, we find the point at which 32% of the residuals are found, and this point is the one standard deviation error bar point. In excess of 32% of the residual volume from the S-curve is beyond one standard deviation, therefore this 'distance' constitutes the one standard deviation error bar for the measurement precision.
- The error bars, upper and lower, are depicted in Figure 1 below for the 822 residual and 'actual' values for the 31 releases studied. Therefore, the blue lines correspond to the confidence intervals for the releases studied. The x axis shows one unit/tick for each of the 822 readings.
- In addition to the error determination derived from using the end-points of the each of the 31 Goel-Okumoto-Shaped cumulative growth curves, we need to add in the standard error (± 1.4%) of the IDL variable derived from the general model equation, since all the residual comparisons assume no

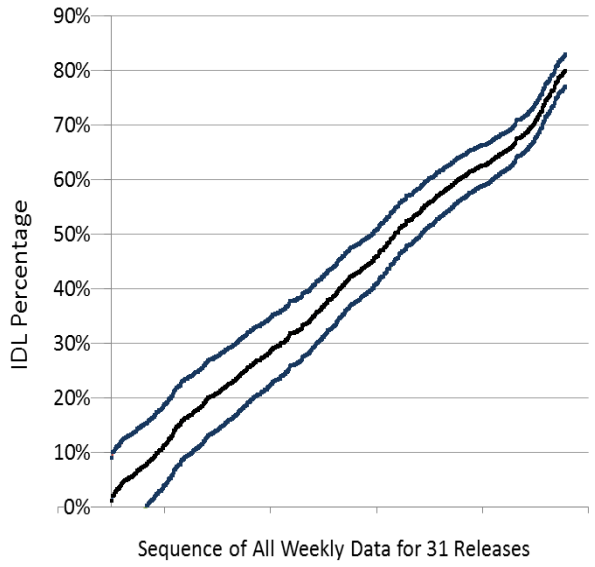

Figure 1: Residuals v. Actuals for 31 Releases

error in the final testing week measurement.

- (The average size of these releases is about 350 thousand lines of new plus modified source code.)

Specific findings of this exercise:

- Using the approach described, we find that the residual-based IDL measurement error is ± 3.0% within the primary region of interest, 60% < IDL < 85%, for the 31 releases studied – to this we need to add ± 1.4% for the model's standard IDL error, for a total of ± 4.4%.
- The residual-based IDL measurement error is ±5.5% within the IDL region from 40% < IDL < 60%, for the 31 releases studied, so the total error is ± 6.9% in this region.
- The IDL measurement error is ±7.4% in the IDL region from 10% < IDL < 40% for the 31 releases studied, so the total error is ±8.8% in this region.

Below are several examples of the product-specific variants of the error bar calculations described above. By 'product-specific,' we mean that an individual regression graph is constructed for the sequence of historical releases applicable to the specific product family, and product-specific error bars are constructed using the independent and dependent variable standard errors applicable to the releases used.

3.1 Case Study

Heat maps and regression graphs have been generated for the recent historical releases that are resident on the Product B platform. The heat map in Table 2 shows IDL, FRR, SWDPMH, and ancillary metrics for releases that have been available to customers for about the past three years. The 'IDL+FRR' column shows the scalar quantity derived from the combination of the two predictive independent variables, weighted according to the coefficients of the variables in the general model equation developed with the Product A and Product D data. The goal for this linear combination of IDL and FRR is >72%, the minimum value needed to enable the successor release sequence to achieve best-in-class SWDPMH levels within three years, whichever type feature or large maintenance release we examine.

Table 2: IDL&FRR/SWDPMH Heat Map, Product A

Product A	IDL+FRR (need >72%)	IDL (need >80%)	FRR (need >45%)	SWDPMH (FCS+150 BugSRs)
Rel. 7	72%	78%	43%	14.1
Rel. 6	74%	73%	80%	9.3
Rel. 5	63%	71%	36%	13.1
Rel. 4	65%	69%	53%	14.4
Rel. 3	61%	66%	35%	16.5
Rel. 2	73%	76%	58%	9.9
Rel. 1	86%	89%	71%	8.7

Figure 2 shows the relationship between the SWDPMH for the Product A historical releases and the percent IDL achieved at 'throttle pull' (i.e., the time most testing is complete):

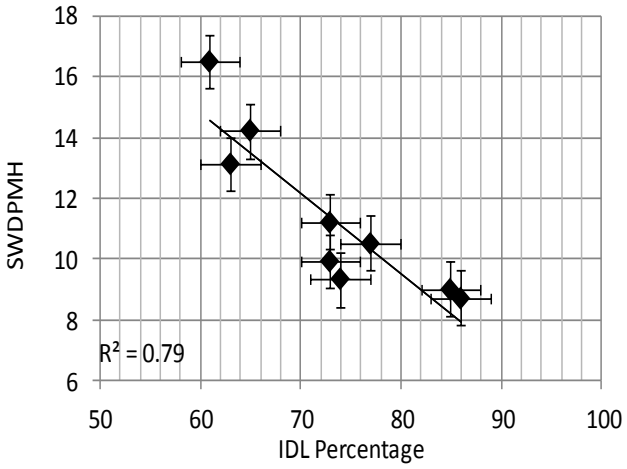

Figure 2: SWDPMH/IDL Regression Graph for Product A Historical Releases

The y-axis error bar for this specific population is ±6.2% (relative %), and the x-axis error bar is ±3.0% (absolute %).

3.2 Summary for All Releases Studied

All case studies have yielded similar IDL error analysis results, results similar to those shown in the case study of Product. Here is a summary:

- A total of 31 releases were studied, with a total of n=822 data points observed.
- The y-axis error bar is ±4.5% (relative%); mean y-axis value is 4.4% and x-axis value is 38%.

- Figure 3 shows the absolute percent error in IDL for various deciles of IDL value:

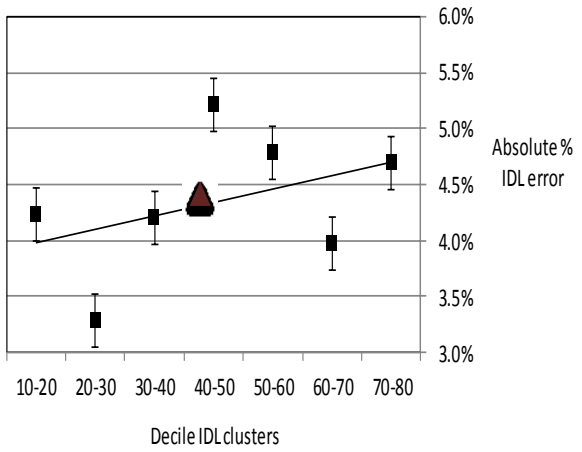

Figure 3: Absolute ±% Error in IDL Measurement for Decile IDL Clusters

The IDL measurement error calculation should include the full error – this error is a function of the average variance of individual incoming rates vis a vis the 'actual' growth curve values – specifically, the variance of the regression residuals that are centered around the actuals, plus the error of the actual readings.

4. CONCLUSIONS

Following are the conclusions from error analysis addressing 31 releases (also see Table 3). The IDL measurement error is:

- ±4.4% within the primary region of interest, 60%<IDL<85% for 31 releases studied; this includes ±3.0% residual-based measurement error and ± 1.4% standard error of the IDL variable.
- ±6.9% in IDL region of 40%<IDL<60%; includes ±5.5% residual error plus ± 1.4% standard error.
- ± 8.8% within the IDL region of 10%<IDL<40% for 31 releases studied; includes ± 7.4% residual-based error plus ± 1.4% standard error.
- IDL SWDPMH prediction error is 8.4%, from the region of 60%<IDL< 85% for the 31 releases.

Table 3: Summary, IDL & SWDPMH Prediction Errors

% IDL	Residual-based measurement error (± %)	Model standard error (± %)	Total measurement error (± %)
10-40	7.4	1.4	8.8
40-60	5.5	1.4	6.9
60-85	3.0	1.4	4.4

5. SUMMARY

The findings of this study are, so far:

- The combination of an incoming bug metric (Incoming Defect Level) and a bug disposal metric (Fix Rate Reduction) have been shown to be highly predictive of SWDPMH for 43 release sequences – the average Spearman correlation is 72% and the standard error of SWDPMH, the response variable, is only 7.6%.
- The model is applicable over a wide range of releases, and has the potential to be a broadly generalizable model. High correlations are seen for all systems studied so far, including router, switch, and datacenter releases, and releases for software-only applications
- Error analysis for one (i.e., IDL) of the two primary customer experience (i.e., SWDPMH) predictors has been completed. The measurement error for IDL varies between ±4% and ±9%, from 10% IDL to 85% IDL. This completes the key step in ascertaining whether or not teams are on track to achieving weekly bug incoming and fix rates, which, in turn, enables reaching best-in-class SWDPMH goals.

6. REFERENCES

[1] T. Menzies, A. Butcher, D. Cok, A. Marcus, L. Layman, F. Shull, B. Turhan, and T. Zimmermann, "Local vs. Global Lessons for Defect Prediction and Effort Estimation," in IEEE Transactions on Software Engineering: (2013), 2013.

[2] J. Musa, A. Iannino, and K. Okumoto, Software Reliability. McGraw-Hill, New York, NY, 1990.

[3] A. Wood, Software Reliability Growth Models: Report 96.1. Tandem Computers, Cupertino, CA, 1996.