

Resource Models and Metrics

Software Resource Models and Measures

We can define a project as a set of tasks that consume resources and produce a product

Thus resources are consumed during a project

What types of resources exist?

Hardware

Software

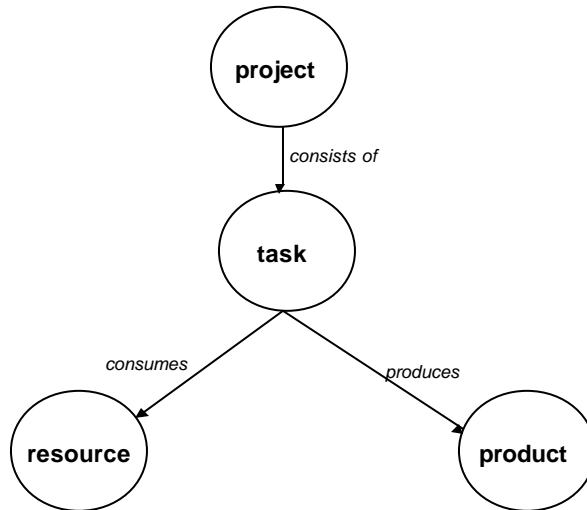
Human

Support (supplies, materials, communications, facility costs, etc.)

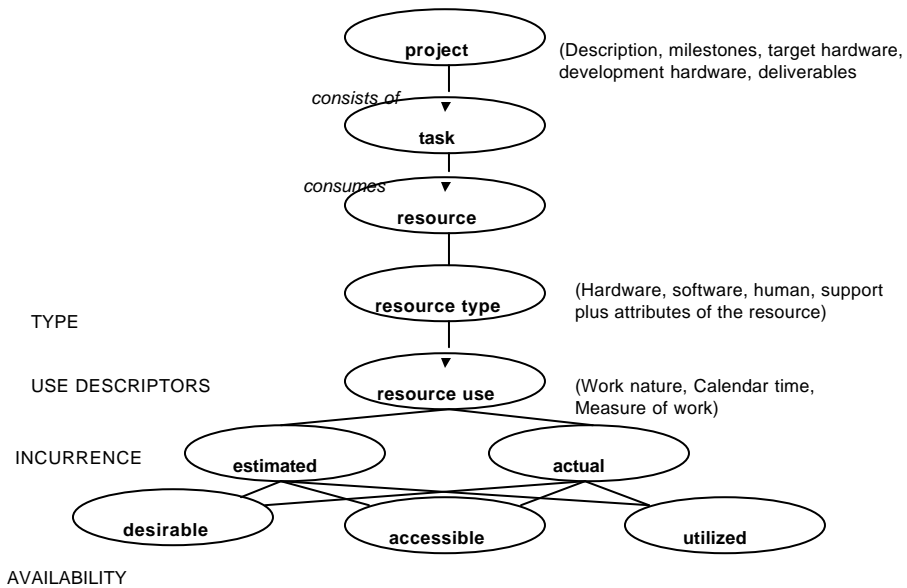
Are the resources estimated or actual?

Are the resources desirable (resources of value), accessible (able to be used) utilized (actually used)?

A Model of a Software Project



A Model of a Software Project



Software Resource Models and Measures

Resource Data

Human Effort data may be measured in staff-hours, weeks, months, years . . .

Calendar time data may be measured in calendar hours, days, weeks, months, date to date

Computer Time data may be measured as calendar time, execution time

They may be associated with various

Processes:

phases: requirements, design, implementation, test,...

activities: reading, design inspections, making changes, meetings,...

Products:

documents: requirements, design, test plan, user's manual,...

program parts: system, module, design document, requirements section,...

Other project characteristics:

calendar time: from date to date

Software Resource Models and Measures Sample Resource Metrics

The data can be aggregated to define various metrics, e.g.,

Total Effort for the project

Design Effort, Design Effort as % of Total

Design Calendar Time from Requirements review to Design review

Staff time to

make a test

run a test and check the result

isolate the fault?

design and implement a fix

retest

Machine time used to run a test suite

This can be based upon actual data or estimated data

Software Resource Models and Measures

Effort by Phase

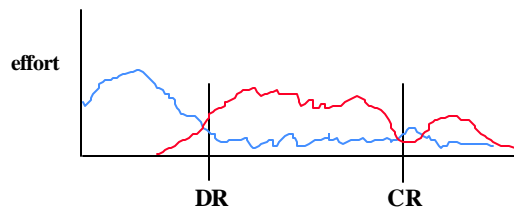
	Analysis and Design	Coding and Auditing	Checkout and Test
Sage	39%	14%	47%
NTDS	30	20	50
Gemini	36	17	47
Saturn V	32	24	44
OS/360	33	17	50
TRW Survey	46	20	34

Boehm/TRW

Software Resource Models and Measures

TRW IBM SEL

			Phase	Activity
Design	40%	35%	20%	21%
Code	20	30	45	28
Checkout/Test	40	25	28	23
Other		10	5	27



Software Resource Models and Measures

Why model/measure resources?

Initial Prediction

Given what we know (or can guess) about a project, what can we predict about effort (cost), staffing, computer use, . . . ?

Description of the Development Pattern

Provides insights into what is going on

How do different parameters change the pattern?

Provides an evaluation of techniques, methodology and engineering

What can we learn about future developments?

Prediction of the Next Phase from the Current Phase

What should happen next?

If it doesn't, why not; is it a sign of trouble, etc.?

Model Validation

Does the model explain our behavior and environment?

Do the factors (model parameters) agree with our environmental factors?

Are they calibrated correctly?

Software Resource Models and Measures

Characteristics of a Good Model

It explains our behavior and the development environment

Parameters are calculable from known data (or easy to guess data) e.g.,

Maximum staffing

Time to delivery

Complexity of software

Lines of code, number of modules, number of I/O formats

Type of software

Amount of old/new software (design, code, specification)

Parameters describe and can be calibrated for our environment

It includes redundancy checks and risk analysis factors

When the model doesn't work

One can gain insight into why and what is different in the environment

Software Resource Models and Measures

What kinds of models can we build?

What is estimated: effort, staffing, cost, computer use, time

Type of Analysis:

- Least square and regression analysis

- Neural networks

- Machine learning approaches, e.g., decision trees

- Multiple Regression Models

 - Single variable vs. multi-variable

 - Adjusted baseline

 - Adjusted table driven

 - Multi-parameter

Static staffing vs. dynamic staffing

Empirical vs. theoretical

Macro vs. micro

Software Resource Models and Measures

Multivariate Modeling Solutions

Least square and logistic regression analysis

- sensitive to outliers

- requires distributional and functional assumptions

- difficult to deal with interactions

- difficult to deal with symbolic data

- models are unstable and difficult to interpret

but these models are based on a solid, well formalized theory

Neural networks

- models are very difficult to interpret

- optimal modeling strategies are still unclear in this area

but do not require explicit functional assumptions

Machine learning approaches, e.g., decision trees

- sometimes lack of solid statistical theory

but the models are easy to interpret for application domain experts

Software Resource Models and Measures

Single Variable Regression Models

Effort equation is based on a single variable, usually a measure of size.

There are several possible variations:

$$\text{Effort} = A * \text{size} + C$$

$$\text{Effort} = A * \text{size}^B$$

$$\text{Effort} = A * \text{size}^B + C$$

where A, B and C are constants determined by regression analysis on historical data.

Effort may be measured in

Staff: hours, weeks, months, years . . .

Size may be measured in

Lines of code, modules, I/O formats . . .

Software Resource Models and Measures

Sizing Methods Approaches

Top-down estimating

Similarities and differences estimating

Ratio estimating

Standards estimating

Bottom-up estimating

Combination of two or more basic methods

Wolverton/TRW

Software Resource Models and Measures

Static Single Variable Example

Goal: Measure rate of production of lines of code by projects, influenced by a number of product conditions and requirements

Data Base: 60 projects
4K to 467K source lines
12 to 11,758 staff months
Variety of task types, languages, ...

Basic Effort Equation Form: $E = A * SIZE^B$

Effort Estimation Equation: $E = 5.2L^{.91}$

where E = effort in staff months
L = lines of code in thousands

Walston & Felix: IBM Federal Systems Division

Software Resource Models and Measures

It is possible to identify the relationship among any pair of variables by plotting the data and calculating a best fit equation

Other Variable Relationship Equations:

$$E = 5.2L^{.91}$$

$$DOC = 49L^{1.01}$$

$$D = 4.1L^{.36}$$

$$D = 2.47E^{.35}$$

$$S = .54E^{.6}$$

where

E = effort in staff months

L = lines of code in thousands

DOC = documentation in pages

D = project duration in calendar months

S = average staff size = E/D

Software Resource Models and Measures

Productivity Index

After calculating the basic relationship between size and effort, how does one identify the effect of other variables?

What are the other variables?

Walston and Felix identified 69 potential influencing variables ranging from context to experience and the nature of the problem to methods used to solve the problem

They correlated all 68 variables with productivity and selected 29 that showed a significantly high correlation with productivity

The goal was then to find a way to measure the influence of these variables on the basic estimated effort value

Software Resource Models and Measures

Productivity Index

The influencing variables were measured on a three point ordinal scale for

For those projects with data on those variables, they were divided into three groups of relatively equal size yielding a group of low, medium and high rating with regard to the variable

Average productivity was calculated for each for the three groups and the difference in productivity between the high and low groups was used as a base for the weighting

They then calculated an effort multiplier, called the **productivity index**, whose goal was to weight the effort estimate based upon the historical "influence" of the variables

Software Resource Models and Measures

Productivity Index

The weights were calculated based upon historical data

$$I = \sum_{i=1}^{29} W_i X_i$$

where

I = Productivity index

W_i = question weight ($1/2 \log_{10}$ (ratio of productivity change for question i))

X_i = question response (+1, 0, or -1), depending on whether the responses indicate increased, nominal or decreased productivity

The productivity index is used to adjust the initial estimator from the baseline equation by explaining deviations from the norm.

Software Resource Models and Measures

Productivity Index

Question or Variable	Mean Productivity (DSL/MM)			Change (DSL/MM)
	<normal	normal	>normal	
Customer interface complexity	500	295	124	376
User participation in the definition of requirements	none 491	some 267	much 205	286
Customer originated program design changes	few 297		many 196	101
Customer experience with the application area of the project	none 313	some 340	much 206	112
Overall personnel experience and qualifications	low 132	average 257	high 410	278

Modeling and Measuring Resources

Productivity Index

Question or Variable	Mean Productivity (DSL/MM)			Change (DSL/MM)
Percentage of program - mers doing development who participated in design of functional specifications	< 25% 153	25-50% 242	> 50% 391	238
Previous experience with operational computer	minimal 146	average 270	extensive 312	166
Previous experience with programming languages	minimal 122	average 225	extensive 385	263
Previous experience with application of similar or greater size and complexity	minimal 146	average 221	extensive 410	264

Modeling and Measuring Resources

Productivity Index

Question or Variable	Mean Productivity (DSL/MM)			Change (DSL/MM)
Ratio of average staff size to duration (people/month)	<0.5 305	0.5-0.9 310	>0.9 173	132
Hardware under con- current development	no 297		yes 177	120
Development computer access, open under special request	0% 226	1-25% 274	>25% 357	131
Development computer access, closed	0-10% 303	11-85% 251	>85% 170	133
Classified security en- vironment for computer and 25% of programs and data	no 289		yes 156	133

Modeling and Measuring Resources Productivity Index

Question or Variable	Mean Productivity (DSL/MM)			Change (DSL/MM)
	0-33%	34-66%	66%	
Structured programming	169	-	301	132
Design and code inspections	220	300	339	119
Top down development	196	237	321	125
Chief programmer team usage	219	-	408	189
Overall complexity of code developed	<average 314		>average 185	129
Complexity of application processing	<average 349	average 345	>average 168	181

Modeling and Measuring Resources Productivity Index

Question or Variable	Mean Productivity (DSL/MM)			Change (DSL/MM)
	<average	average	>average	
Complexity of program flow	289	299	209	80
Overall constraints on program design	minimal 293	average 286	severe 166	107
Program design constraints on main storage	minimal 391	average 277	severe 193	193
Program design constraints on timing	minimal 303	average 317	severe 171	132
Code for real-time or interactive operation, or executing under severe timing constraint	<10% 279	10-40% 337	>40% 203	76

Modeling and Measuring Resources Productivity Index

Question or Variable	Mean Productivity (DSL/MM)			Change (DSL/MM)
Percentage of code for delivery	0-90% 159	91-99% 327	100% 265	106
Code classified as non-mathematical application and I/O formatting programs	0-33% 188	34-66% 311	67-100% 267	79
Number of classes of items in the data base per 1000 lines of code	0-15 334	16-80 243	>80 193	141
Number of pages of delivered documentation per 1000 lines of delivered code	0-32 320	33-88 252	>88 195	125

~~Software Resource Models and Measures~~

Issues with the model

Contributions

- Empirical model based upon historical data
- Looks at the relationship between several variables
- Show the relationships between size and effort can be used for characterization, evaluation, and prediction
- Takes into account many variable classes, e.g., experience, methodology, customer interface, context
- Uses subjective metrics (ordinal scale)
- Shows the relations are not always monotonic

Software Resource Models and Measures

Issues with the model

Concerns

- Definition of values on an ordinal scale metrics
- What's the underlying distribution for each metric
- There are lots of variables relative to the degrees of freedom, the data points
- The values for size or inconsistent across languages
- Doesn't take into account the effect of combined variables
- Many of the variables are interdependent
- Correlation is not cause effect