



Practical use of defect detection and prediction

Christopher Beal
Principal Engineer
Sun Microsystems



Sun Microsystems

- Founded in 1982
 - > Stanford University Network
- Large R&D for size
- Delivers
 - > Software
 - > Services
 - > Systems
 - > Microelectronics
- Guided by a singular vision
 - > "The Network is the Computer"

Solaris

- The most advanced Operating System
 - > Zfs
 - > Fault management
 - > Dtrace
 - > Containers
 - > Open Source
 - <http://www.opensolaris.org>
- Made up of many different projects
 - > Known as consolidations

Solaris

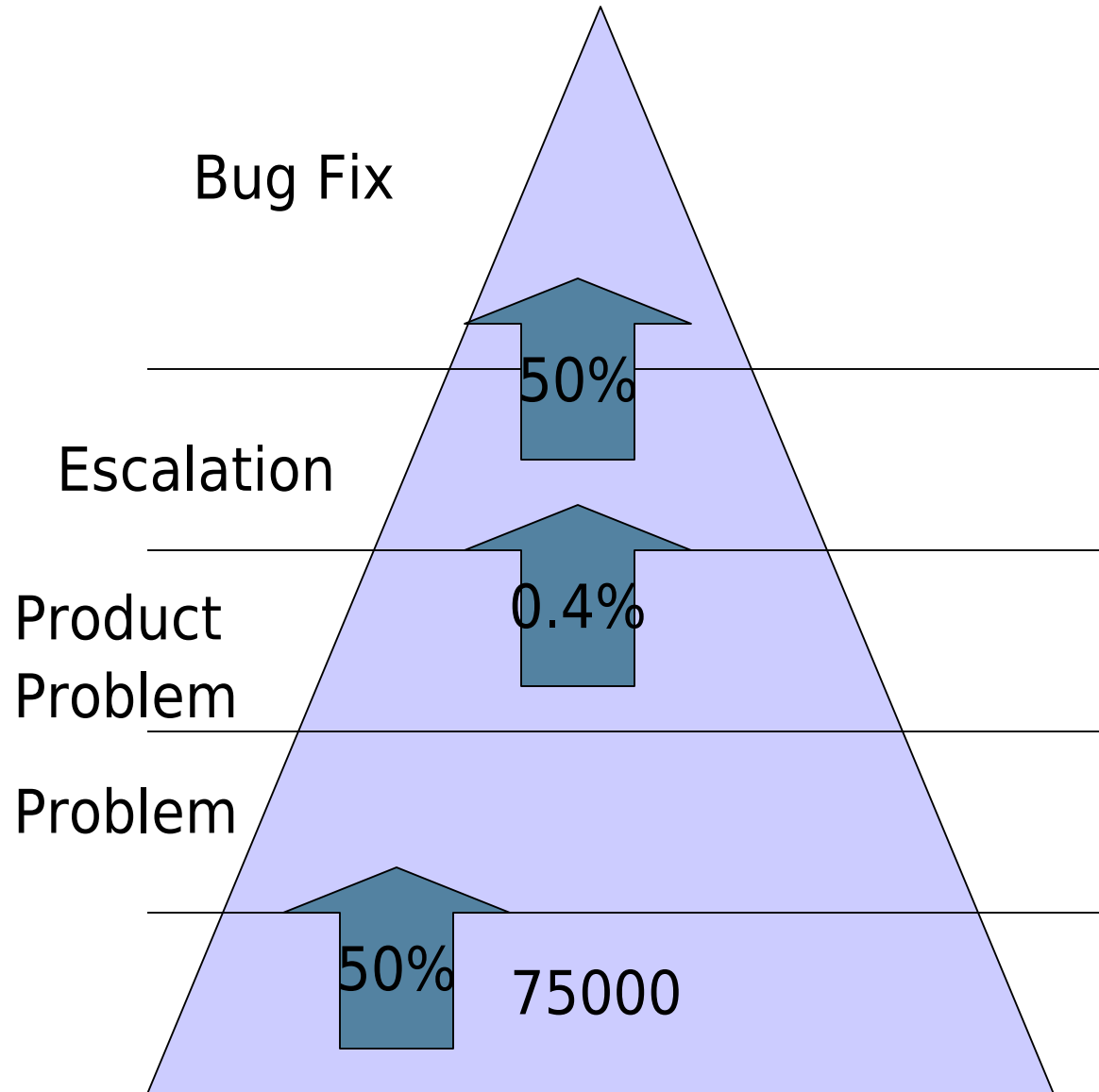
- Projects include
 - > Desktop: What you see
 - > Install: To get it installed
 - > X: The windowing system
 - > Docs
 - > Internationalization/Localization
 - > ON: Operating System and Networking

Solaris

- Multiple releases
 - > Opensolaris
 - Project nevada
 - <http://www.opensolaris.com>
 - > Solaris 10
 - 3-4 update releases a year
 - New features
 - Fixes delivered via patches
 - > Solaris 9
 - No new features
 - Fixes delivered via patches

Sun Support

- 5 tier support model
- More complex problems get passed up



One Consolidation: ON

- The core OS
 - > Kernel, networking, libraries, utilities
 - > Low Level platform code
 - > Multithreaded, complex interactions
- Size and complexity
 - > 15,555,000 lines
 - > 44324 files
- Contributors
 - > ~2000 people contribute

The Data Sets

- Defects are recorded in a database
 - > Bugs
 - > Request for enhancement (RFE)
 - > Thousands recorded over the years
- Escalations are recorded in a separate database
 - > < 0.1% of incoming calls result from a bug

The Problem

- It was noticed that about 1/6 of the incoming escalations were from a pre-existing bug
- If these can be fixed before customers hit them it will save both Sun and the Customer money

The Supposition

- Proactive bug fixing is good
- Saves money
 - > Each escalation cost \$32000
 - > Proactive fixing costs \$9000
 - To generate a patch
- Promotes quality
 - > Fewer customers hit problems
- Prevents damage to reputation
 - > Fewer escalations
- The key is fixing the right bugs

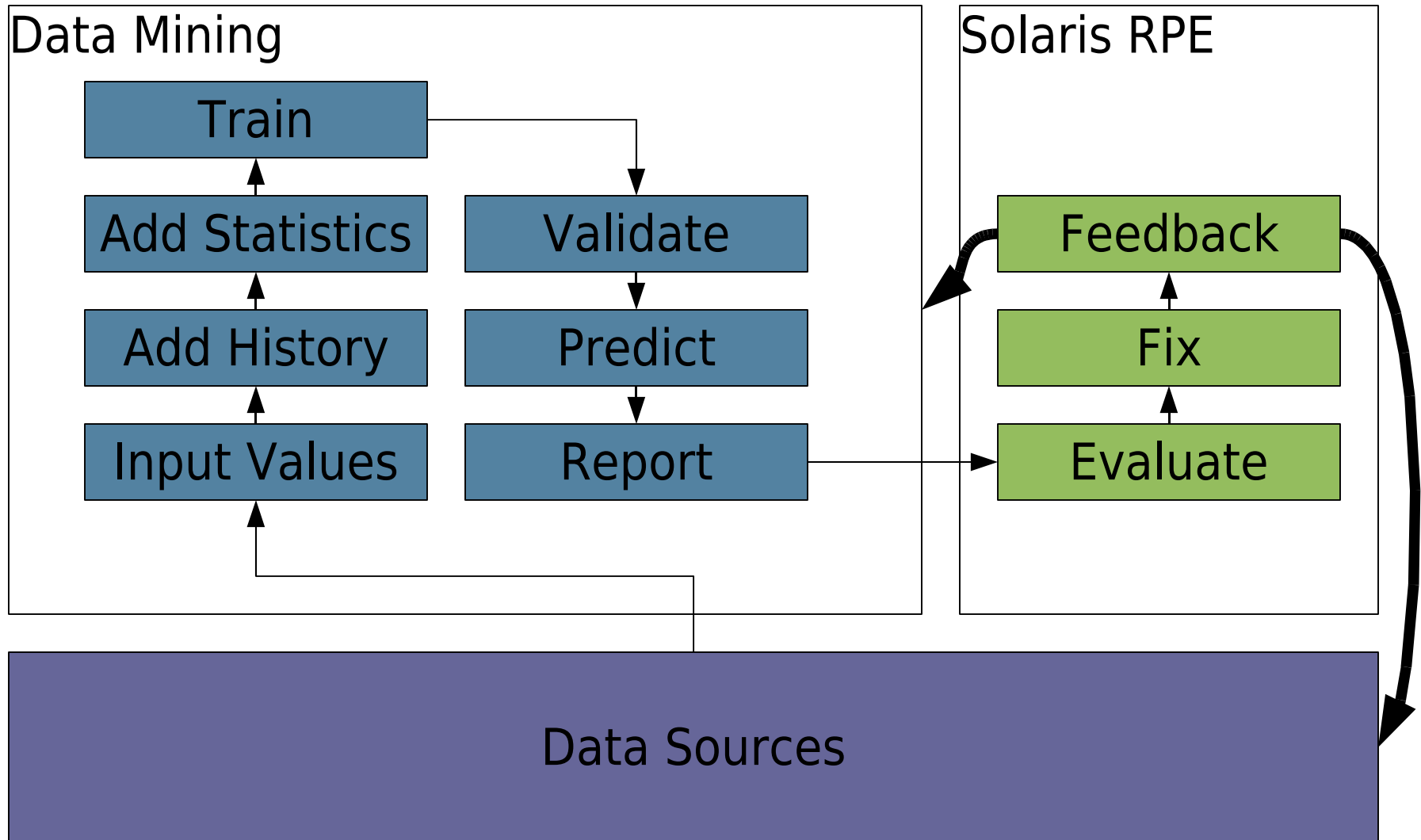
Too much work

- Escalation work is done by the Revenue Product Engineering team
 - > ~200 engineers
- Aim to fix the right bugs first
 - > High customer impact
 - > Low risk
- The key is identifying the right ones to fix
 - > How can we do this?

Escalation Prediction

- Use data a tool to learn which Bugs are likely to get escalated
 - > Commercially available data mining A/I used
 - Data captured from bugs and escalations
 - Pre processed to add historical data, and statistics
 - Train it on historical data,
 - Derive a suitable model
 - Validate results with real escalation data
- Run on current data
 - > Initially verify based on knowledge and experience
 - > Feedback for improvement to the model

The Model



The Model

- Data Mining uses a subset of the data points
 - > Numerical and range based field
 - > Fields with limited number of possibilities
 - Category and subcategory
 - Customer Name
 - Statistically corrected
 - > No text analysis

Sample Report

Open P1/2/3

BugID	Category / SubCat	Priority	Severity	Risk	State	Resp. Manager	Synopsis
6382236	kernel syscall	1	1	37.46%	Dispatched		During Explorer 5.2 captured, SF V8
6371753	driver e1000g	2	1	34.62%	Fix in Progress		aggregation of e1000g NICs fails to
6395124	kerberosv5_bund	2	2	34.39%	Cause Known		pam_krb5 tries to validate twice whe
6400068	consolidation os-	2	2	34.17%	Dispatched		Error message output while add pat
6402299	solaris_install flas	2	3	34.17%	Dispatched		S10 U1: Flash archive created while
6404167	consolidation os-	2	3	34.17%	Dispatched		Prepatch script in Solaris 10 X86 ker
6395110	solaris_install live	2	3	34.17%	Dispatched		Cannot install a flash archive via Live
6389053	kernel boot	1	2	29.25%	Fix Understood		Oracle RAC defines "uadmin 1 1" as
6382658	sma-net-snmp a	2	1	24.51%	Dispatched		SNMP Table creation with an invalid

The Implementation

- Identify resources to proactively work
 - > Solaris RPE
- Run a weekly report of Escalation Prediction
 - > Use knowledge and judgement to identify the most important
 - > Proactively Fix
 - > For false positives provide feedback to EP team
 - To improve the model

Successes

- ✓ 41 patches for EP fixes released
 - > Proactively preventing escalations
- ✓ Notable fixes include
 - > 4964867 boot.bin: boot panic requires console presence for reboot
 - Allows better diagnosis after a problem
 - > 4876829 console: bind with uninitialized sin6_scope_id
 - Identified a regression in a patch
 - > 6280143 lufs_enable() strikes again
 - Panic on boot

More Successes and some issues

- ✓ With training the model improved
- x Some bugs which could not be hit always reported
- x Slow cycle of learning
- x Too much reactive work to do
- x Lack of text analysis in the prediction
 - x Led to people mistrust results

Why we discontinued

- Resource crunch
 - > The teams identified to proactively work these suddenly was overloaded by reactive work
 - Success os Solaris 10
 - Reduction in force elsewhere moved more work to RPE
- Lack of expertise
 - > The author of the model left the company
 - > Difficult to retrain the model
 - Gradual decline in report quality

Why we discontinued

- Lack of a champion
 - > Existing executive champions moved on
 - > New executives struggled to grasp the model
- All this in a background of tough financial times
 - > Many people leaving Sun
 - > Reduction in force
 - > Focus on the present

Lessons Learned

- Ring Fence your resources
 - > If you have a team implementing your model then keep them safe
- Keep it simple
 - > Exec sponsors move on, make sure it can be resold
 - Without a champion funding will be challenging
- Keep improving it
 - > If it goes backwards it loses credibility
 - > Implementor buy in is key

What we do now

- Release criteria
 - > Review P1 and P2 bugs for fixing
- Incoming triage
 - > All new bugs reviewed to see if they are complete
 - ie. an analysis can be performed with the data provided
 - > Identify need to be fixed protectively

What we do now

- Work on getting bugs defined properly
 - > Use Kepner Tregoe based trouble shooting methodology
- Fix in update releases
 - > Allow all changes to be tested together
 - > Improves quality
 - > Regular release schedule allows for focused work

What are we considering next?

- Upgrading static analysis tools
 - > Better analysis of source code
- Performing run time checking
 - > Use tools to verify code correctness during execution
- Use of defect prediction?
 - > Should we be looking at ways of identifying code likely to contain bugs



Practical use of defect detection and prediction

Christopher Beal
chris.beal@sun.com