ADVANCE:
Implementing a Defect Model for Performance Prediction

Stan Martin, P.E.
Principal Systems Engineer
Modeling & Simulation/Operations Analysis
Greenville, TX
stan.martin@L-3com.com

L-3 communications
Integrated Systems
CMMI®
Integrating Best Practices To Achieve Excellence

Overview of "ADVANCE"

Å Stochastic model

Å Includes Defect Creation, Spread, and Detection

Å Includes Effects of Rework

Å Based on Historic Company Performance

Å Uses Company-specific Key Process Attributes

Å Predicts Defects (mean, ů) per phase & total

Outline

Å Basis and Previous Work

Å Implementation

Å Calibration

Å Deployment

Å Conclusions and Future Work

Å References

Outline

- **Basis and Previous Work**
- Implementation
- Calibration
- Deployment
- Conclusions and Future Work
- References

Basis: Requirements

- OPP-SP 1.5 requires Process Performance Models (PPM) to estimate or predict the value of a process-performance measure from the values of other process, product, and service measurements
- Chrissis, et.al., list four uses of PPMs:
  1. *The organization uses them for estimating, analyzing, and predicting the process performance associated with the processes in the organization's set of standard processes.*
  2. *The organization uses them to assess the (potential) return on investment for process improvement activities.*
  3. *Projects use them for estimating, analyzing, and predicting the process performance for their defined processes.*
  4. *Projects use them for selecting processes or subprocesses for use.*

Basis: Summary of Approach

Å The technical approach is based on established methodologies for reliability prediction of software defect densities:

- ï Historical data is used to produce a process performance baseline
- ï The process performance baseline is characterized by key process attributes
  - Å Specific attributes are defined for each lifecycle stage
- ï At the beginning of the project, the model will predict process performance based on the defined attribute values
- ï At the end of each lifecycle phase, predicted phase attributes are replaced by actual phase attributes, actual phase results are entered, and the remaining lifecycle performance is updated

Basis: Previous Work

Å Rout and Abel 1993 ñA Model for Defect Insertion and Detection in Software Developmentò

Å Chulani 1999 ñConstructive Quality Modeling for Defect Density Prediction: COQUALMOò

Å Others (see paper or References)

Basis: Rout & Abel Framework

1. ñErrors are introduced into a software component at each stage of development, the rate of insertion being dependent on a number of factorsò
2. ñErrors are detected and removed during all stages of development, at a rate which is primarily dependent on the detection technique employedò
3. ñErrors that are not detected during one stage of development may result in multiple errors in succeeding stagesò

Basis: Rout & Abel Framework

4. ñWhen an error which results from an error in a preceding stage is detected, all related errors are not necessarily detectedò
5. ñWhen an error is corrected, there is a non-zero probability that new errors will be introducedò

Basis: Rout & Abel Framework

- Difficulty: Rout and Abel were unable to represent framework points 4 and 5 in an analytic equation.

Solution: use a discrete-event model to represent processes without an analytic equation

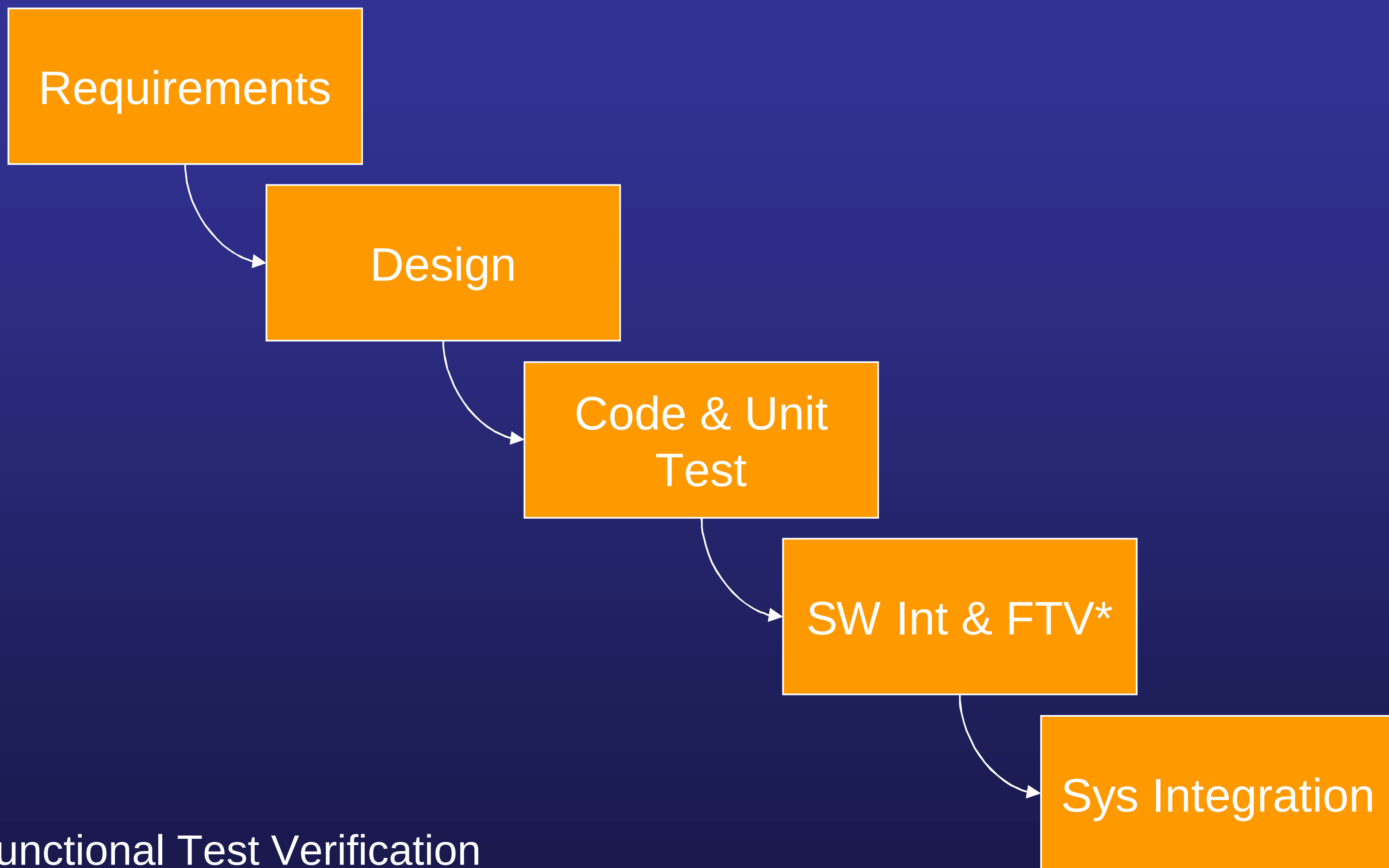
Basis: Chulani COQUALMO

- Chulani utilized the COCOMO production function:

$$E = a(S)^b$$

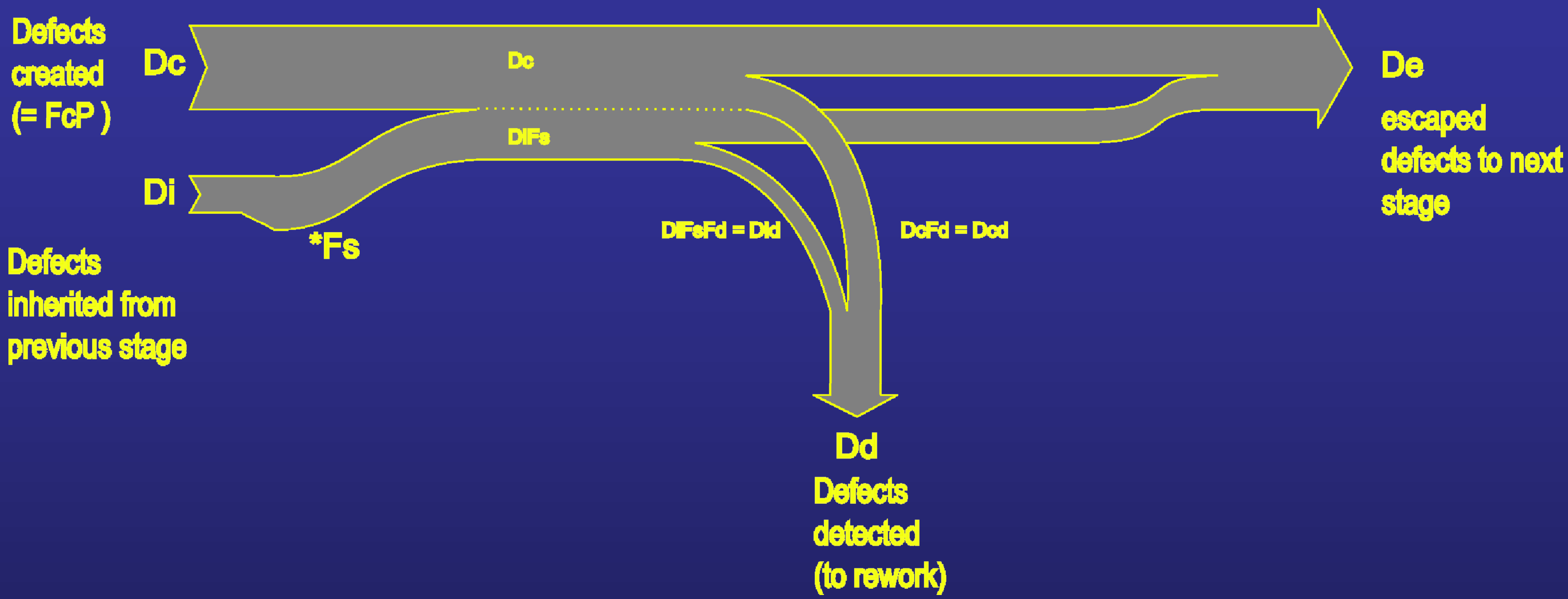
where
S = size (SLOC),
a and b are empirically derived,
a is a product of ñQuality Adjustment Factorsò

Basis: SW Development Phases



* Functional Test Verification

Basis: Defect Model (Per Phase)



Multiplicative Drive Factors
Fc = Creation (as function of phase Products P)
Fs = Spread (of inherited defects)
Fd = Detection (as % of phase total defects)

Basis: Three Factors per Phase

Å Defect Creation:

$$Dc = Fc * f(P)$$

Å Defect Inheritance:

$$Di_{adj} = Di * Fs$$

Å Defect Detection:

$$Dd = Fd * (Dc + Di * Fs)$$

Å 3 Factors x 5 Phases = 15 Factors

Outline

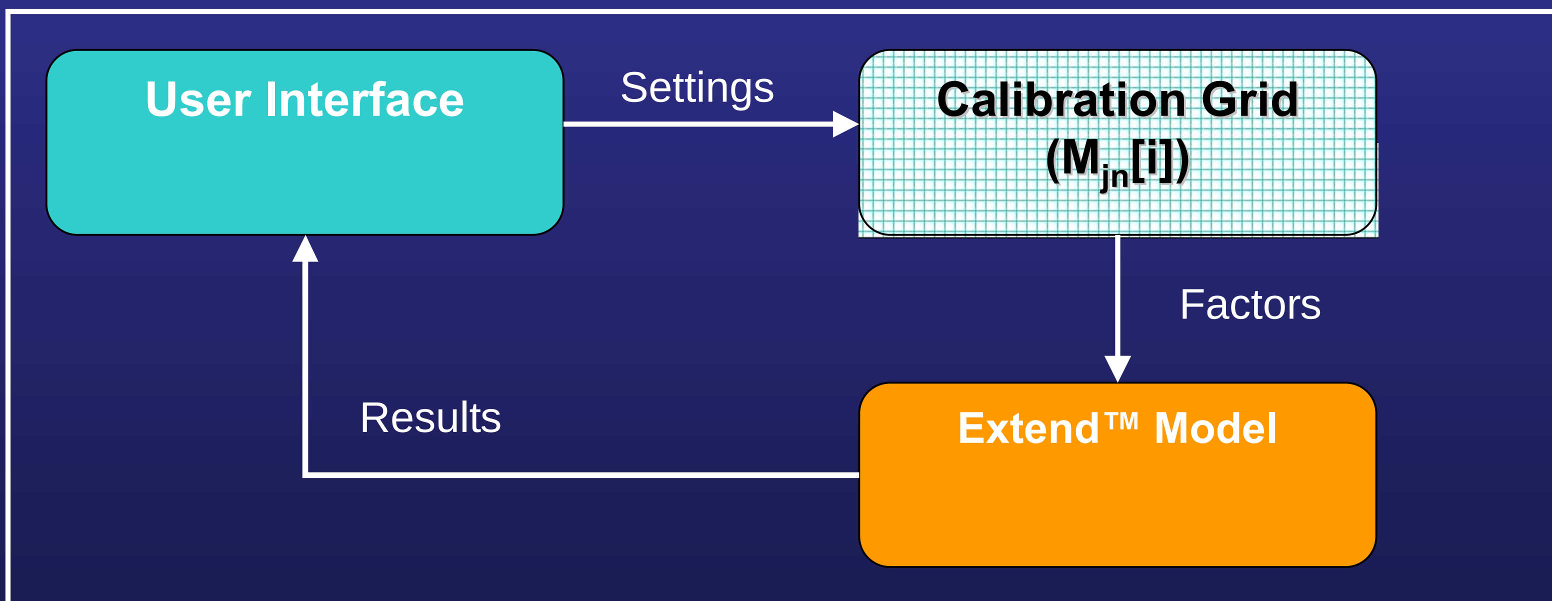
- Å Basis and Previous Work
- Å **Implementation**
- Å Calibration
- Å Deployment
- Å Conclusions and Future Work
- Å References

mplement: 3 Parts

Å ExtendÊ Discrete-Event model

Å User Interface (Microsoft Excel)

Å Calibration Grid (Microsoft Excel)



(trademark of Imagine That, Inc. http://www.imaginethatinc.com)

Implement: Drive Factors

$$F_n[i] = F_{nom}[i] \prod_{j=0}^{k} M_{jn}[i] \qquad n = \{C, I, D\}$$

Where:

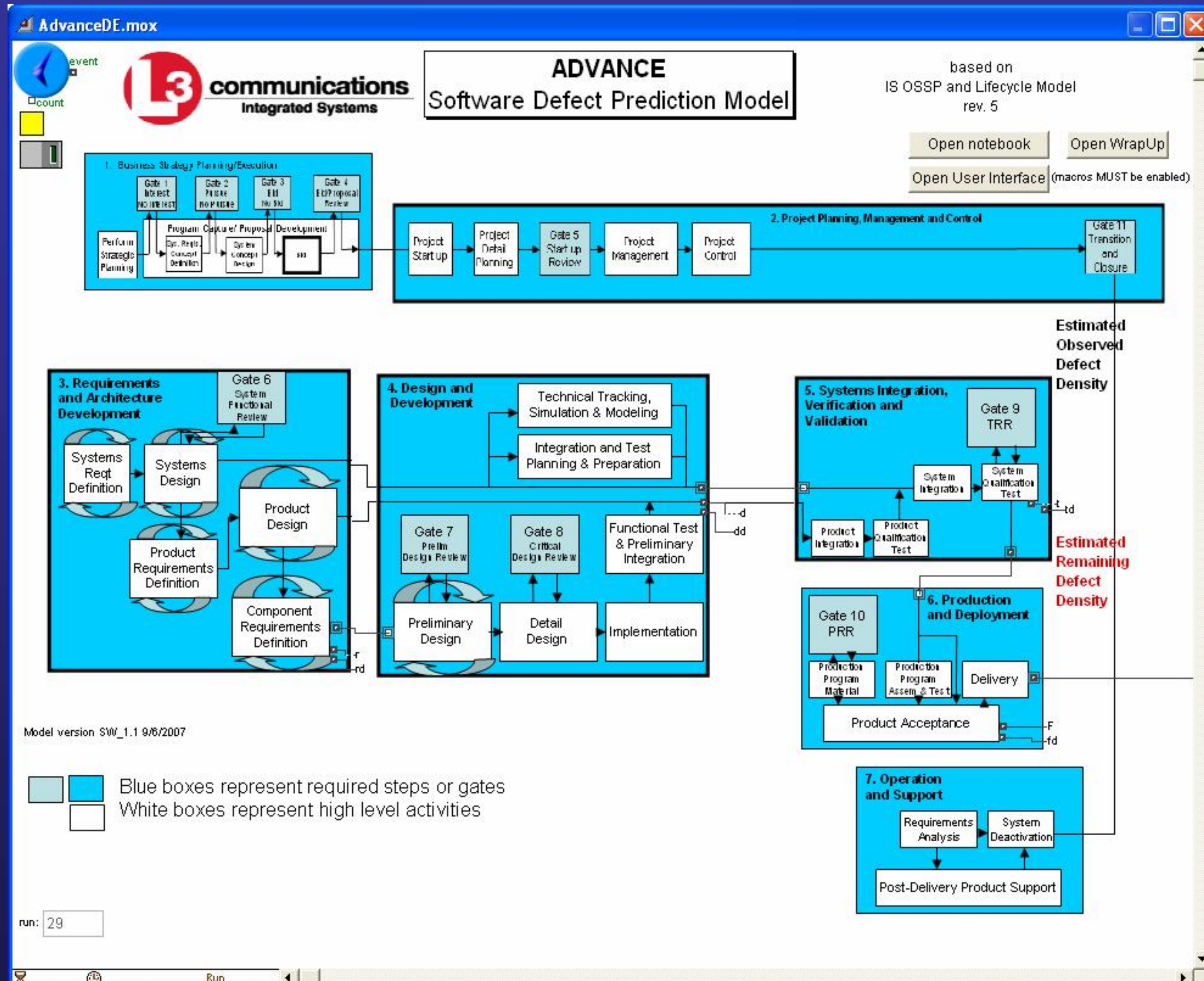
Å $M_{jn}[i]$ is the Model Drive Factor for attribute j as applied to factor n for phase i.

Å $F_{nom}[i]$ is a nominal value for $F_n[i]$

Å Similar to use by Chulani in COQUALMO

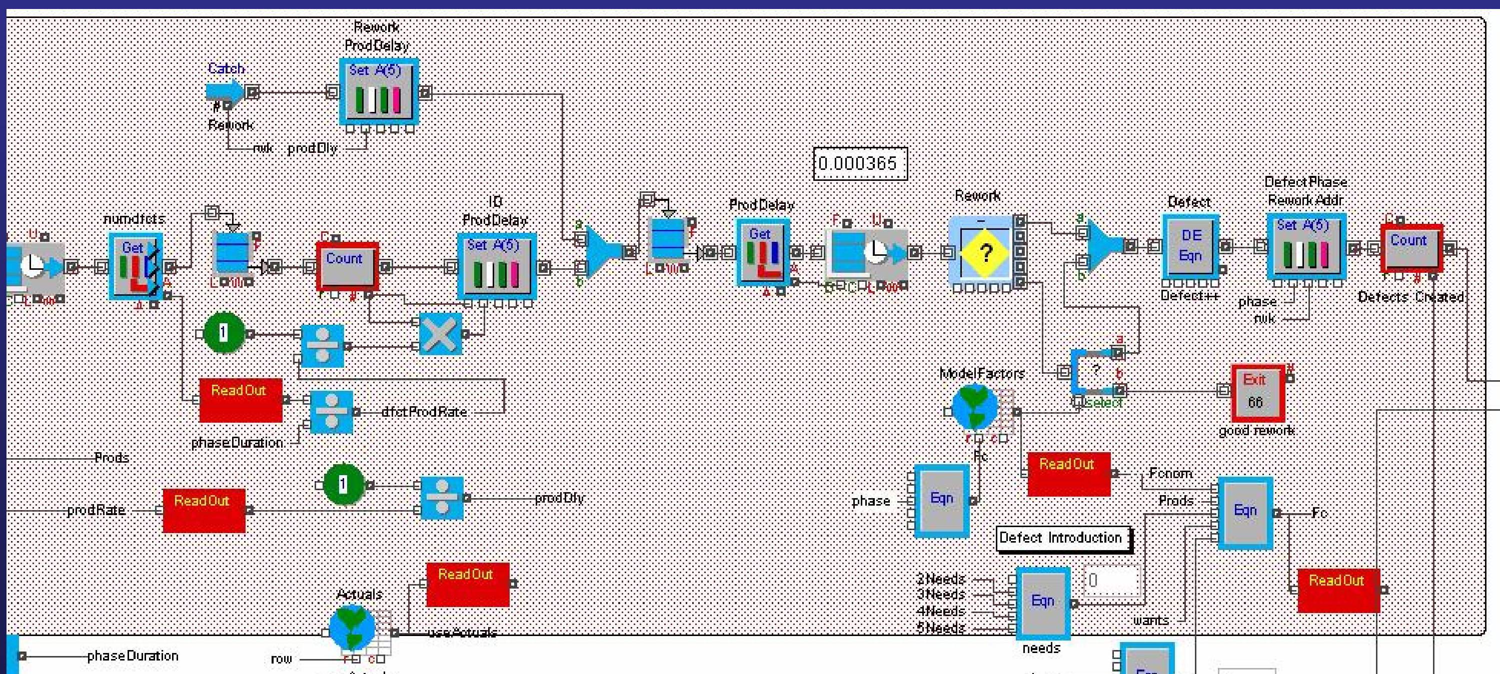
mplement: Extend Model

- Commercial process modeling tool
- Graphical, Hierarchical
- Discrete Event



mplement: e.g. Defect Creation

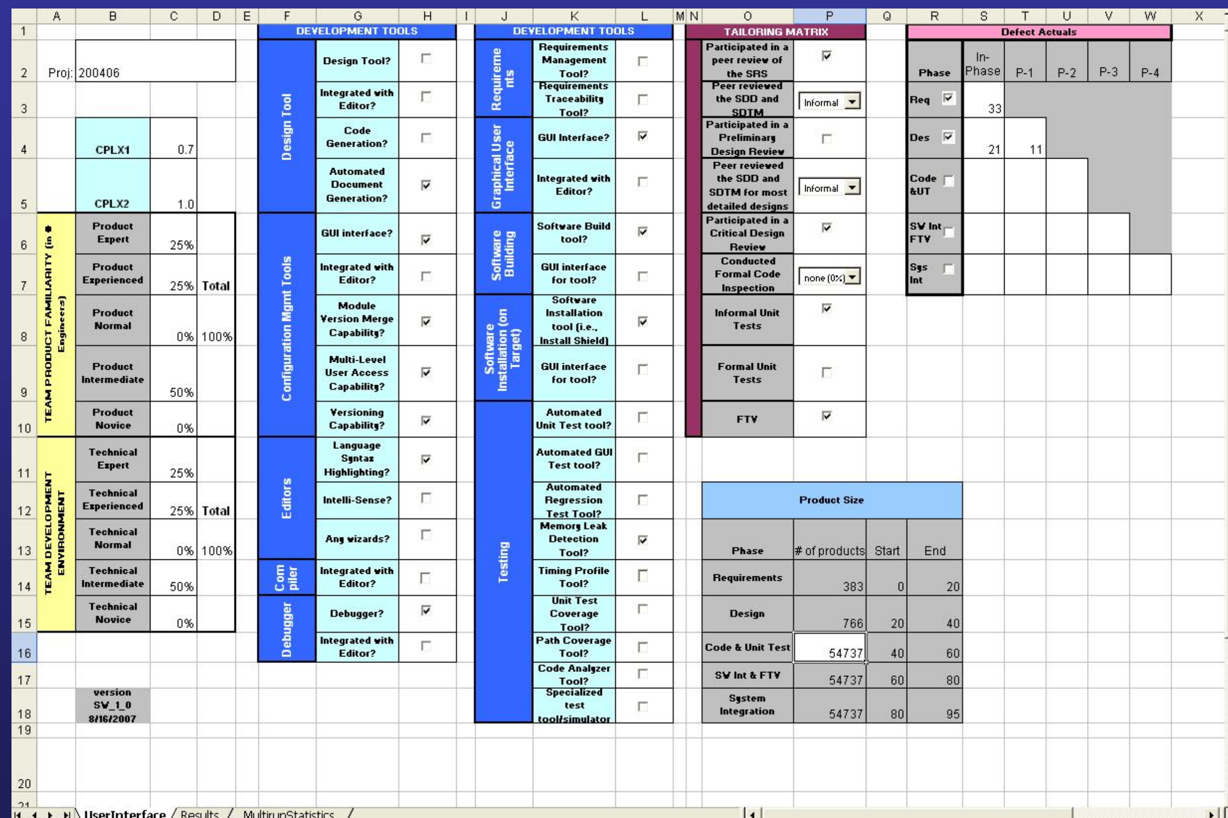
- Å Items flow through process network
- Å Queuing, time delay, decisions
- Å Numeric calculations, statistical distributions



mplement: User Interface

Å VBA controls for user inputs

Å Receives Extend model outputs

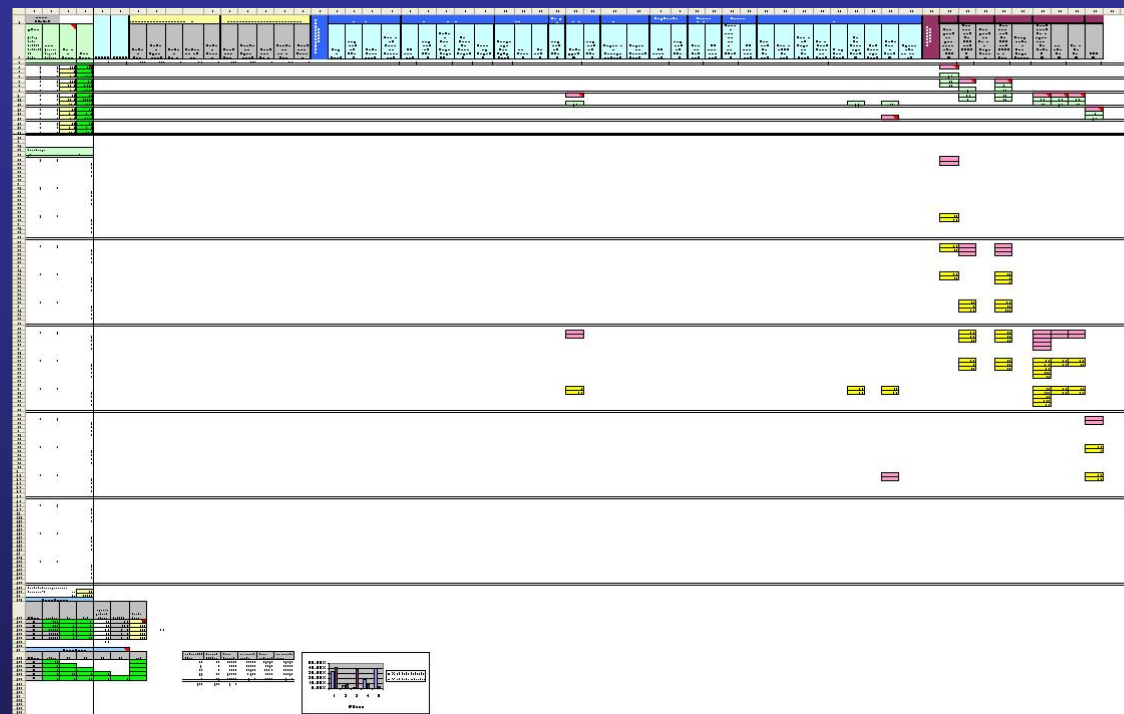


mplement: Calibration Grid

Å Converts UI settings to Model Drive Factors

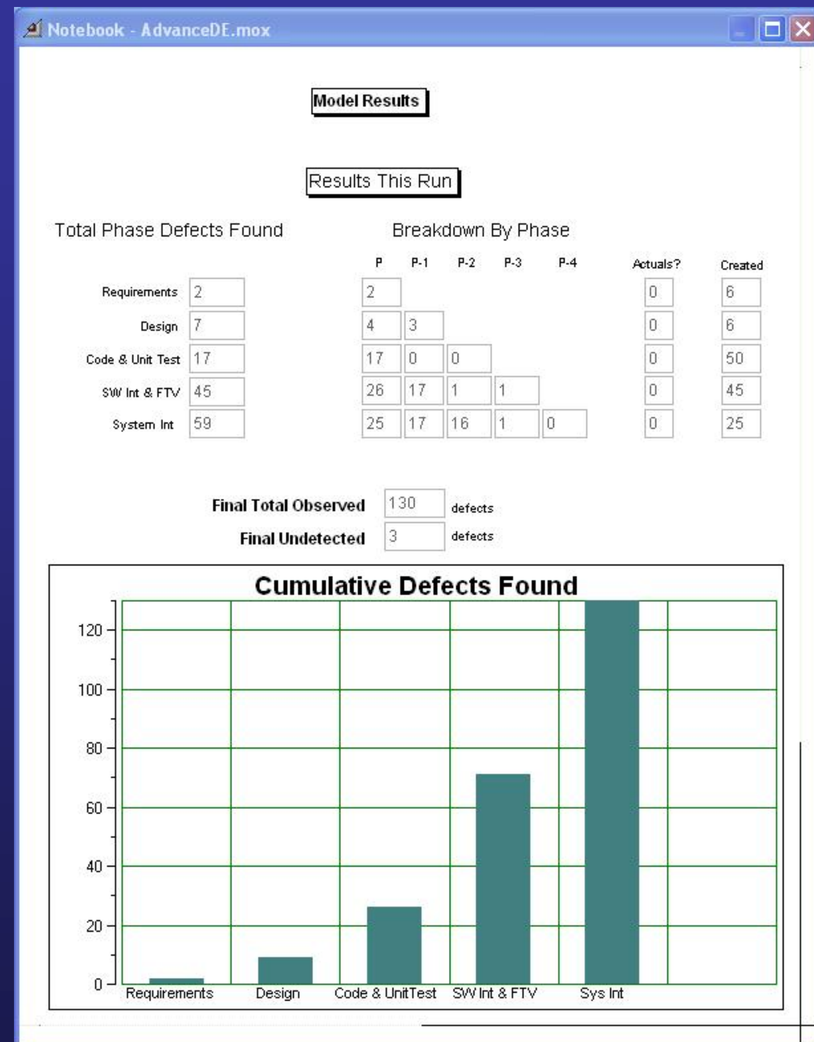
Å Isolates Extend model from calibration

Å Does not contain actual historical data



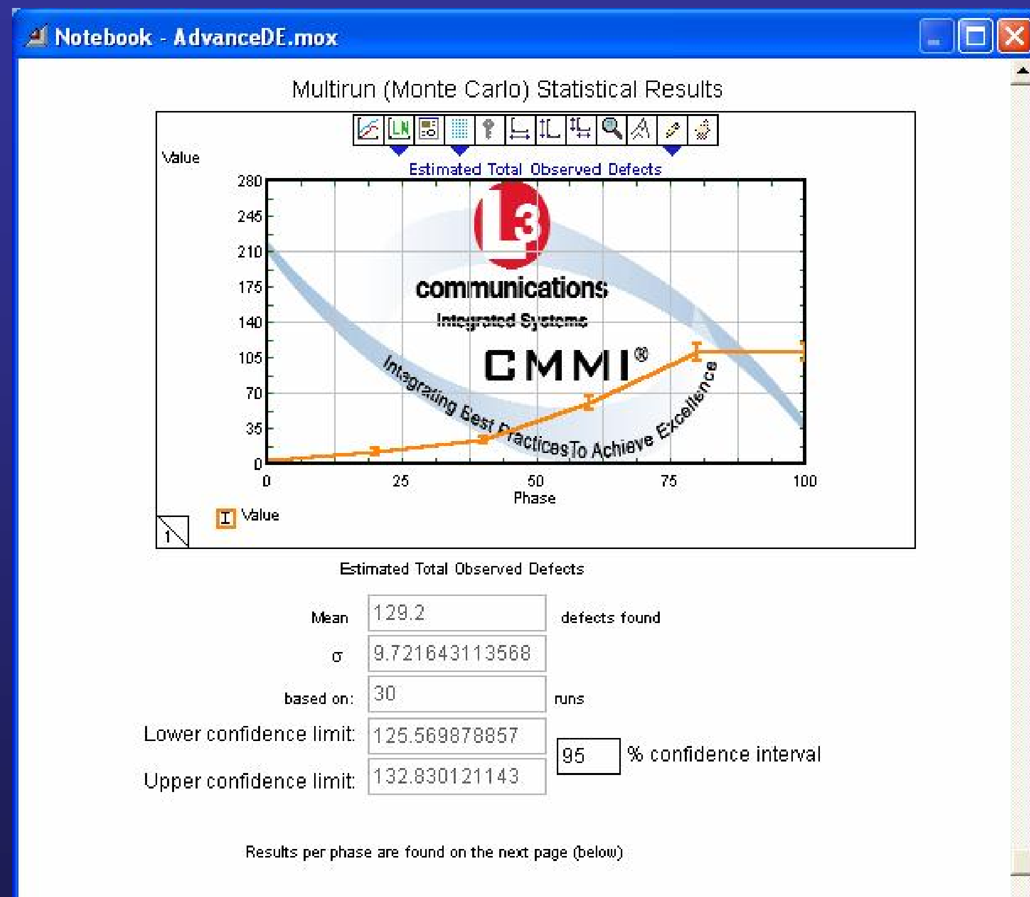
mplement: Results (per-run)

- In-Phase Defects
- Out-of-Phase Defects
- Cumulative per Phase
- Estimated Remaining



mplement: Results (statistical)

Å Mean, Variance, Std. Deviation
  ï Per phase
  ï Total
Å Lower & Upper Confidence limits
  ï Per phase
  ï Total



Outline

Å Basis and Previous Work

Å Implementation

ÅCalibration

Å Deployment

Å Conclusions and Future Work

Å References

Cal: Mapping the Problem

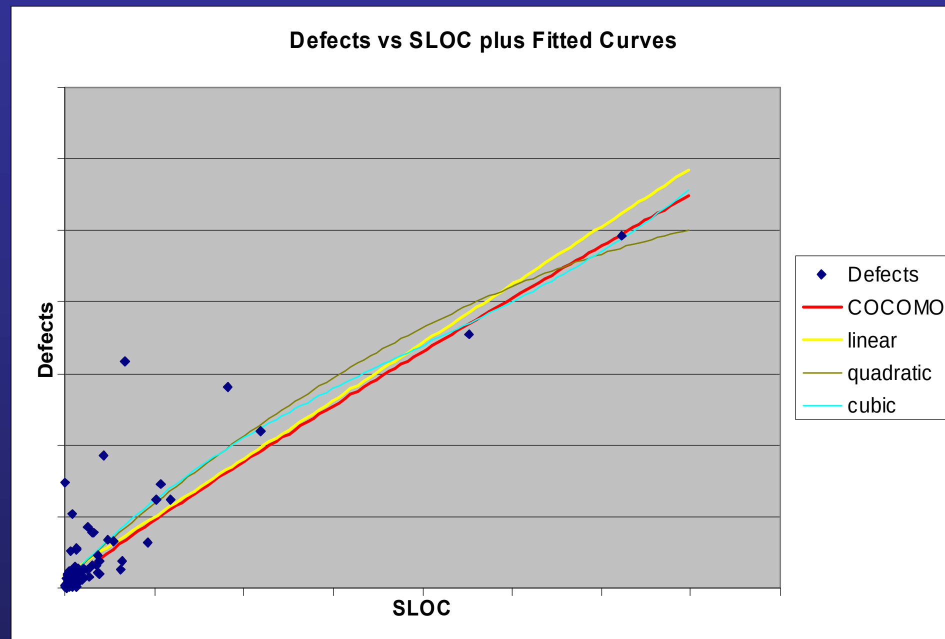
$$F_n[i] = F_{nom}[i]\prod_{j=0}^{k} M_{jn}[i] \qquad n = \{C, I, D\}$$

- Determine the values of $M_{jn}[i]$, $F_{nom}[i]$
- From 53 User Interface inputs, 10 best were selected for initial calibration
- 10 inputs mapped into 21 drive factors
- Multiple settings per input factored into 55 values requiring calibration

Cal: Fitting the Curve

Å Curve fitting was used to determine the *f*(P) (F_{nom}[i]) for the Defect Creation function

Å Several different curve functions were evaluated



COCOMO-type power equation was the best fit
$E = a(S)^b$

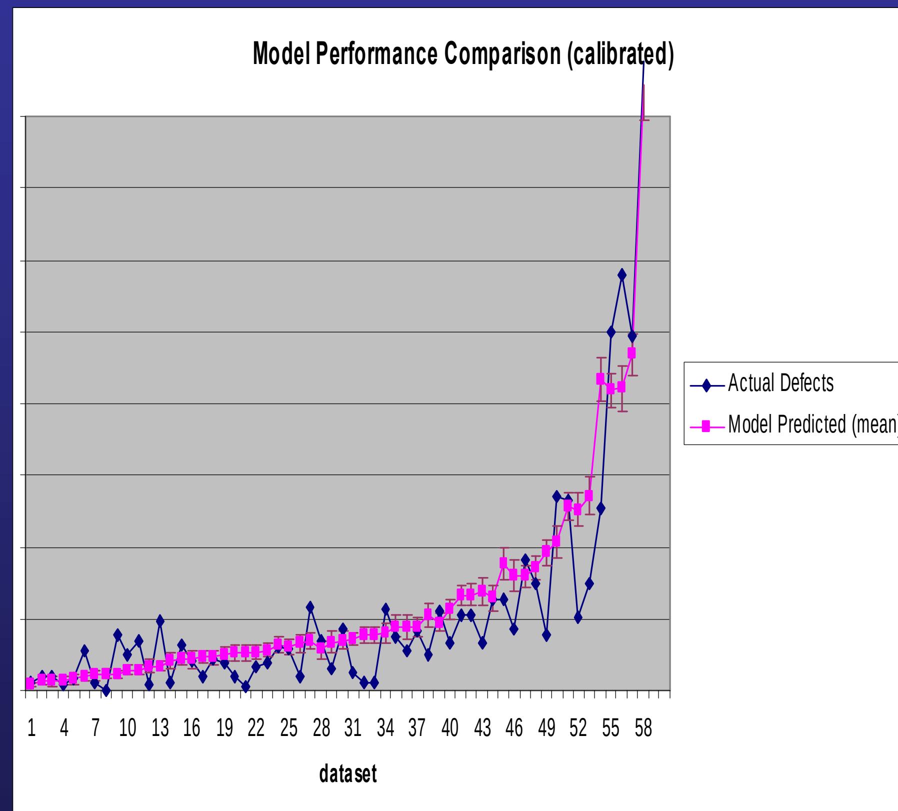
Cal: Running the Calibration

Å Linear regression and iteration were used to optimize $M_{jn}[i]$

Å 55 values calibrated to 72 (later, 66) historical datasets

Å Approximately 500,000 model runs required

Å Final correlation 0.984, improvement of .109



Outline

- Å Basis and Previous Work
- Å Implementation
- Å Calibration
- **ÅDeployment**
- Å Conclusions and Future Work
- Å References

Deployment: Pilot

Å Pilot deployment to SW Engineering Organization

Å Feedback positive

- ï minor changes to UI
- ï Identified and removed 11 ōutliersò and recalibrated

Deployment: Application

- SW Engineering runs model
- Initial recalibration each year with new data
- Organization use:
  - Evaluate organization process performance
  - Assess quantitative return-on-investment for potential process improvement impact
  - Establish organizational objectives

Deployment: Application

- Project use:
  - Run to establish realistic and achievable project objectives
    - Establishes quantitative basis for negotiations
  - Run to estimate or predict the project's performance of selected subprocesses
    - Ensure project's success by predicting future outcomes based on current performance
    - Establish corrective actions today to alter the future course of the project
    - Proactive process performance risk identification
  - Run to assess progress and evaluate corrective action
  - Compare predicted vs. actual
  - Update calibration only if project rebaselines

Outline

- Basis and Previous Work
- Implementation
- Calibration
- Deployment
- **Conclusions and Future Work**
- References

Conclusions

- Significant Effort to Develop
  - But, completely fitted to our process
- Development Effort helped refine metric collection
- Development helped organizational buy-in

Future Work

- SW Model:
  - Calibrate additional UI attributes
  - Calibrate internal distribution spreads
  - Add defect type categories
- Others:
  - Family of models to include other Engineering disciplines: Aero, Systems, HW
- Integrated Product Model

Technique is adaptable to all disciplines

References

- Rout, T.P., and Abel, D.E., (1993) *A Model for Defect Insertion and Detection in Software Development*, Australian Computer Science Conference, Brisbane, Jan. 1993.
- Chulani, S., (1999) *Constructive Quality Modeling for Defect Density Prediction: COQUALMO*, IBM Research, Center for Software Engineering
- Devnani-Chulani S., (1997), *Modeling Software Defect Introduction*, USC – Center for Software Engineering
- Chrissis, M.B., Konrad, M., and Shrum, S. (2007), *CMMI – Guidelines for Process Integration and Product Improvement, 2nd Edition*, Addison-Wesley, Upper Saddle River, NJ. ISBN 0-321-27967-0 pp.-342
- Fenton, N., Neil, M. (1999), *A Critique of Software Defect Prediction*, IEEE Trans. On Software Engineering Vol 25 No. 5, Sept/Oct. 1999 pp. 675
- Li, P.L., Shaw, M., Hebsleb, J., Ray, B., Santhanam, P. (2004), *Empirical Evaluation of Defect Projection Models for Widely-deployed Production Software Systems*, ACM SIGSOFT 12th Int'l Symposium, Nov. 2004, pp. 263-272
- Hartman, P. (2002), *Utility of Popular Software Defect Models*, 2002 Proceedings Annual Reliability and Maintainability Symposium, pp.306-311
- Boehm, B., (1981), *Software Engineering Economics*, Prentice-Hall
- Jones, C., Measuring *Programming Quality and Productivity*, IBM Systems Journal, 17, 1, 1978, p. 39-63.
- Hu, Q., (1997), *Evaluating Alternative Software Production Functions*, IEEE Trans On Software Engineering Vol. 23 No. 6, June 1997 pp. 380

Contact

- **Stan Martin, P.E.**
  903.408.8756
  Stan.Martin@L-3com.com
- **Dawn Littrell**
  903.457.6011
  Dawn.H.Littrell@L-3com.com

L-3 Communications/Integrated Systems
P.O. Box 6056 CBN-108
Greenville, TX 75403-6056