56-61
12688
P- 15

N94- 36490

1994031983

Assessing Efficiency of Software Production for NASA-SEL Data

Anneliese von Mayrhauser Armin Roeseler

Computer Science Department
Colorado State University
Fort Collins, CO 80523

Abstract

This paper uses production models to identify and quantify efficient allocation of resources and key drivers of software productivity for project data in the NASA-SEL database. While to analysis allows identification of efficient projects, many of the metrics that could have provided a more detailed analysis are not at a level of measurement to allow production model analysis. Production models must be used with proper parameterization to be successful. This may mean a new look at which metrics are helpful for efficiency assessment.

1 Introduction

Many organizations collect a plethora of metrics to help them analyze efficiency of software development and maintenance. Just how helpful are they? We used production models and associated metrics to assessment efficiency of NASA-SEL projects.

While production models have been used in Operations Research for quite a while, their use in the computing field has been limited [1], [10], [13]. One reason for this is that successful development of such models for the software development and maintenance process requires appropriate parameterization (i. e. metrics that are able to help with root cause analysis for process inefficiencies). We can consider software development and maintenance as a production process and model it accordingly. Inputs to the production model are various indicators of resources (effort, tools, capital, expertise, etc.). Outputs reflect the characteristics of software produced (size, quality, etc.). The production model then identifies which development activities were efficient and which factors are related how much to inefficiencies found. This targets specific production activities for improvement. At this point we need to look at more detailed metrics to identify further cause and possible improvement actions. Except for [10, 9] all other applications in the computing field have not provided metrics with the production model that make process improvement possible. Further, [1] only models maintenance requests. To make production models useful for quantitative assessment (and improvement) we must provide a hierarchy of metrics for further analysis of production model results. Both metrics and production model are then bound into a process improvement program.

Section 2 gives a short synopsis of how production models work and how they analyze parameters. A more complete description can be found in [13]. Section 3 reports on the efficiency analysis of NASA-SEL project data. Section 4 gives recommendations for using a production model for efficiency assessment.

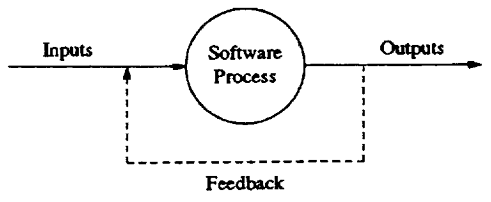


Figure 1: A Simple Causal Model of the Software Process

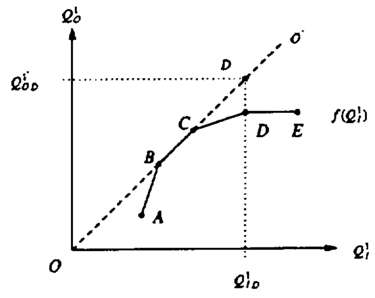


Figure 2: Production Function and Efficiency Frontier

2 Production Models

Production models are causal models that integrate technical and economical analysis perspectives to assess the efficiency of resources used to achieve software development goals. The motivating idea of the production system model is the notion that the software development process transforms resources (e.g., Programmer time, CPU time) into software products. The production function f relates the inputs to the outputs and describes the resource transformation process. We say a software development process is optimized if maximal levels of outputs are attained, given a set of production input quantities.

Figure 1 depicts a production system model that describes how (possibly multiple) input factors (resources) are transformed into (possibly multiple) output factors (e.g., deliverables, quality aspects) using a software development process. Feedback in the production system model is provided through managerial decision making (e.g., deciding on process, project plan and resources).

Because of the complex interactions of software development components, the analytic specification of the production function f is rarely feasible. In the absence of quantitative means to determine the interactions and causalities of components in the production process *directly*, we take an empirical approach to identify optimal production conditions based on historical data on the software development process.

Via linear programming techniques, a convex set of production component data is constructed, and a piecewise linear description of the efficient production frontier is obtained. The efficient production frontier consists of observations that maximize software development goals, given resources consumed, and sets the standard against which other projects or development periods are evaluated. In the process, input efficiencies (slack values), and desired output targets are obtained. Figure 3 shows a production function (A, B, C, D, E) and an efficient production frontier (OO') for a model with one input and one output.

Knowledge of a project's relative (in)efficiency, amount of excess input, and desired output goals can then be used to

- evaluate the efficiency of a SW process,
- decide on strategies to improve efficiency, and
- develop improved SW processes.

3 Production Model Analysis of Project Data in the NASA-SEL Database

We are evaluating project data in the NASA-SEL database with regards to the following:

- does the production model identify efficient and inefficient periods of production?
- are the metrics pinpointing the proper cause?

3.1 Production Model Analysis

We selected 49 projects for analysis. The selection criteria was completeness of project data recorded. We wanted to start our analysis with a rich set of project descriptors. The following projects qualified:
2,6,8,10,19,26,34,35,36,37,38,39,40,46,47,48,49,50,51,52,53,54,55,56, 65,68,70,73,74,80,81,90, 101,102,103,104,105,106,108,110,114,115,116, 117,126,131,132,134,135

Production model analysis must identify inputs and outputs to the production process. These must be at a ratio level of measurement. Rightaway, we face a severe restriction in possible inputs and outputs to the model, since many of the data items are really ranks (e. g. Stability of Requirements). Pretending such data is ratio level is inappropriate, much as we would like to include such key productivity drivers. We decided on a two phase analysis, the first phase uses the (small) set of production inputs and outputs that are at the proper level of measurement. The second phase analyzes rank data, how they appear to influence efficent and inefficient projects.

Input Factors
P132 Total technical and management hours expended on project
P135 CPU hours used

Output Factors
P139 Number of changes made to system components
P141 Total SLOC for all components in the system

The production model clearly identified efficient and inefficient projects. The efficient projects were: 53,54,55,74,110,134. We also included Project 48 with an Efficiency Score of 0.98. The production model clearly identified the periods of inefficiency and the magnitude of inefficient resource usage.

3.2 Metrics Analysis

Next lower level analysis uses the remaining metrics to identify:

- major factors that impact overall project efficiency

- factors that pertain to efficient or inefficient projects only
- Identify factors that most sharply divide efficient from inefficient projects

A properly defined set of metrics is indispensable for successful use of the production model. We found the following results:

1. Factors that correlate with both efficient and inefficient projects.
    - Pos. Correlation (i.e., 'more' is beneficial)
        - P90 Stability of Requirements
        - P100 Stability of Management Team

      While this confirms other analyses (e. g. those underlying the COCOMO model), this correlation by itself does not tell us whether a lack of stability in requirements and management team caused the inefficiency. Had this data been collected at a ratio level of measurement, we could have identified cause.
    - Neg. Correlation (i.e., 'less' is beneficial)
        - P93 Rigor of Requirements Review
        - P115 System Response Time

      Again, this data is ordinal, and gives rise to possible interpretations. One might venture to say that this result indicates that rigor can go overboard and thus causes inefficiencies, but this might also be due to inconsistent data collection, specifically lack of inter-rater reliability for P93. This would point to a need for metrics validation before collecting them on a large scale.

2. Factors that correlate with inefficient projects, but do not correlate with efficient projects. That is, efficient projects are immune to the factors listed below, while inefficient projects are influenced by them.
    - Pos. Correlation (i.e., 'more' is beneficial)
        - P95 Development Team Application Experience

      This is a very interesting result as it appears to say that "if you're not as efficient as you could be, the experts bail you out".
    - Neg. Correlation (i.e., 'less' is beneficial)
        - P88 Problem Complexity
        - P105 Discipline in Requirements Methodology
        - P112 Access to Development System
        - P113 Ratio of Developers to Terminals

      Again, much of this is ordinal data, some might be dependent, so if anything we should investigate this further. What is quite interesting is that the efficient projects don't seem to be affected by this.

3. Factors that most sharply discriminated efficient and inefficient projects (statistics probably not significant)
    - Pos. Correlation (i.e., 'more' is beneficial)
        - P106 Discipline in Design Methodology

    - P119 Quality of SW
  - Neg. Correlation (i.e., 'less' is beneficial)
    - P91 Quality of Requirements

4. All Other Factors:
  - No significant correlations to efficient and inefficient projects detected.
  - Incomplete data for factors P104 and P117.

While we could identify efficent and inefficient production outcomes for these 50 projects, many of the State-of-the-art metrics in the NASA-SEL database do not have enough power to assess the efficiency of software production in enough detail to suggest improvements. Two problems, subjective ranking and questions about inter-rate reliability are key to the situation. Unfortunately, this is a very common problem.

4 Conclusion

We clearly need methods to assess efficiency of software production. Reliable quantitative methods paired with engineering judgement appear most promising. This paper described production models and how to use them for efficiency assessment. We applied the approach to data from 49 projects in the NASA-SEL database to show its benefits and the needs for better, more relevant metrics to drive any quantitative evaluation.

References

[1] R. Banker, S. Datar, Ch. Kemmerer; "A Model to Evaluate Variables Impacting the productivity of Software Maintenance Projects", *Management Science 37*, 1(Jan. 1991), pp. 1–18.

[2] A. Roeseler; *A Production-Based Approach to Performance Evaluation of Computing Technology*, PhD Thesis, Illinois Institute of Technology, 1991.

[3] A. Roeseler, A. von Mayrhauser, "A Production-Based Approach to Performance Evaluation of Computing Technology", *Journal of Systems and Software*, to appear 1993.

[4] von Mayrhausr, A., Roeseler, A.; "Software Process Assessment and Improvement using Production Models", *Procs. COMPSAC 93*, Nov. 1993, Phoenix, AZ.

Slide 1

Assessing Efficiency of Software Production for NASA–SEL Data

Anneliese von Mayrhauser
Computer Science Department
Colorado State University
Fort Collins, CO 80523
avm@cs.colostate.edu

Armin Roeseler
AT&T Bell Laboratories
Warrenville Road
Naperville, IL
doit@ihlpa.att.com

Slide 2

Outline

1. Production models in software engineering
2. Production model analysis
3. Analyzing NASA–SEL Data
    - Productivity Analysis
    - Metrics Analysis
4. Conclusion

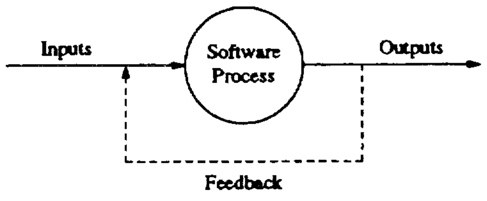
Slide 3

1. Production Models in Software Engineering

- Productivity is multi-faceted
- Analyze software development & maintenance as a microeconomic production process
  - resource transformation (empirically-based)
  - mathematical approach to efficiency measurement
  - incremental process of achieving/maintaining successively higher levels of efficiency

Slide 4

2. Production Model Analysis



$$f(Q_O;Q_I) = f(Q_O^1,\ldots,Q_O^M;Q_I^1,\ldots,Q_I^N) = 0$$

- Q_O^m level of m-th output, $m = 1,\ldots,M$
- Q_I^n level of n-th input, $n = 1,\ldots,N$
- $f : R_+^N \mapsto R_+^M$ production function giving maximal output for given input

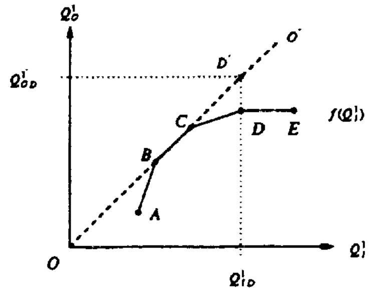
Slide 5

Production Function

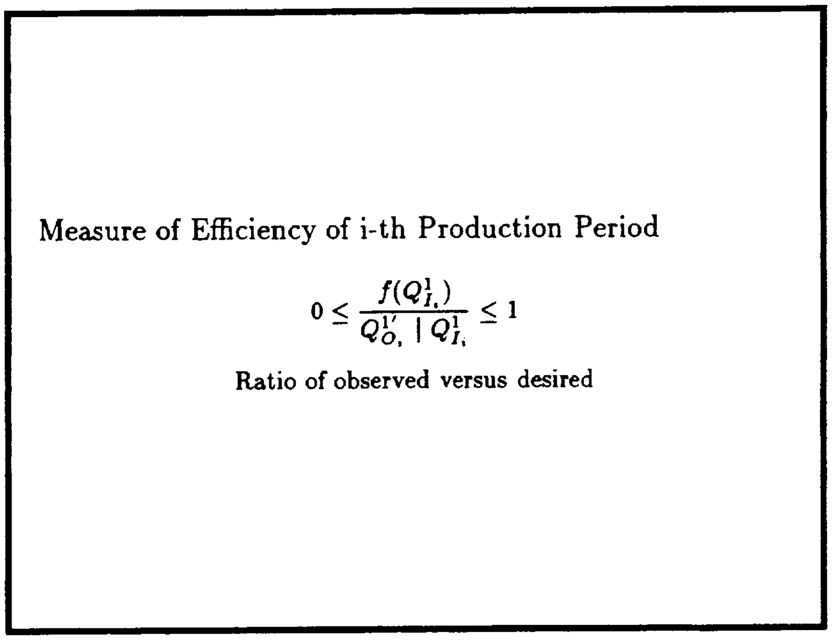
- complex interactions
- analytic specification rarely feasible
- use empirical approach based on historic production observations

Slide 6

Estimation of Production Function



Slide 7

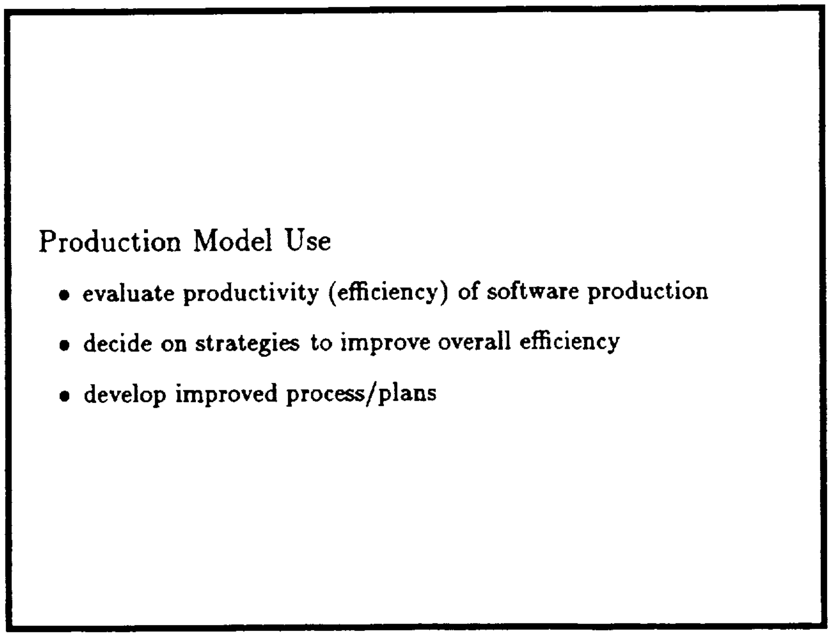


Measure of Efficiency of i-th Production Period

$$0 \leq \frac{f(Q^1_{I_i})}{Q^{1'}_{O_i} \mid Q^1_{I_i}} \leq 1$$

Ratio of observed versus desired

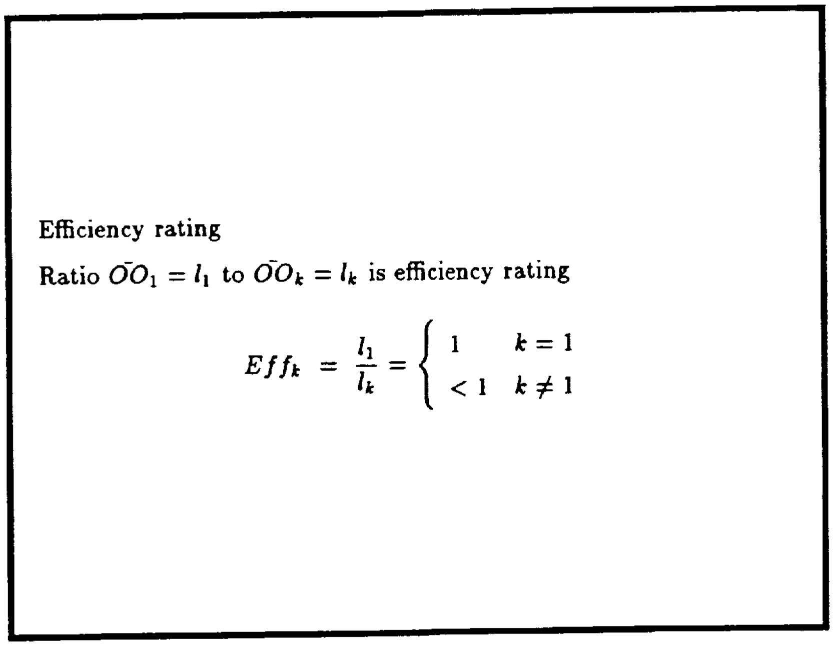
Slide 8



Production Model Use

- evaluate productivity (efficiency) of software production
- decide on strategies to improve overall efficiency
- develop improved process/plans

Slide 9

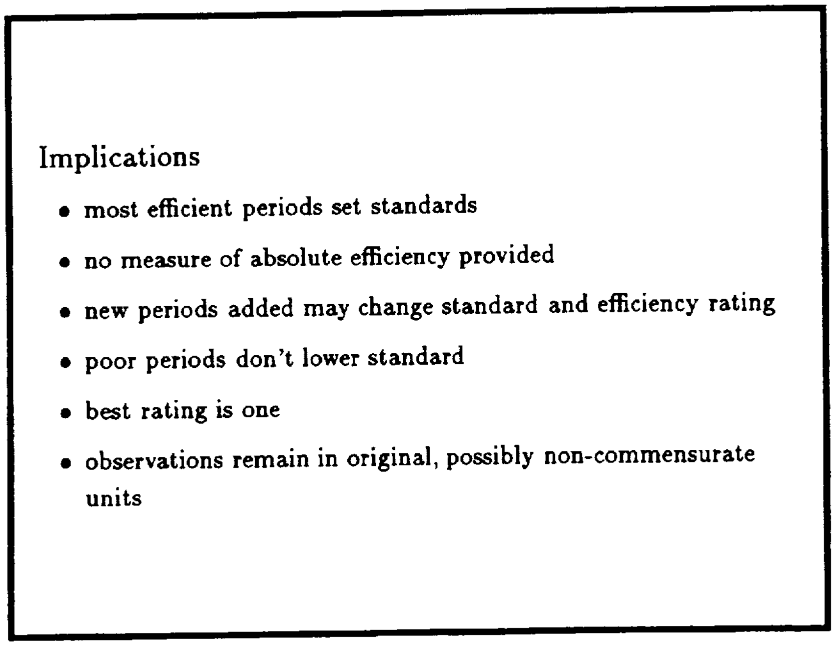


Efficiency rating

Ratio $\bar{OO}_1 = l_1$ to $\bar{OO}_k = l_k$ is efficiency rating

$$Eff_k = \frac{l_1}{l_k} = \begin{cases} 1 & k = 1 \\ < 1 & k \neq 1 \end{cases}$$

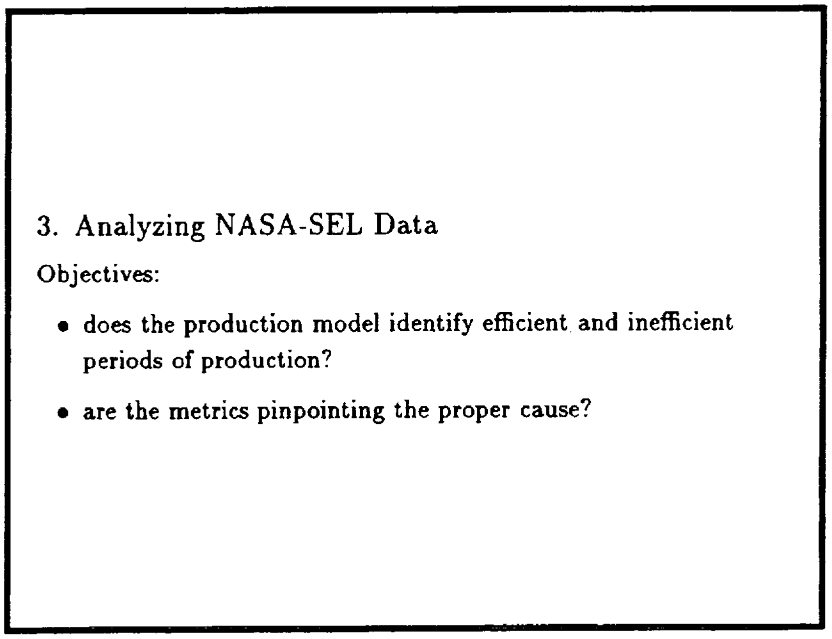
Slide 10



Implications

- most efficient periods set standards
- no measure of absolute efficiency provided
- new periods added may change standard and efficiency rating
- poor periods don't lower standard
- best rating is one
- observations remain in original, possibly non-commensurate units

Slide 11

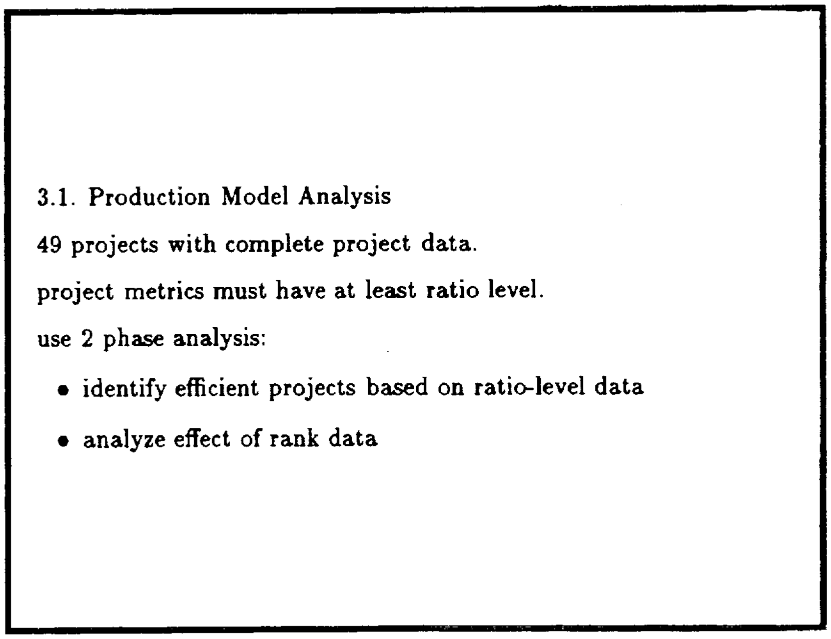


3. Analyzing NASA-SEL Data

Objectives:

- does the production model identify efficient and inefficient periods of production?
- are the metrics pinpointing the proper cause?

Slide 12



3.1. Production Model Analysis

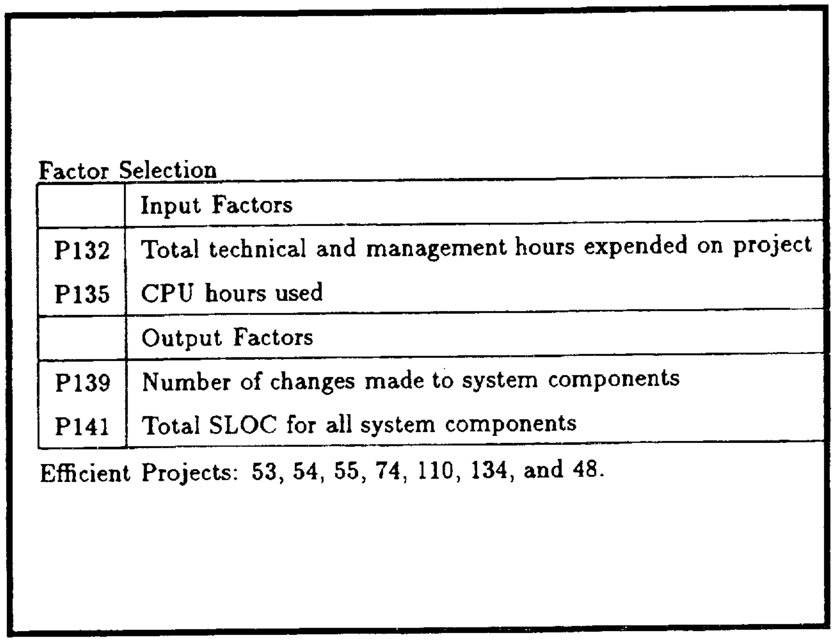
49 projects with complete project data.

project metrics must have at least ratio level.

use 2 phase analysis:

- identify efficient projects based on ratio-level data
- analyze effect of rank data

Slide 13

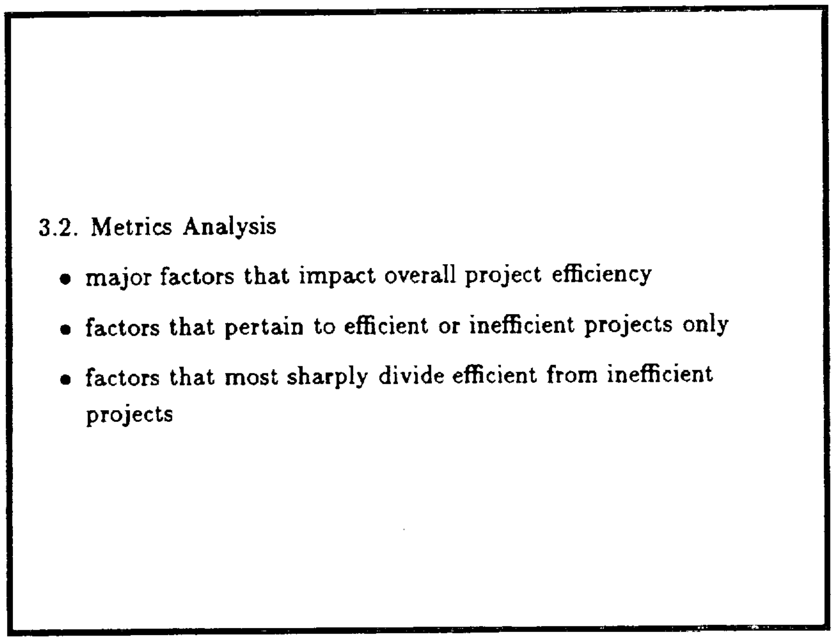


Factor Selection

	Input Factors
P132	Total technical and management hours expended on project
P135	CPU hours used
	Output Factors
P139	Number of changes made to system components
P141	Total SLOC for all system components

Efficient Projects: 53, 54, 55, 74, 110, 134, and 48.

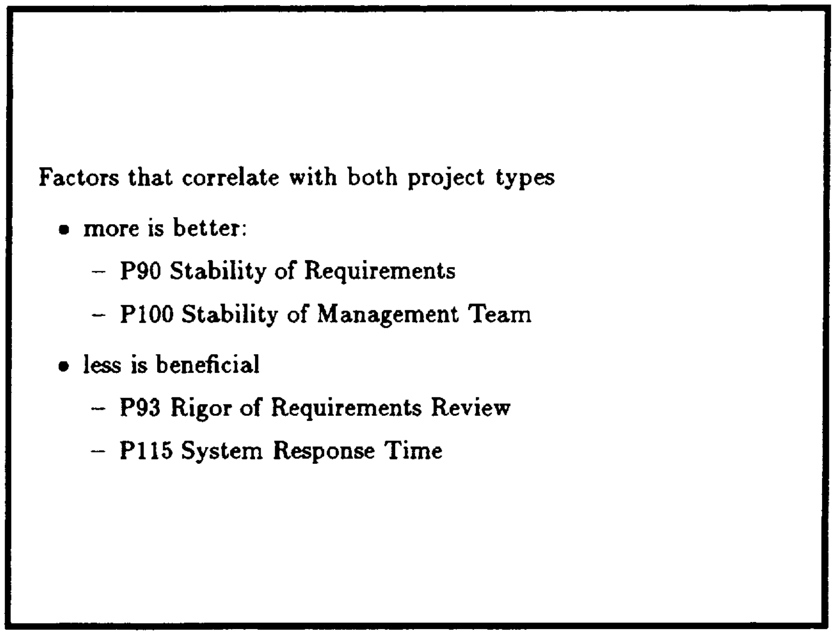
Slide 14



3.2. Metrics Analysis

- major factors that impact overall project efficiency
- factors that pertain to efficient or inefficient projects only
- factors that most sharply divide efficient from inefficient projects

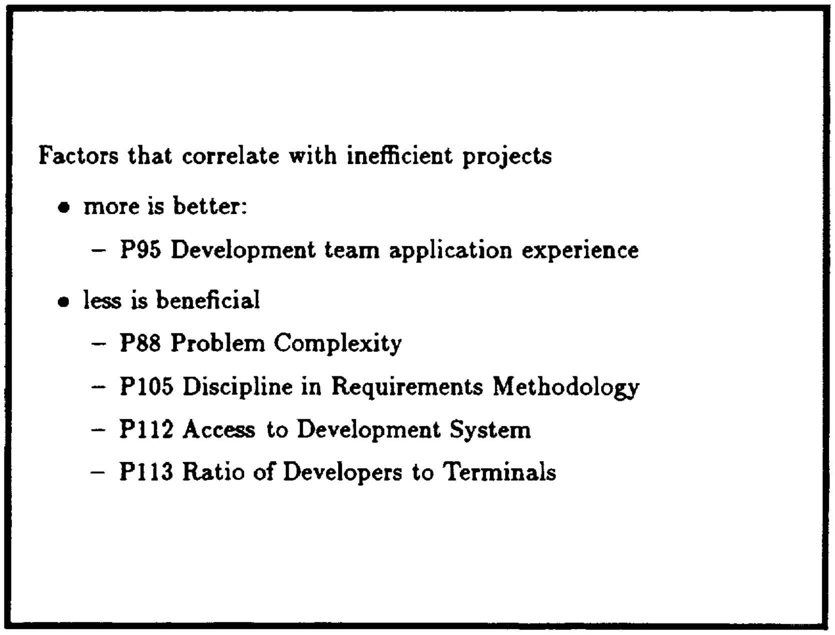
Slide 15



Factors that correlate with both project types

- more is better:
  - P90 Stability of Requirements
  - P100 Stability of Management Team
- less is beneficial
  - P93 Rigor of Requirements Review
  - P115 System Response Time

Slide 16



Factors that correlate with inefficient projects

- more is better:
  - P95 Development team application experience
- less is beneficial
  - P88 Problem Complexity
  - P105 Discipline in Requirements Methodology
  - P112 Access to Development System
  - P113 Ratio of Developers to Terminals

Slide 17

Factors that sharply discriminate efficent vs. inefficient projects

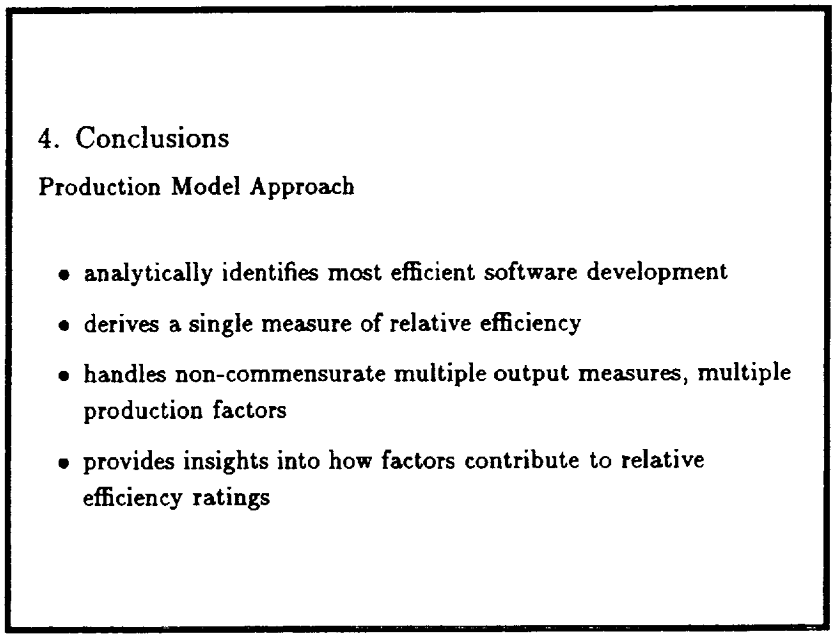
- more is better:
  - P106 Discipline in design Methodology
  - P119 Quality of Software
- less is beneficial
  - P91 Quality of Requirements

Slide 18

Disciplined Metrics Development

- evaluate quality of current metrics
  - validity
  - reliability
- develop hierarchy of production relevant factors to measure
- identify for all metrics: what/why/meaning
- parameterize the software development process
- determine goal oriented selection process
- bind into general metrics program

Slide 19



4. Conclusions

Production Model Approach

- analytically identifies most efficient software development
- derives a single measure of relative efficiency
- handles non-commensurate multiple output measures, multiple production factors
- provides insights into how factors contribute to relative efficiency ratings