U.S. AIR FORCE

DCARC Data Analysis

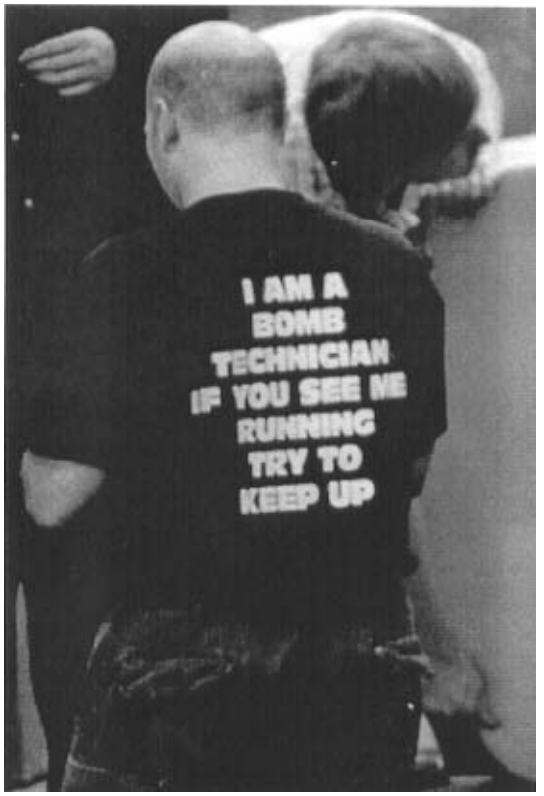
Les Dupaix
Software Technology Support Center
Hill AFB, UT

Data analysis summary

- Background
- Four data types anticipated in the DCARC 2630 software data
- Summary of the DCARC 2630-3 normalized data points
- Autocode problem
- Conclusions

Simplified DCARC reports

- **Form 2630-2**
  - Initial estimate (based on scope)
    - Size
    - Effort (Cost)
    - Schedule (Milestone)
  - Interim estimates
  - Needed for growth metrics
- **Form 2630-3**
  - Completion (actuals) report
    - Size (based on scope)
    - Effort (Cost)
    - Schedule (Milestone)



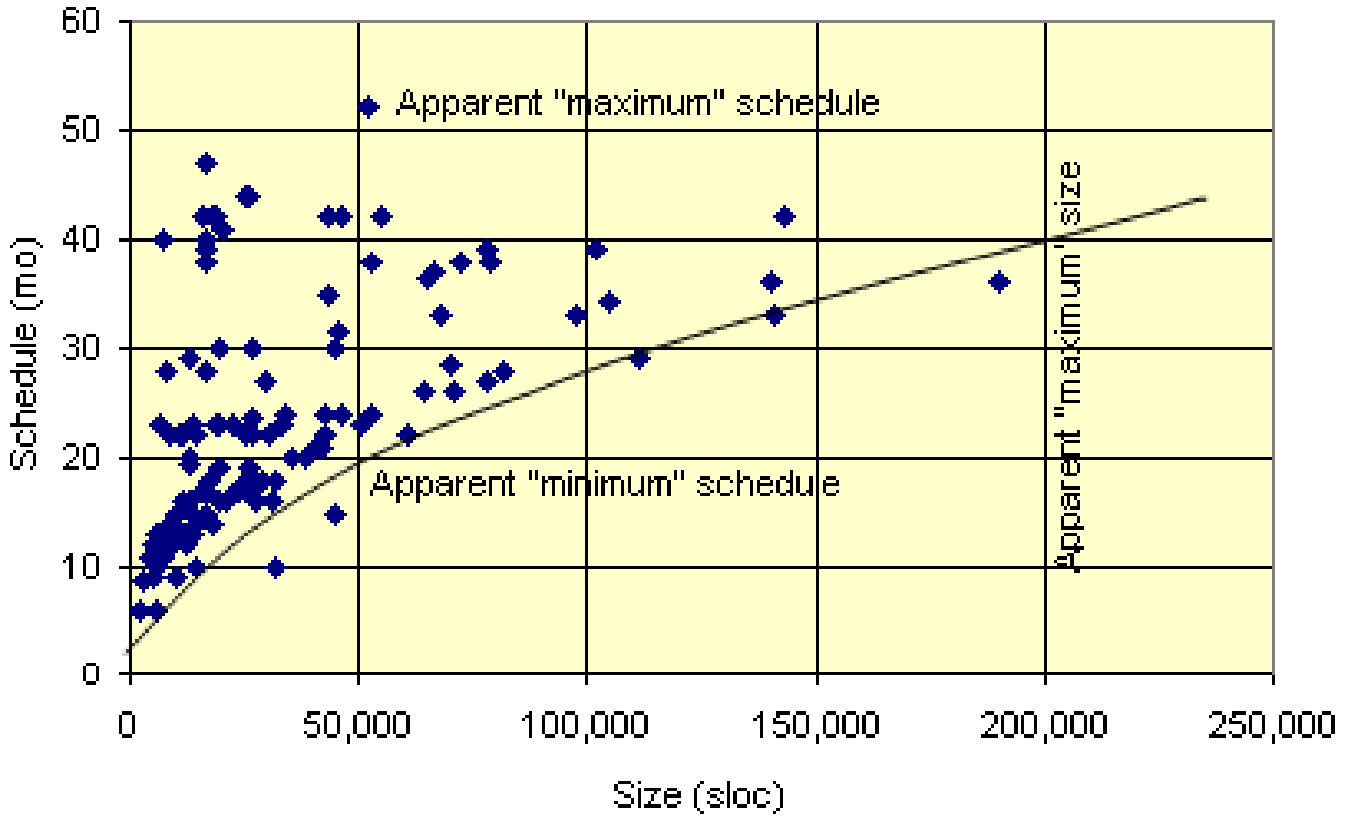
Software project data types

- Component (CSCI)
  - Compatible with 2630 data requirements
  - Validation tests available
- System of systems (SoS)
  - Lack of historical data
  - Data collection strategy needs refinement (What data should be collected?)
- Auto-generated code
  - Lack of historical data
  - No foundation for data collection strategy
- Just plain bad or incomplete

Effective size

- Reflects work required to produce product
  - New and modified source code
  - Reused source code
  - COTS
- SLOC is produced by human effort
- Is approximated by code counting tools
  - Cannot discern code types
  - Includes *dead* code
- Is not related to code generated by
  - Compilers
  - Code generating tools (auto-generated)
  - Is not SLOC as we use it

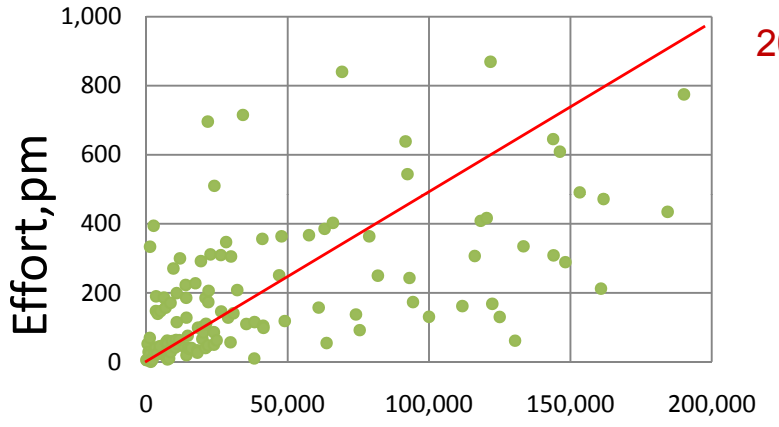
Historic project data:
Minimum schedule – Maximum size



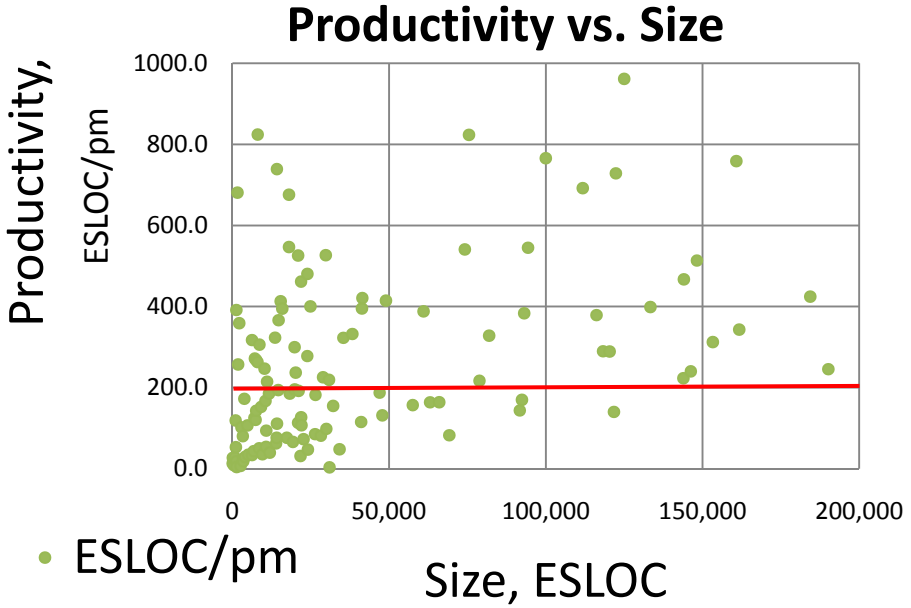
Source: Long, L. G. et al, Aerospace Corp Report, 2004

DCARC Size, Effort, and Productivity

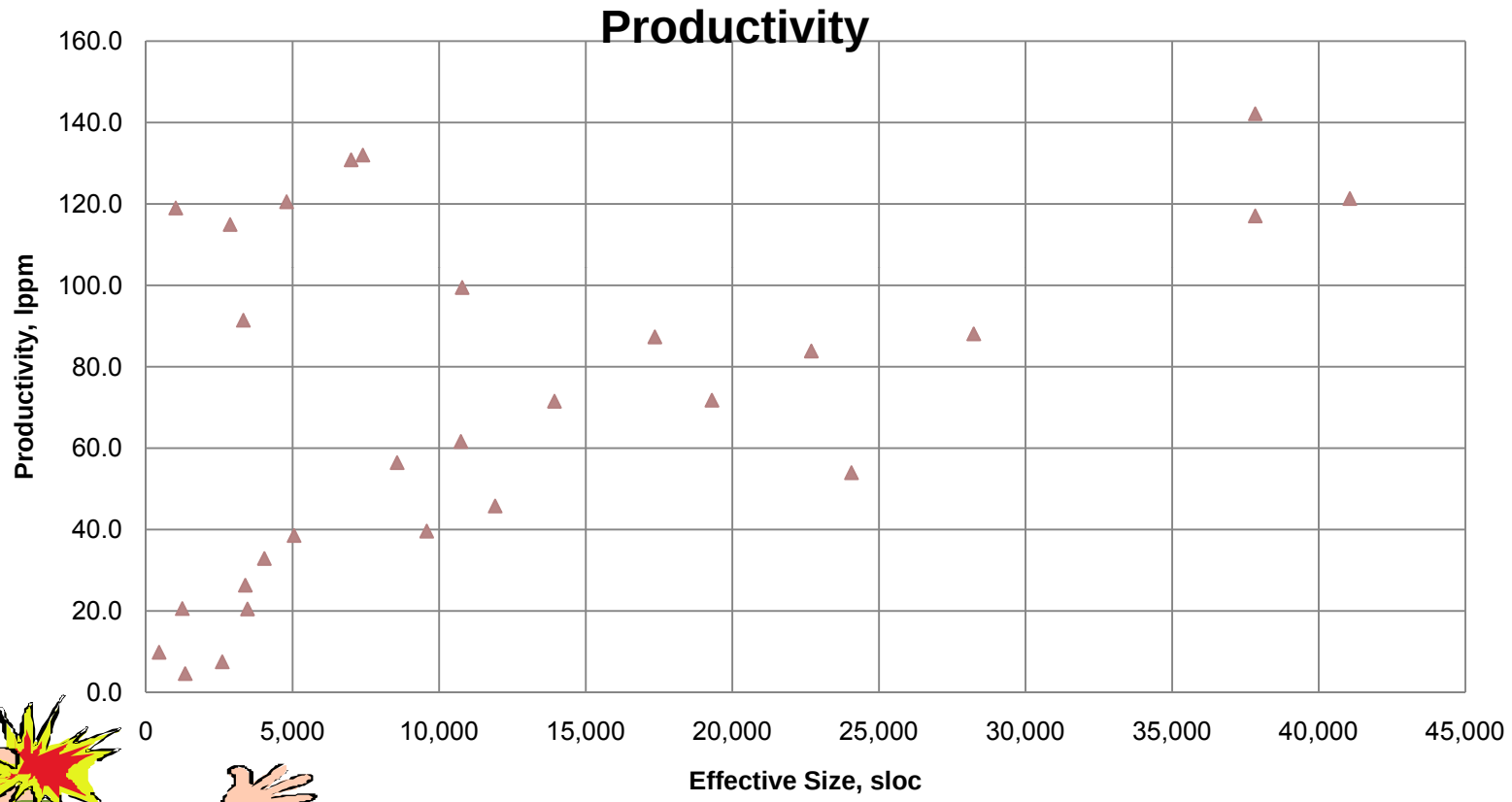
Development Effort vs. Size

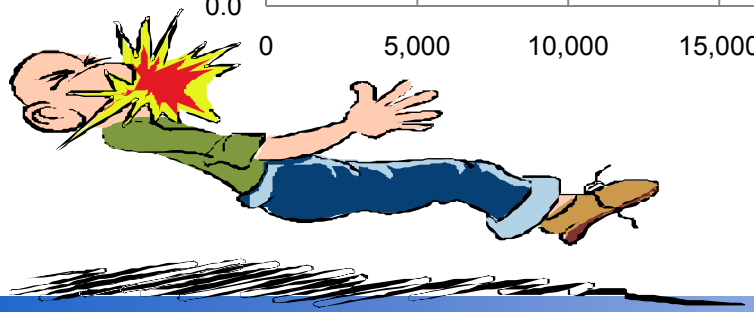


Productivity vs. Size



Small Program Productivity

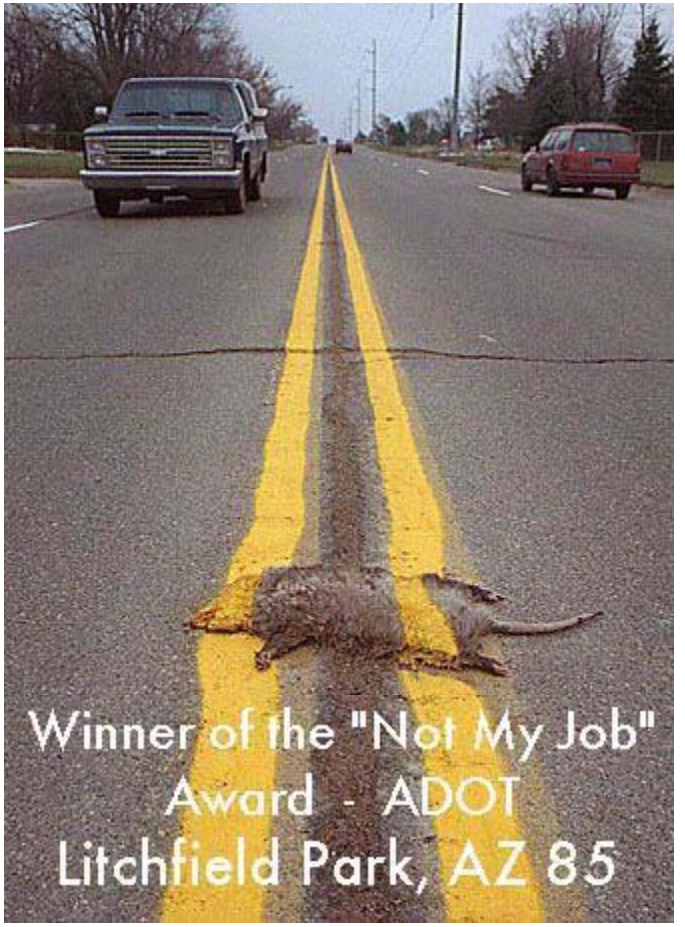




Normalized data validation tests
(Minimum schedule -- CSCI level)

- Based on historical data
- "Paul Masson Rule"

  We will deliver no software before its time
- Normalized schedule (SRR through FQT)
- Determined by
  - Effective size
  - Complexity
  - Application type
  - Developer capability



Data normalization guide

- Size normalization (ESLOC)
  - Based on historic definition (NAVAIR NEMO)
  - $ESLOC = S_{new} + 0.5S_{\text{modified}} + 0.05S_{reused}$
- Development effort normalization (Ed)
  - All software effort expended between SRR and FQT
  - Includes management effort
- Schedule normalization (Td)
  - Elapsed months between SRR and FQT
- Productivity (ESLOC/pm)
  - $PR = ESLOC / E_d$ (sloc/pm)

Minimum schedule calculation

- "Paul Masson" formula
  - Historically validated
  - Based on Jensen Model minimum development time projection
  - Implemented in Sage and SEER-SEM
  - $T_{\min} = \left[C_{te}^{0.4} D^{0.2}\right]^{-1} S_e^{0.4}$ months

- $T_{\min} = 0.23 S_e^{0.4}$ months
  - Assumptions
    - High effective technology constant (C_{te} = 5000)
    - Lowest complexity rating (D=15)

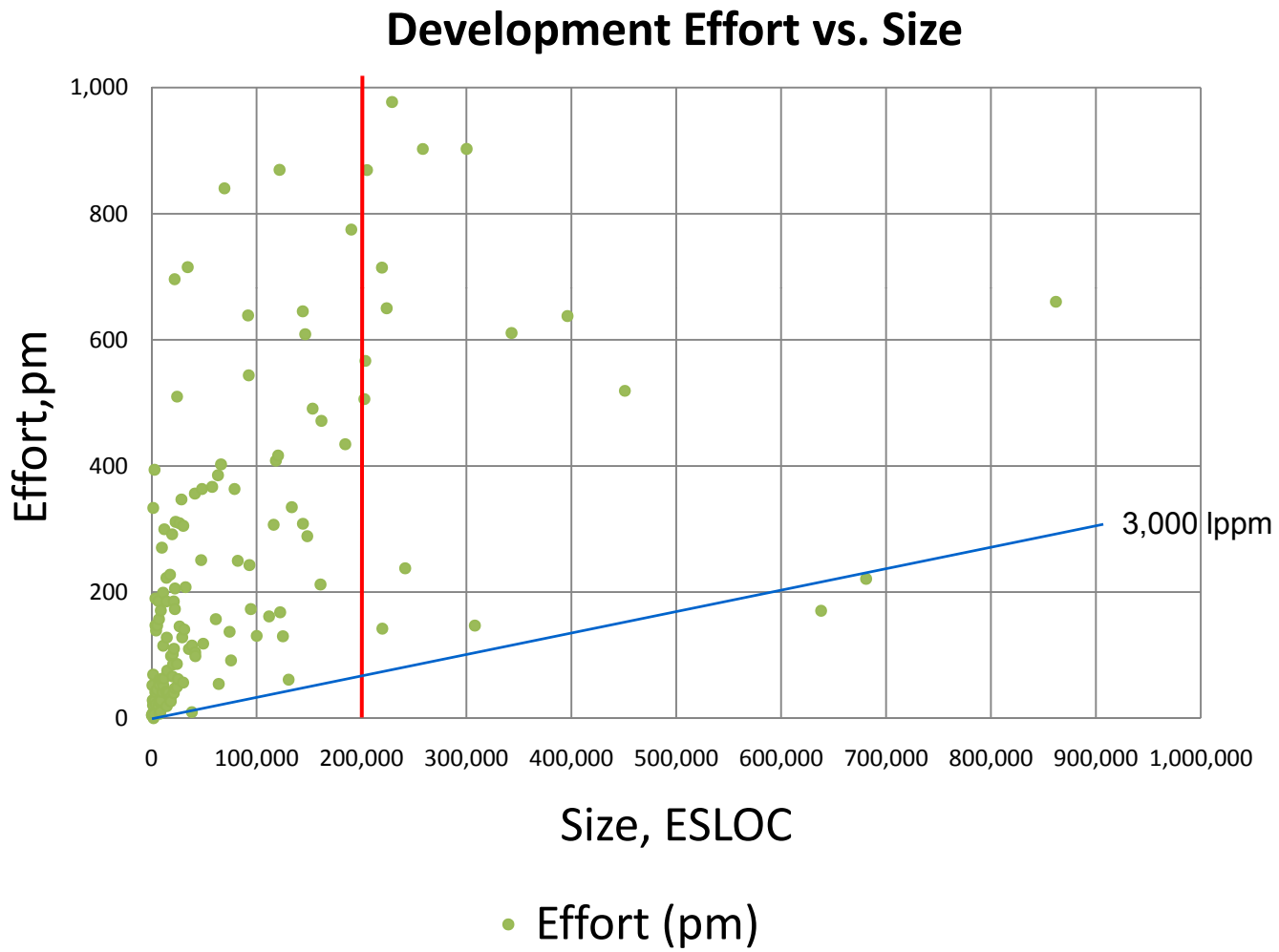
Normalized data validation tests (CSCI level)

- Technology constant (or Productivity Index)
  - Calculated from normalized data
  - Constant value determined by
    - Application type
    - Developer capability
  - Practical upper bound of approximately
    - $C_{te} \approx 7000$ (Sage, SEER-SEM)
    - PI ≈ 11 (SLIM)

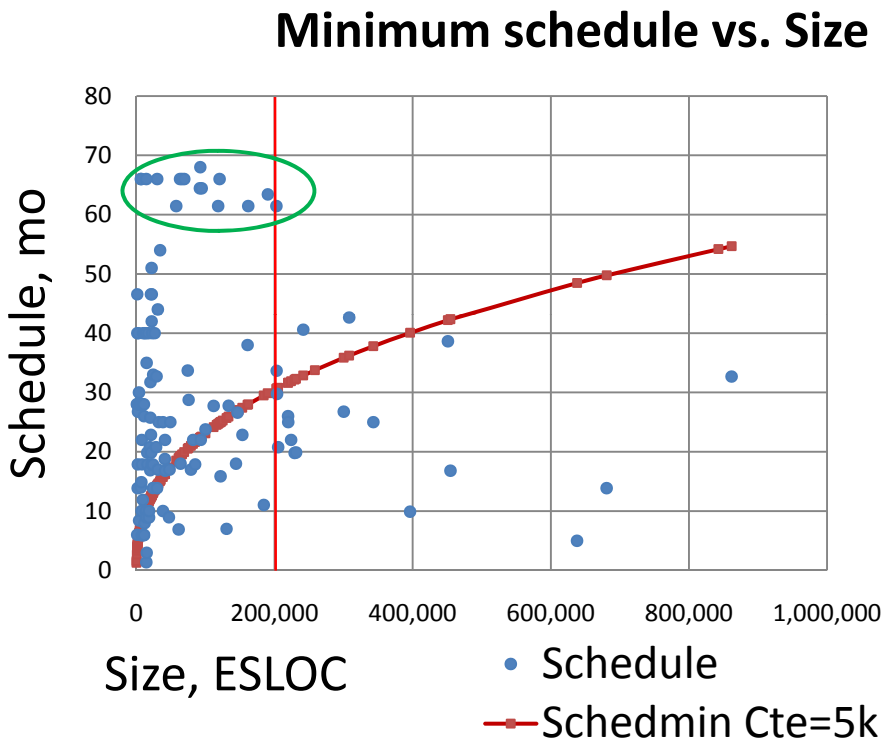
- Effective size
  - Historic CSCI upper limit ≈ 200,000 ESLOC

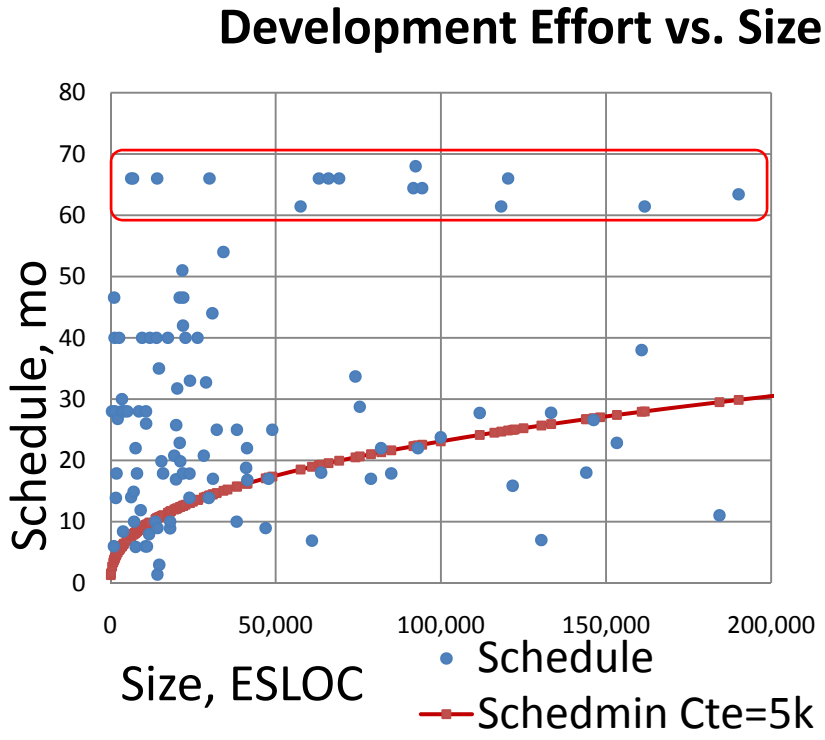
    (Note: Approximate 5 year development)

Normalized DCARC effort vs size

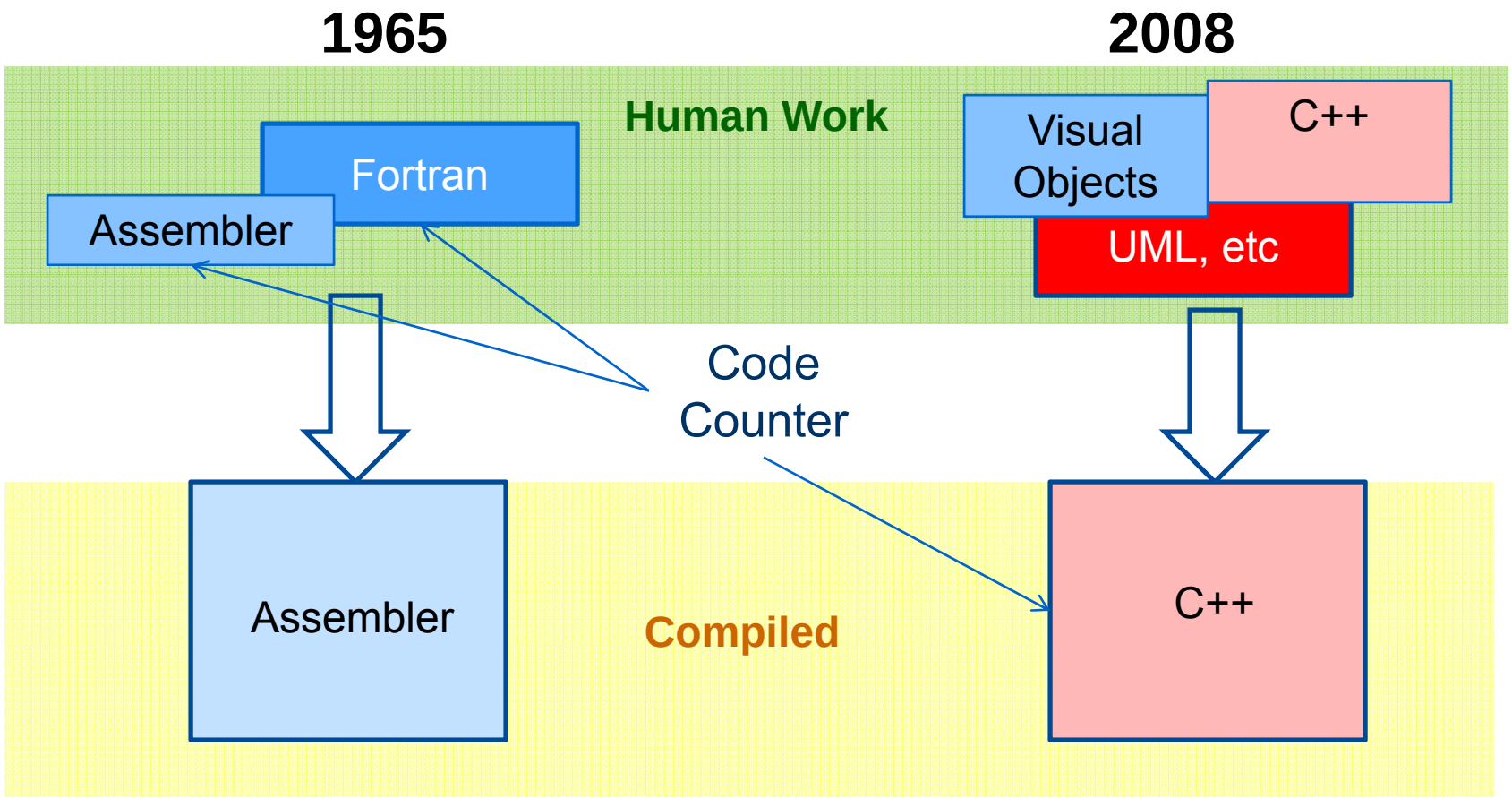


DCARC Development schedule vs size



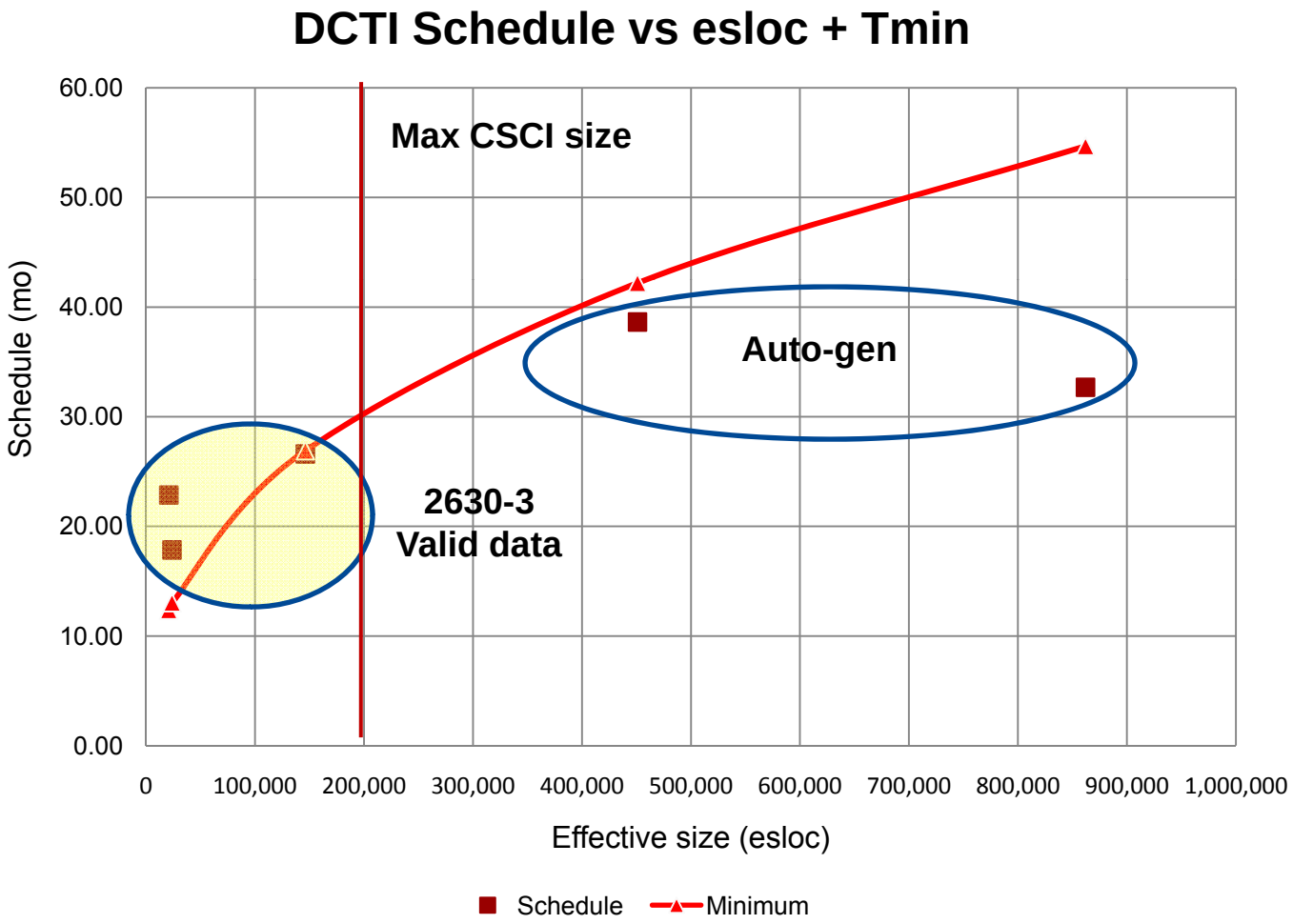


Auto-Code from History



Count should reflect "human" work performed

MPS DCARC 2630-3 data



Conclusions

- 2630 data sheets provide data resources that do support normalization
- Data suppliers are a separate problem
  - Interpretation / understanding / culture are inconsistent
- Growth can only be calculated where -2 and -3 data points are related.
  - Change in scope invalidates growth projections
- Auto-generated code distorts effective size results
  - Current practices do not support effort to measure
- Data inadequate for model calibration or development
  - Environment not included in 2630 data
  - Size information not consistent
  - Fuzzy link between data supplier and DCARC database

Experience

We learn from experience that we don't learn from experience