Defect Prediction for Large, Long-lived Software Systems

Elaine Weyuker
Tom Ostrand
AT&T Labs Research

WG10.4 Workshop on Experimental Computer Science
July 2-5, 2009 Obidos, Portugal

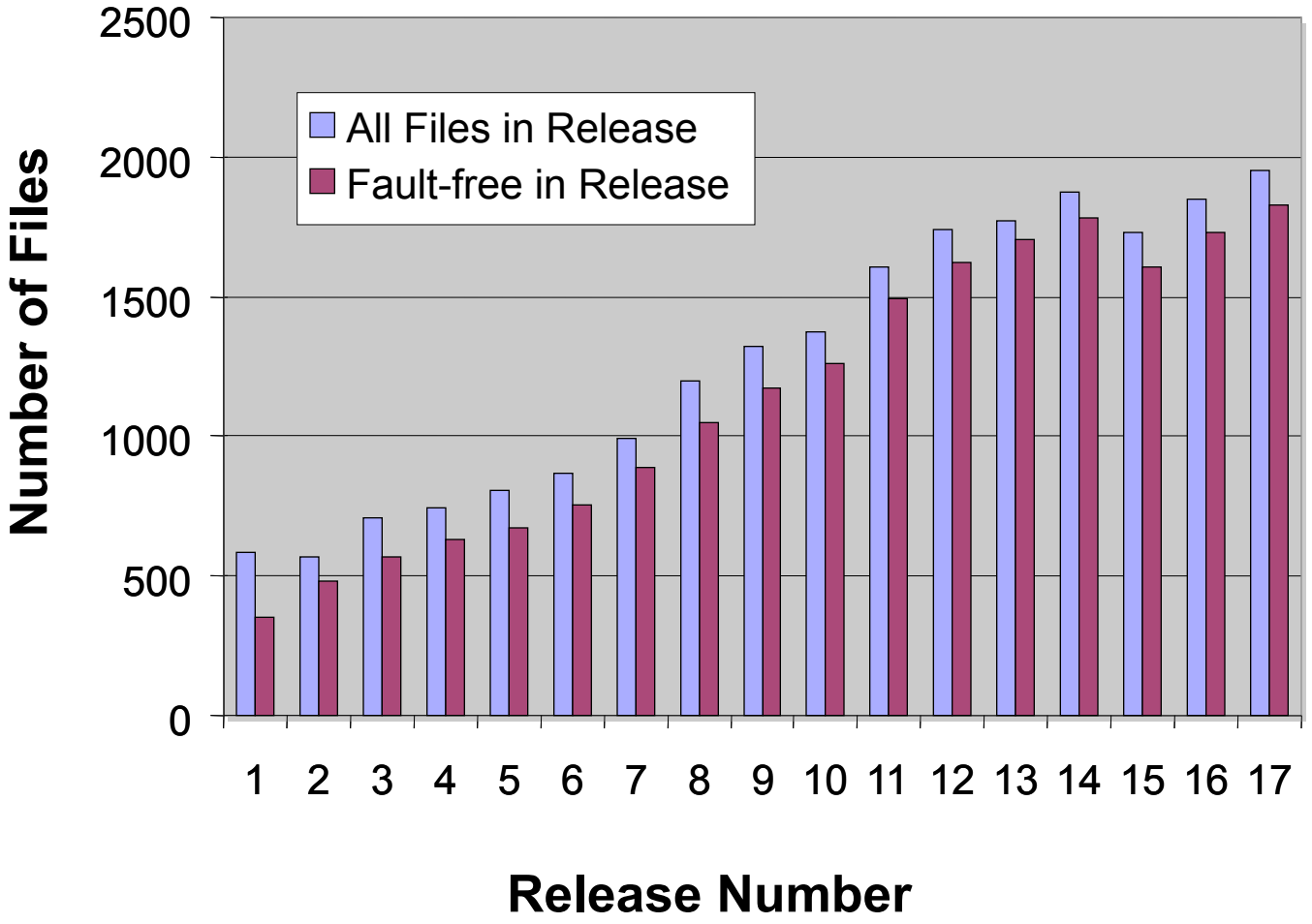
Outline

- Motivation
- Systems we've studied
- Making predictions
- Results
- A case study issue
- Current status

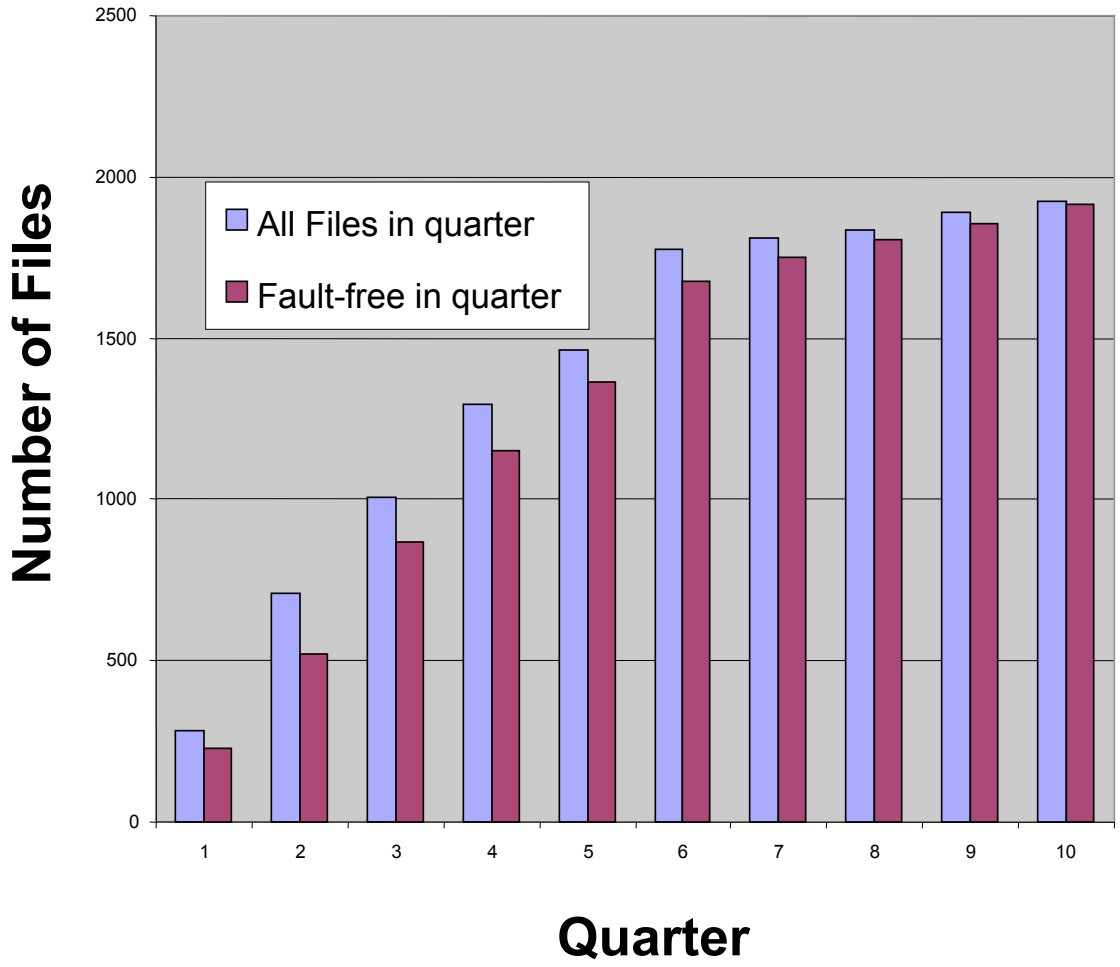
Motivation

- Goal: to determine which files of a large, long-lived system are most likely to contain faults in next release
- Faults are not uniformly distributed over files
- Faults are usually concentrated in a very small percentage of a system's files
- Knowing in advance which files are most likely to be faulty is a big advantage for system testers and developers

System N Fault-free Files in all Releases



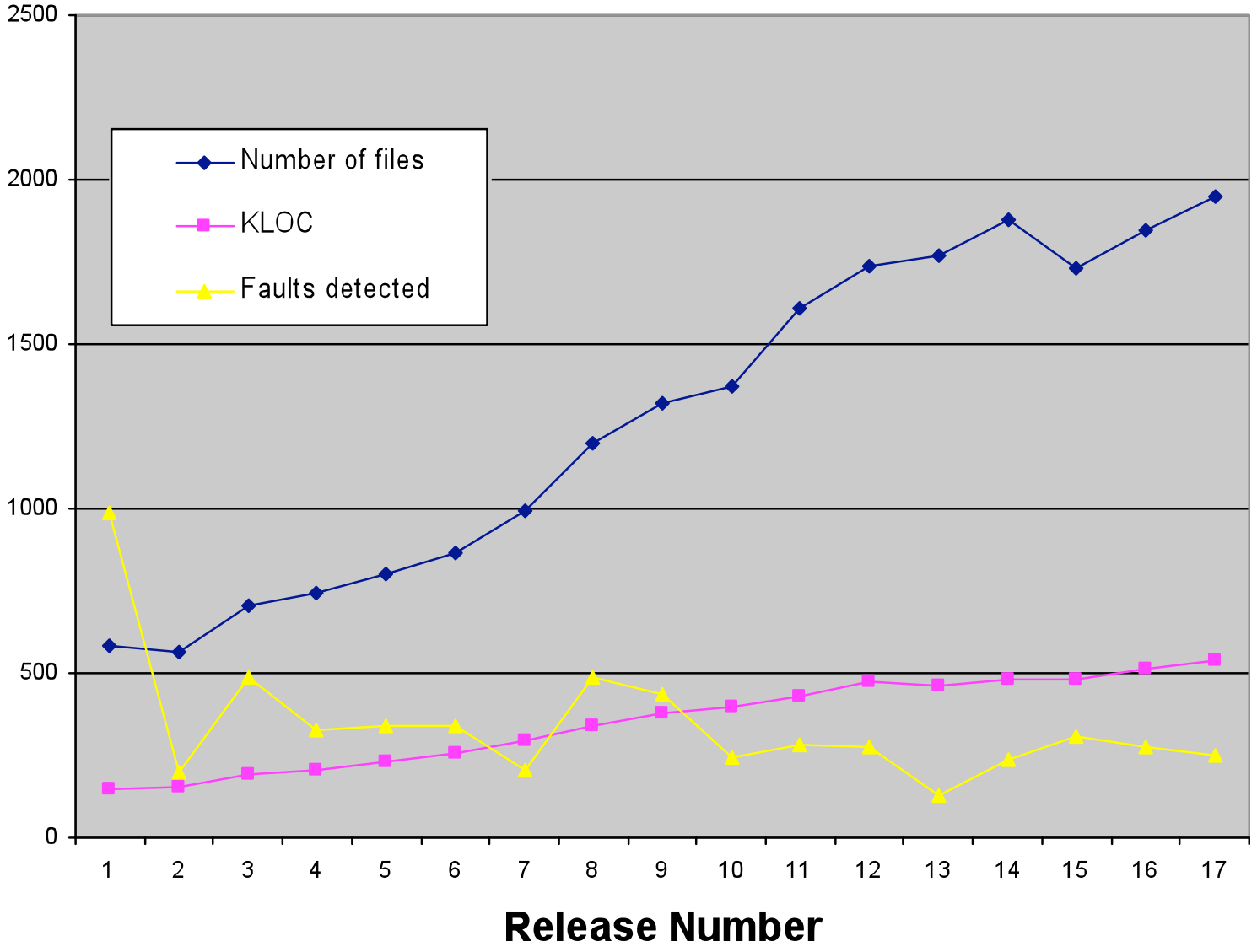
System V fault-free files in all quarters

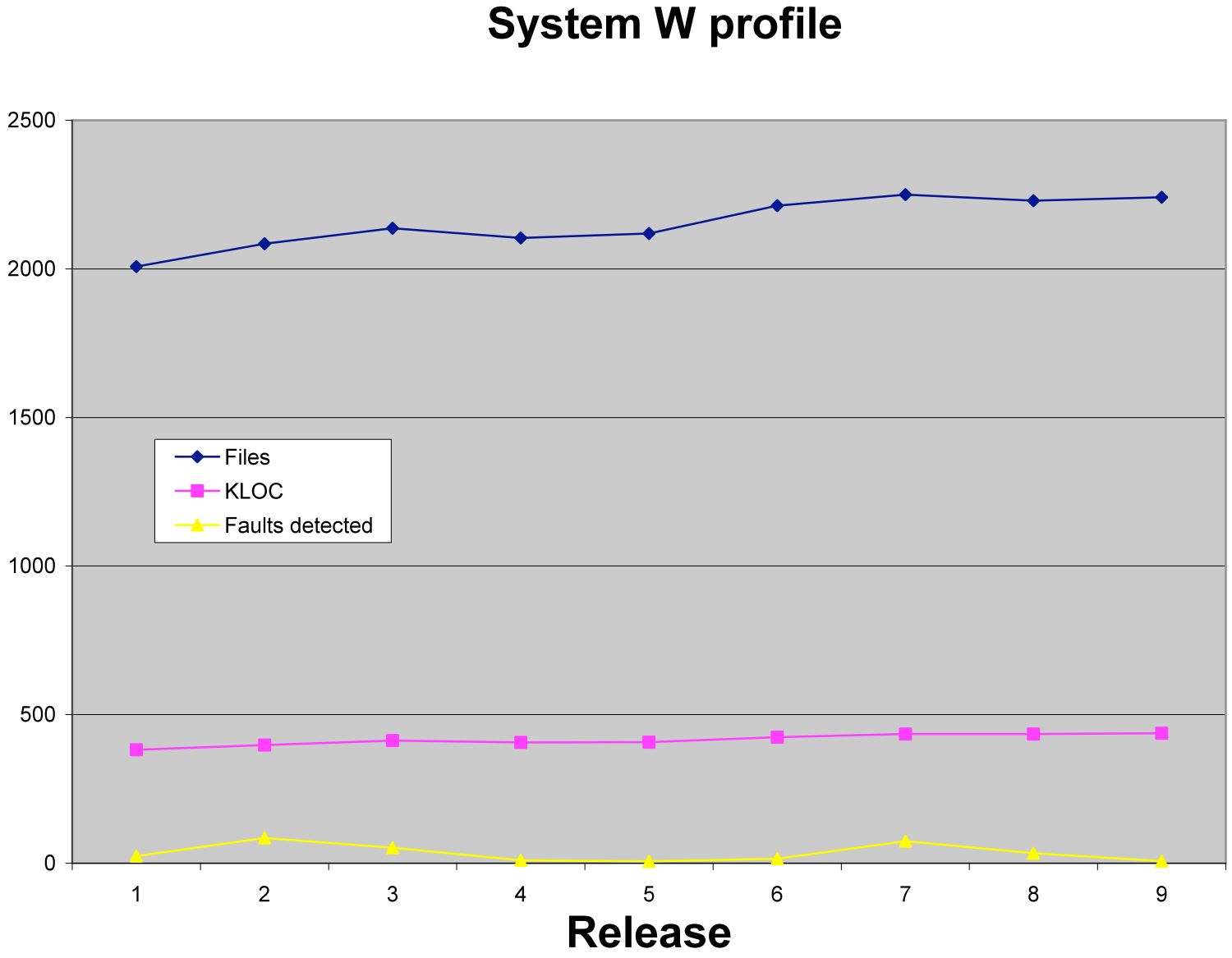


Systems we've studied

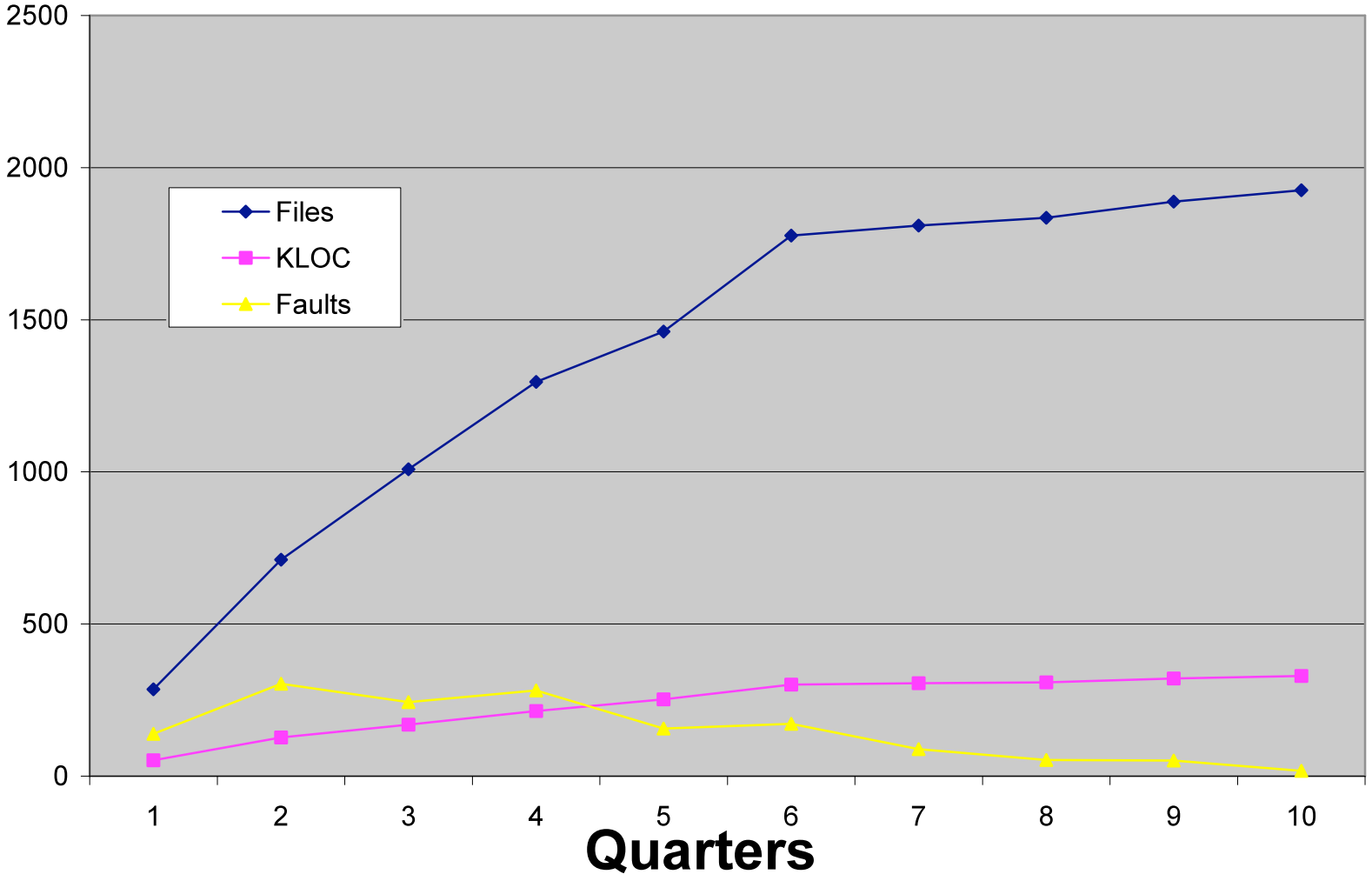
	Releases/ Lifetime	LOC in last release	Files in last release	Avg # Faults/Rel	Pct Faulty Files/Rel
N: Inventory	17/ 4 years	538,000	1950	342	4.0-39.9
W: Provisioning	9 (3)/ 2 years	439,000	2271	34	0.3-3.0
V: Voice response	---/ 2¼ years	329,000	1926	151 (per quarter)	0.5-27.0
Maintenance support BS	35/ 9 years	442,000	668	46	0.9-41.7
BW	35/ 9 years	384,000	1413	40	0.1-5.4
BE	28/ 7 years	329,000	584	48	0.2-13.5

System N Profile

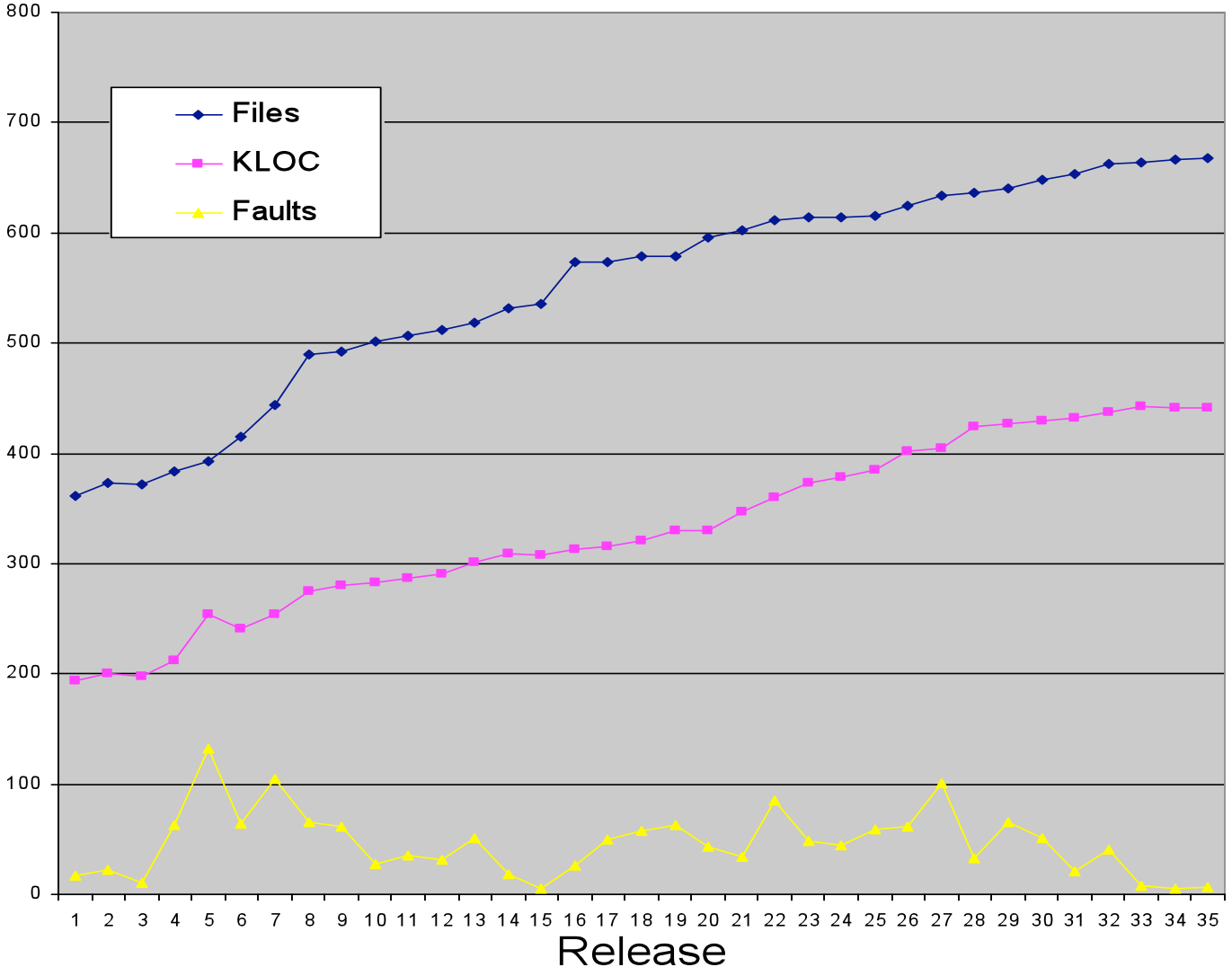



System W profile
2500
2000
1500
1000
500
0
Files
KLOC
Faults detected
1
2
3
4
5
6
7
8
9
Release

System V profile



System BS Profile



Basic attributes used for prediction

- KLOC
- Previous faults (n-1, n-2)
- Previous changes (n-1, n-2)
- File age
- File status (new, changed, unchanged)
- File type (C,C++,java,sql,make,sh,perl,...)

Additional attributes for prediction

- Developer count attributes
  - Number of developers (release n-1)
  - Number of new developers (release n-1)
  - Cumulative developers (releases 1:n-1)
- Calling structure attributes
  - calling files, called files
  - (new, changed, faulty)

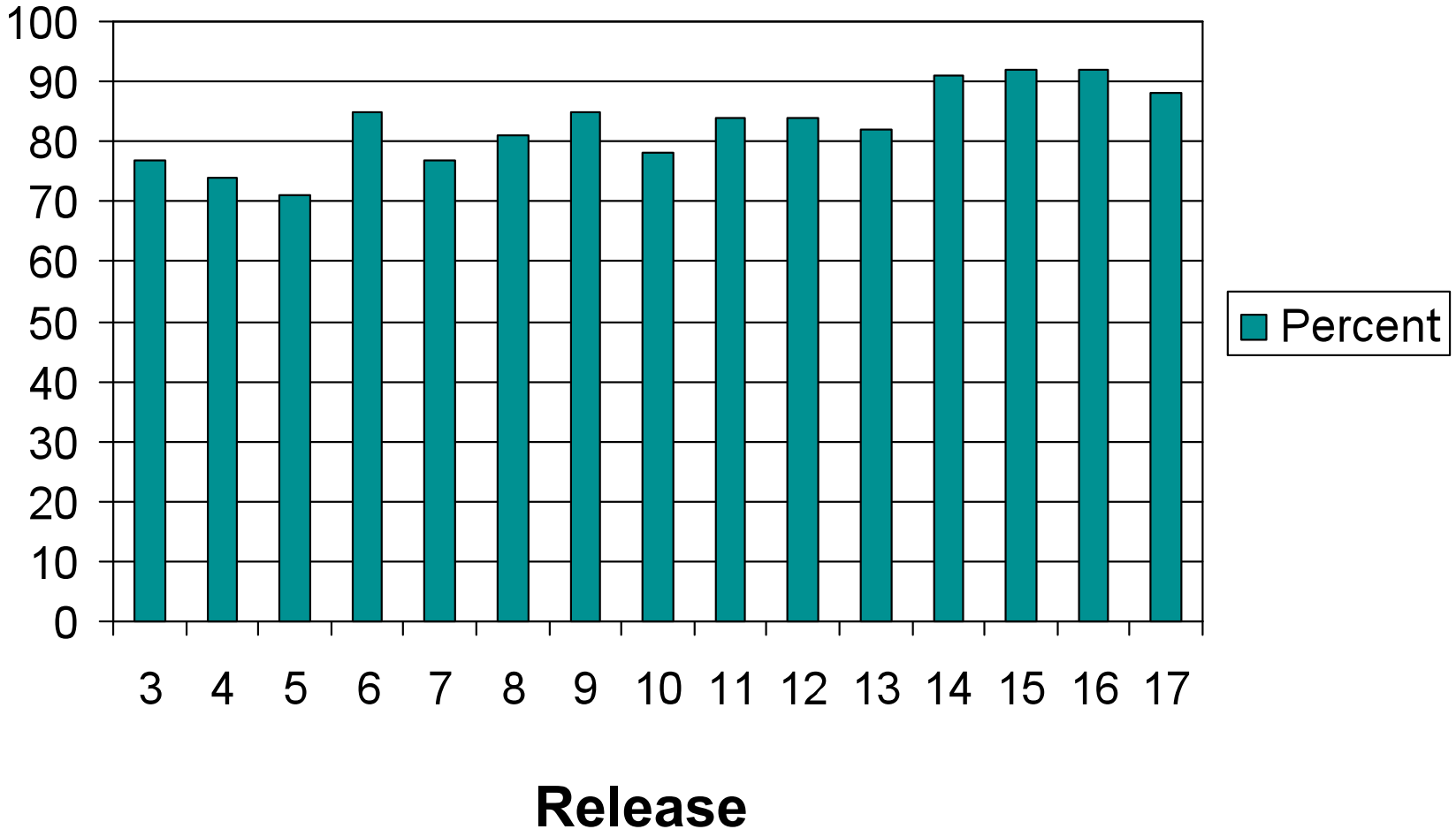
Statistical models used

- Negative binomial regression
- Recursive partitioning
- Random forests
- Bayesian additive regression trees

System N Results

- Negative binomial regression
- Basic attributes

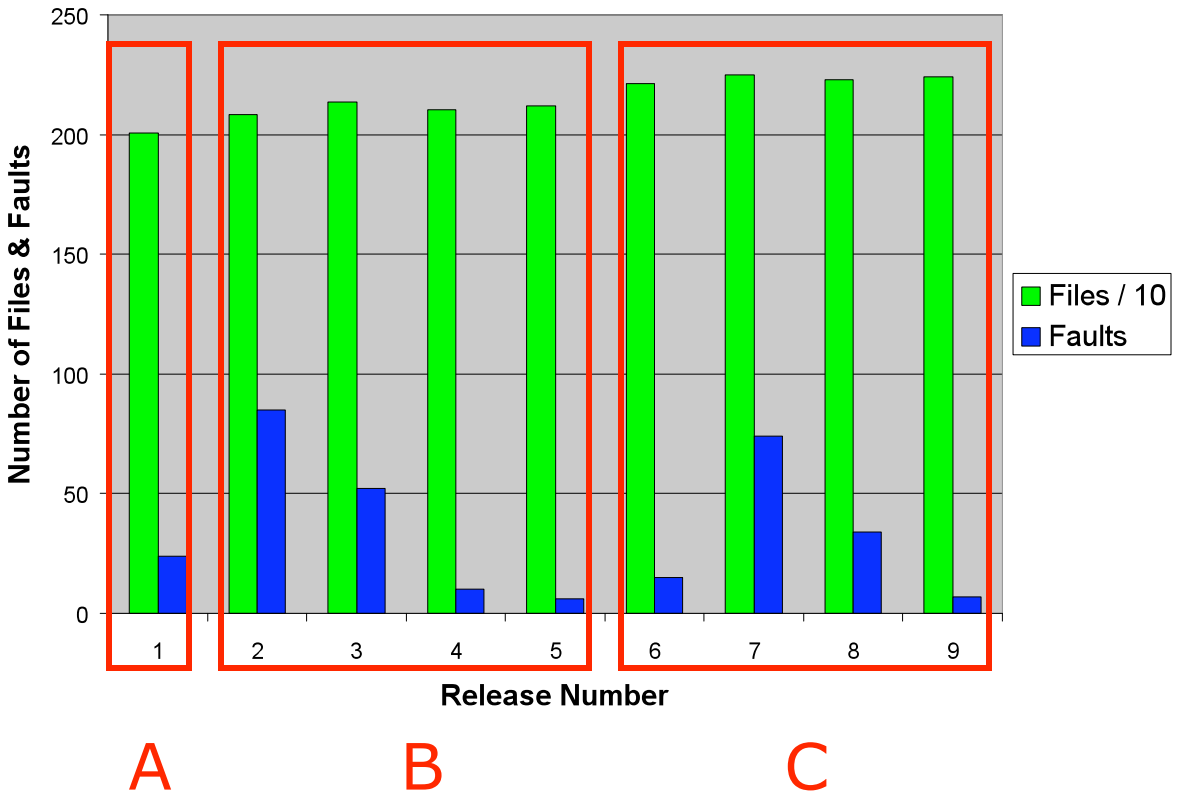
Percent of Faults Contained in Top 20% of Files Selected by Model (Average = 83%)



System W results

- Negative binomial model
- Basic attributes
- Low fault count made per-release predictions not possible

Number of Files and Faults by Release (System W)



System W Grouped Releases

"Release"	based on	# Faults
A	Release 1	24
B	Release 2-5	153
C	Release 6-9	130

Rel A used to establish file status in Rel B.

Rel B data used to make predictions for Rel C.

Predictions for Release C of System W

- Top 20% of files contain 83% of faults
- Top 10% of files contain 68% of faults

System V results

- Negative binomial model
- Basic attributes
- “Releases” are defined as consecutive 3-month periods.
- Top 20% of files contain 61% - 97% of faults, for quarters 3-9.
- Average is 75%

Summary of prediction results

System: Type	***Period Covered***	***Faults in 20% Files***
N: Inventory	4 years	83%
W: Provisioning	2 years	83%
V: Voice Response	2.25 years	75%
Maintenance Support Systems BS	9 years	84%
BW	7 years	93%
BE	9 years	76%

Collecting and Analyzing Data

All 6 projects use a common version control/ change management system

Every SW change is recorded in a detailed MR (modification request)

MRs: requested changes to software

- Date & release-id of request & changes
- Who requests the change
- Who makes the change
- Attributes of the request & change
- Lifecycle phase of request & change
- Specific files that are changed
- Natural language description

A Case Study Issue: What is a fault?

- based on attributes?
- based on life-cycle phase?
- based on size of the change?
- based on natural language description?

Which MRs are defects?

- Attributes
  - Category: action, issue, enhancement, modification, defect, other
  - Type: initialization, new feature, change to existing feature, fix existing feature
- Life-cycle: reqts, code, unit test, system test, integration test, UAT, ORT, introduction, customer use
- Fewer than N files modified (at least 1)
- Keywords in the description:
  - bug, fault, defect, fix

Which MRs are defects?

- Attributes
  - Category: action, issue, enhancement, modification, defect, other
  - Type: initialization, new feature, change to existing feature, fix existing feature
- Life-cycle: reqts, code, unit test, system test, integration test, UAT, ORT, introduction, customer use
- Fewer than 3 files modified (at least 1)
- Keywords in the description:
  - bug, fault, defect, fix

Status

- NBR model using basic attributes gives good results on a variety of systems
- Various supplements to basic attributes provide little or no improvement in accuracy
- GUI has been implemented to provide easy access to prediction model for users
- Next step: introduce model for use in existing large, long-lived AT&T systems

Comparison of models