Planning, Predicting and Demonstrating Software Quality and Reliability

V.A. French

Vern.French@sympatico.ca

Defect Removal Approach

Percent Defects In (This Phase)

+ Percent Defects Left (Previous Phase | All Previous Phases (I&T and O & M))

= Percent Defects Visible

x {100 - Percent Defects Removed (First Technique)}

= Percent Defects Remaining

x {100 - Percent Defects Removed (Second Technique)}

= Percent Defects Remaining

.

.

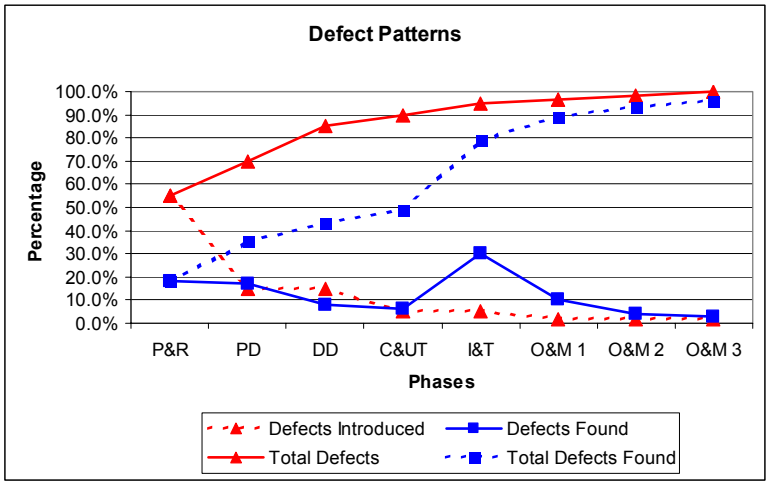
x {100 - Percent Defects Removed (Last Technique)}

= Percent Defects Left (This Phase)

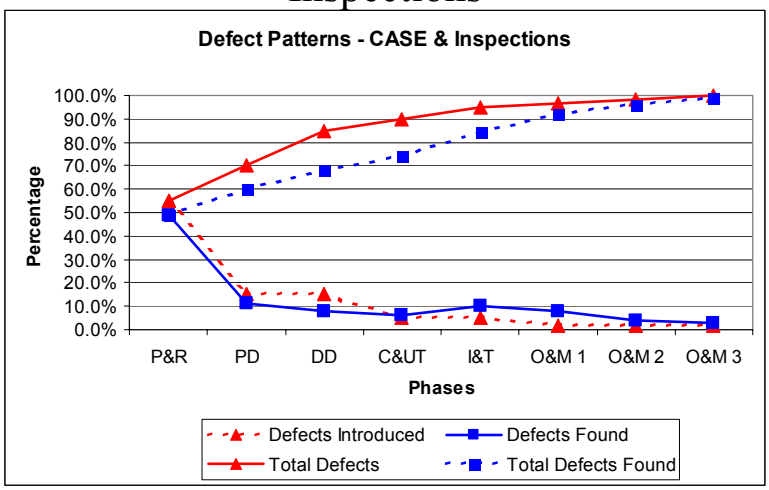
Defect Introduction and Removal Model

	% of Defects Introduced		% of Defects Detected				
Development Phase	Normal	Reuse	Review & Testing	Inspections	Metrics	CASE	IV&V
Plans & Requirements	55%	55%	33%	60%	0	60%	0
Preliminary Design	15%	15%	33%	60%	0	60%	0
Detailed Design	15%	1.5%	33%	60%	80%	60%	0
Code & Unit Test	5%	.5%	40%	70%	70%	0	0
Integration & Test	5%	5%	60%	0	0	0	45%
Operations & Maintenance	5%	5%	100%	0	0	0	0

Defects Introduced and Found



Defects Introduced and Removed – CASE & Inspections

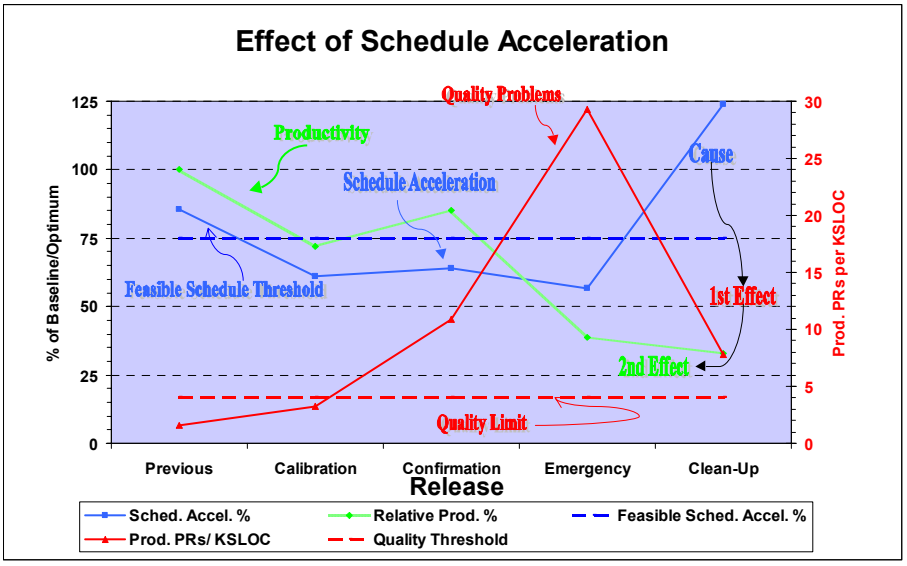


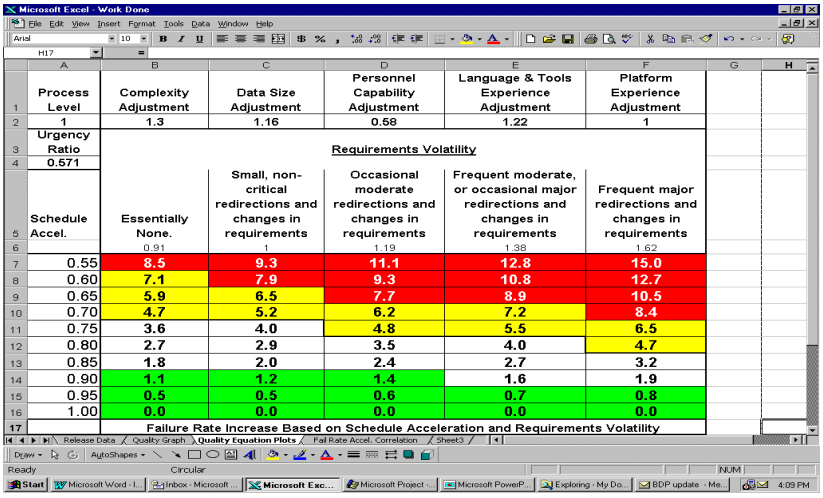
Factors Affecting Total Defects Introduced

- Primary Factors
  - Process Maturity and Domain Expertise
  - Schedule Acceleration (% of Nominal Schedule)
  - Requirements Volatility
- Other Factors
  - Complexity
  - Personnel Capability
  - Data Size
  - Language and Tools Experience
  - Platform Experience

Process Maturity and Domain Expertise

Process Maturity	Domain Expertise		
	Poor	Average	Good
CMM Level 1	50	40	30
CMM Level 2	24	18	12
CMM Level 3+	20	14	8





Process Level	Complexity Adjustment	Data Size Adjustment	Personnel Capability Adjustment	Language & Tools Experience Adjustment	Platform Experience Adjustment
1	1.3	1.16	0.58	1.22	1

Urgency Ratio	Requirements Volatility				
0.571					
Schedule Accel.	Essentially None.	Small, non-critical redirections and changes in requirements	Occasional moderate redirections and changes in requirements	Frequent moderate, or occasional major redirections and changes in requirements	Frequent major redirections and changes in requirements
	0.91	1	1.19	1.38	1.62
0.55	8.5	9.3	11.1	12.8	15.0
0.60	7.1	7.9	9.3	10.8	12.7
0.65	5.9	6.5	7.7	8.9	10.5
0.70	4.7	5.2	6.2	7.2	8.4
0.75	3.6	4.0	4.8	5.5	6.5
0.80	2.7	2.9	3.5	4.0	4.7
0.85	1.8	2.0	2.4	2.7	3.2
0.90	1.1	1.2	1.4	1.6	1.9
0.95	0.5	0.5	0.6	0.7	0.8
1.00	0.0	0.0	0.0	0.0	0.0

Failure Rate Increase Based on Schedule Acceleration and Requirements Volatility

Failure Rates

Reliability Model

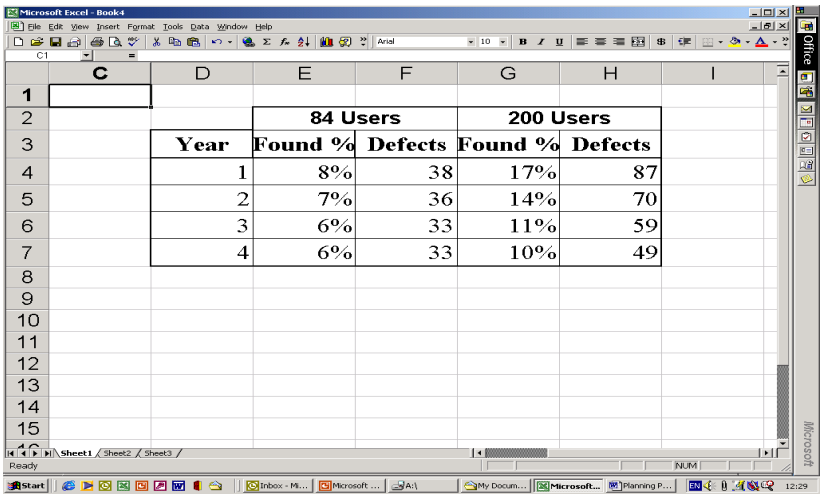
Defects Detected = Size * Defect Density * Found %

Found % = $(1-e^{-Failure\_Rate*Time})$

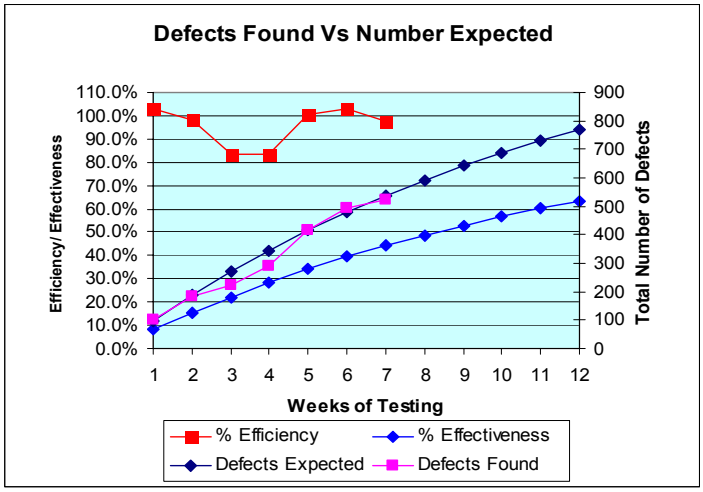
Time = Years Used * Users *Utilization

Failure Rate specific to system/environment
= 0.0038

Defects Detected



		84 Users		200 Users	
	Year	Found %	Defects	Found %	Defects
4	1	8%	38	17%	87
5	2	7%	36	14%	70
6	3	6%	33	11%	59
7	4	6%	33	10%	49



System Testing Results

Conclusions

- The number of defects introduced can be predicted
- The number of defects found can be predicted based on the development techniques used
- The quality and reliability of software can be predicted, measured and tracked

Presentation Title: Planning, Predicting and Demonstrating Software Quality and Reliability

Presenter: Vern French, XWave

Presentation Abstract:
Predicting software quality and reliability has been at best a black art. Few can plan and predict software quality and often the quality and reliability of software is not truly known until it has been released and in use for some time. Over the past ten years, the presenter has been developing techniques for estimating the number of software defects introduced and removed during software development and measuring software quality during system and acceptance testing. These techniques are now reaching the point where they can be used to plan and predict software defect rates and demonstrate that the required software quality and reliability has been attained before the software is released. The presenter will brief the audience on the techniques, provide some results obtained to date and some lessons learned along the way.

Brief Bio:
Mr. French has over ten years experience in the management, maintenance, development and testing of aircraft software and an additional ten years experience in systems engineering, software process engineering, software quality assurance, software configuration management, software release management and planning, Computer Aided Software Engineering (CASE)/Integrated Project Support Environment (IPSE) tool setup/integration and software development in embedded and object oriented systems environments. He also has seven years of aircraft systems/hardware and maintenance planning experience including avionics, armament and mechanical systems experience on sophisticated aircraft.

Mr. French is an expert in software estimation, planning, tracking, measurement and oversight. He has been estimating and planning software projects for 18 years and measuring and tracking such projects for 12 years. He was the principal author of the Canadian Air Force Policy and Procedures for Air Weapons System Software Support and the consultant to the Treasury Board Enhanced Management Framework (EMF) software Project Planning, Tracking and Oversight (PPTO) working group. Mr. French has developed detailed plans for and tracked progress on a massive, object oriented development project. He has also managed an SEI-CMM level 4 project, calibrated estimation equations and a software reliability model and used the reliability model to measure system testing effectiveness.

Mr. French has written and presented papers entitled "Applying Software Engineering and Process Improvement to Legacy Defence System Maintenance: An Experience Report", "Extending Cost Models to Model Software

Quality and Life Cycle Costs", "Establishing Software Metric Thresholds" and "Getting Creative - Measurement Without Expensive Tools" at the 1995 IEEE Conference on Software Maintenance, the ASQC's 6th International Conference on Software Quality in 1996, the 9th International Workshop on Software Measurement in Sept 1999 and the 11th International Workshop of Software Measurement in August 2001 respectively.