# Software Escalation Prediction with Data Mining

Tilmann Bruckhaus
*Customer Network Services*
*Sun Microsystems, Inc.*
*USCA22-104, 4220 Network Circle*
*Santa Clara, CA 95054*
*Tilmann.Bruckhaus@Sun.Com*

Charles X. Ling, Nazim H. Madhavji, Shengli Sheng
*Department of Computer Science*
*University of Western Ontario*
*London, Ontario N6A 5B7, Canada*
*{cling, madhavji, ssheng}@csd.uwo.ca*

## Abstract

*One of the most severe manifestations of poor quality of software products occurs when a customer "escalates" a defect: an escalation is triggered when a defect significantly impacts a customer's operations. Escalated defects are then quickly resolved, at a high cost, outside of the general product release engineering cycle. While the software vendor and its customers often detect and report defects before they are escalated it is not always possible to quickly and accurately prioritize reported defects for resolution. As a result, even previously known defects, in addition to newly discovered defects, are often escalated by customers. Labor cost of escalations from known defects to a software vendor can amount to millions of dollars per year. The total costs to the vendor are even greater, including loss of reputation, satisfaction, loyalty, and repeat revenue. The objective of Escalation Prediction (EP) is to avoid escalations from known product defects by predicting and proactively resolving those known defects that have the highest escalation risk. This short paper outlines the business case for EP, an analysis of the business problem, the solution architecture, and some preliminary validation results on the effectiveness of EP.*

## 1. Introduction

Escalations of software products occur when1 a customer service case is "escalated to a local manager, Customer Service Manager (CSM), or designee", when "customer situations require management attention, continuous effort, and have an existing action plan which provides customer relief", or in case of "Critical situations which have substantial business or financial impact to [the vendor] or the customer." Product defect escalations are costly to the vendor as well as the customer, and associated labor costs can amount to millions of dollars each year. In addition, product defect escalations generally result in loss of reputation, satisfaction, loyalty and repeat revenue.

The objective of Escalation Prediction (EP) is to avoid such escalations from known product defects using data mining technology [1, 2]. If a software vendor can accurately predict the escalation risk of known defects, escalations can be prevented by fixing high-risk defects before customers escalate them. Accuracy, precision and recall are the common measures of assessing the performance of a prediction model. To some extent, they reflect the return on investment (ROI) of fixing predicted escalations. For a given predictive model, one can use historical data, known as a "hold-out data set" [3], to measure precision as the ratio of predicted escalations which in fact were escalated, and recall as the ratio of actual escalations that were predicted by the model. If we assume that an escalation has a specific cost X whereas fixing a defect has another, smaller, cost Y, we can express the "net profit", or the ROI in terms of precision and recall, and find their optimal values to maximize ROI.

## 2. Solution Architecture

The diagram (refer to Figure 1) illustrates the Escalation Prediction Solution Architecture. Data is captured periodically from the vendor's Online Transaction Processing (OLTP) systems providing defect and escalation information. Data is captured weekly and stored indefinitely in a datamart. While the information in the OLTP systems continues to change from moment to moment, the weekly snapshots in the

[1] Sun internal documentation

datamart remain constant so as to provide the historical data required for training predictive models.

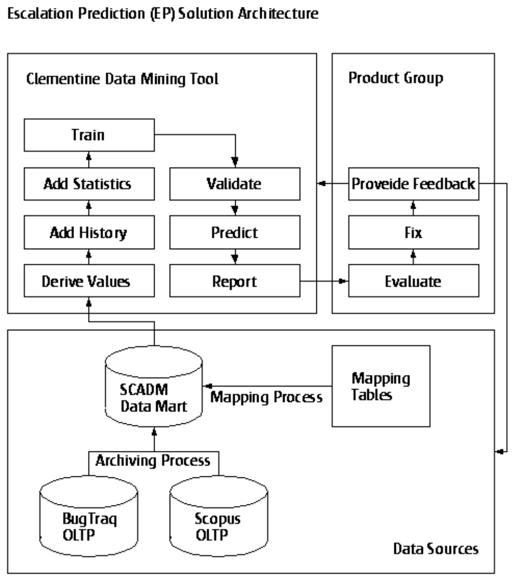

Figure 1.

The historical snapshot data is then augmented within the SPSS Clementine data-mining tool. Derived fields, historical information, and statistics are added to yield the set of available input fields: 1) data fields which come directly from the defect tracking system (e.g., priority, hardware platform, and customer company), 2) fields which are derived from individual source records (e.g., the source fields "user type" and "user role" are concatenated to derive "user type and role"), 3) fields capturing the history of bugs (e.g., a field containing the sequence states of the bug over the last 12 weeks), 4) escalation statistics for previous time periods (e.g., the percentage of bugs escalated by each customer company during the previous fiscal quarter), and 5) unsupervised learning techniques are used to cluster the data, and the resulting cluster membership data is available as additional inputs for modeling.

The next step is to train and validate predictive models. Training involves selecting a subset from among about 200 possible input variables, and setting up rule induction or neural network training parameters [4, 5] in Clementine. Generated models are then validated against historical data. The available data is split into two disjoint sets for training and validation so that the validation of the model is performed only against defects the model "has not seen" [6].

Once one or more satisfactory models have been found the most appropriate model is selected and run against the most recent snapshot of defect data. The predicted escalations are then reported to the product group for evaluation and proactive resolution.

The product group provides feedback to allow for ongoing improvement of the overall escalation prediction and prevention effort. Results of the overall program are also tracked in terms of actual precision and recall, as well as run rates of escalations from known defects.

## 3. Preliminary Results

The following charts show validation results. Our goal was to assess whether our models could predict escalation risk accurately so that bugs that were later escalated would receive high-risk scores. The EP model used here is based on neural networks. The data used for validation is historical data with known results from the live product development process from 9/15/2003 to 10/13/2003.

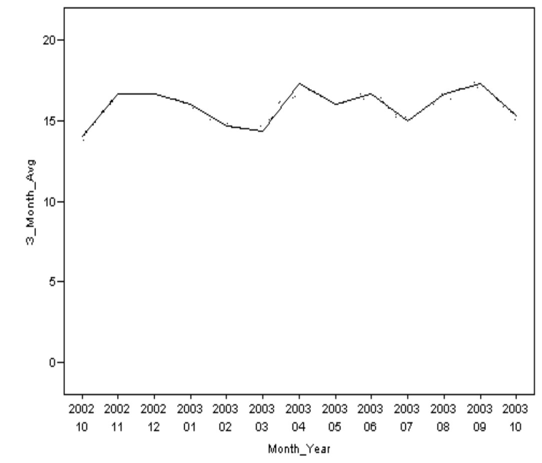

Figure 2.

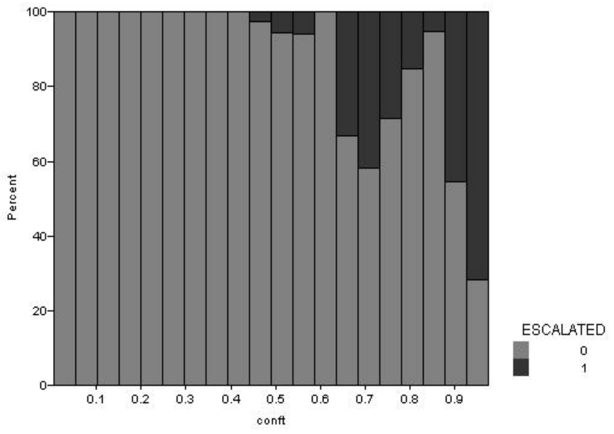

Figure 3.

Figure 2 shows the three-trailing-months average of Escalations from Known Defects against one large software product. The run rate ranges from 14 to 18 per month. This chart will be used to track the run rate of escalations from known defects over time. Our goal is to prevent such escalations, and to push the trend line as close to zero as we can. So far, this chart does not show evidence of run rate reduction because the product group has piloted EP for just one month.

Figure 3 shows validation results from an EP run. One of the outputs of the model is the "confidence" the model has in predicting that a given defect will be escalated. In essence, confidence translates into an "escalation risk level" assigned by the model. If the model works as desired, then defects with greater risk level will have a greater probability of becoming escalated. The histogram shows the risk level assigned to defects on the X axis, and on the y axis it shows the proportion of escalated vs. not escalated defects. Escalations are shown in dark gray and non-escalations in light gray. As the chart shows, the EP model successfully sorted defects, which later became escalated into the high end of the 'confidence' or EP Risk Level.

Based on the above information, we can calculate the probability of an escalation based on the risk level. To do so, we divide the number of escalations by the number of defects at each risk level. We consider probabilities of escalations for defects, which have "at least a given risk level", rather than for only defects "exactly a given risk level". This approach allows us to more easily see the trend of escalation probability with increasing risk level.

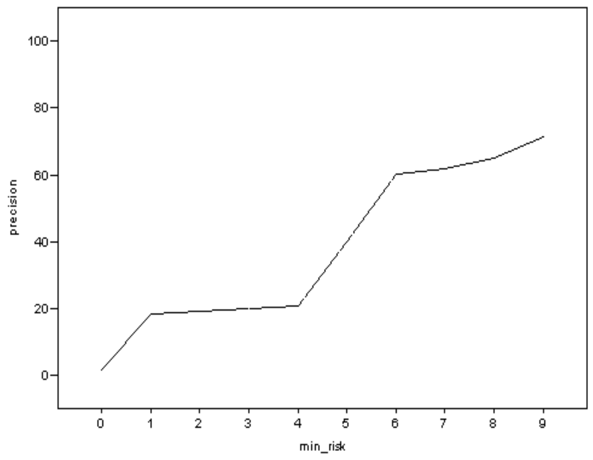

Figure 4.

Figure 4 shows that risk level 9 defects have a greater than 70% probability of becoming escalated. Escalation probabilities decline with declining risk level, to 60% for "risk level 6 and greater", 40% for risk level 5 and greater, 20% for risk level 4 and greater, and finally 1% for risk level 0 and greater. "Risk level 0 and greater" effectively means "overall escalation probability, independent of risk level". This figure is a quantitative assessment of how Escalation Prediction allows the vendor to focus on the "vital few" while expending less energy on the "trivial many".

## 4. Conclusions

In this short paper, we have made a business case for predicting and preventing escalations from known product defects. While the labor cost of escalations from known product defects is significant, the total cost is even greater, including loss of image, customer satisfaction, loyalty and repeat revenue. By applying predictive technologies that have been used successfully to similar problems in the financial services industry software vendors can proactively resolve known product defects with the greatest risk of escalation. An escalation prediction solution has been set up and tested, and is currently deployed at Sun. The solution has been deployed for only a small number of weeks. Preliminary results provide evidence that we can indeed predict escalations. We plan to continue to improve the effectiveness of the program and track its results.

## 5. References

[1] M.J.A. Berry and G. Linoff, *Data Mining Techniques: For Marketing, Sales, and Customer Support*, John Wiley & Sons, 1997.

[2] H. Dai (editor), *Proceedings of The International Workshop on Data Mining for Software Engineering and Knowledge Engineering,* 2003.

[3] T. Mitchell. *Machine Learning*, McGraw Hill, 1997.

[4] C.M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press. 1995.

[5] M. Smith, *Neural Networks for Statistical Modeling*, Van Nostrand Reinhold, 1993.

[6] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy (editors), *Advances in Knowledge Discovery and Data Mining*, AAAI/MIT Press, 1996.