

Correcting a flaw in the Project Model

Motivation

The system dynamics modeling process is an iterative process, moving repeatedly through the stages of conceptualization, formulation, testing, and refinement. This is an exercise in reformulation to fix a flaw in the Project Model in Richardson and Pugh (1981). As a step in learning how to build useful models, the exercise has two desirable characteristics: (1) we start with a more or less complete model that runs, shows plausible behavior related to the reference modes of the study, and has, for the most part, reasonable structure; and (2) We get to experience the sort of thinking, modeling, simulating, and rethinking that characterizes model reformulation and refinement.

The Problem

The simple project model in Richardson and Pugh (1981, 190-213) has two weaknesses. The first is that the formulation for perceived productivity of the workers on the project does not reflect what real actors in the system would do to assess productivity. We solved that problem in class by reformulating perceived productivity to be the ratio of cumulative perceived progress (tasks) over cumulative effort (person*months), yielding the number of tasks per person per month completed on average up to that point in the project.

The second problem is the subject of this reformulation exercise. In the simple Richardson and Pugh model, the rate of detecting rework is formulated as the stock of undiscovered rework divided by the time to detect rework. This oversimplification has the dramatic flaw that rework would continue to be discovered when there are no workers active on the project.

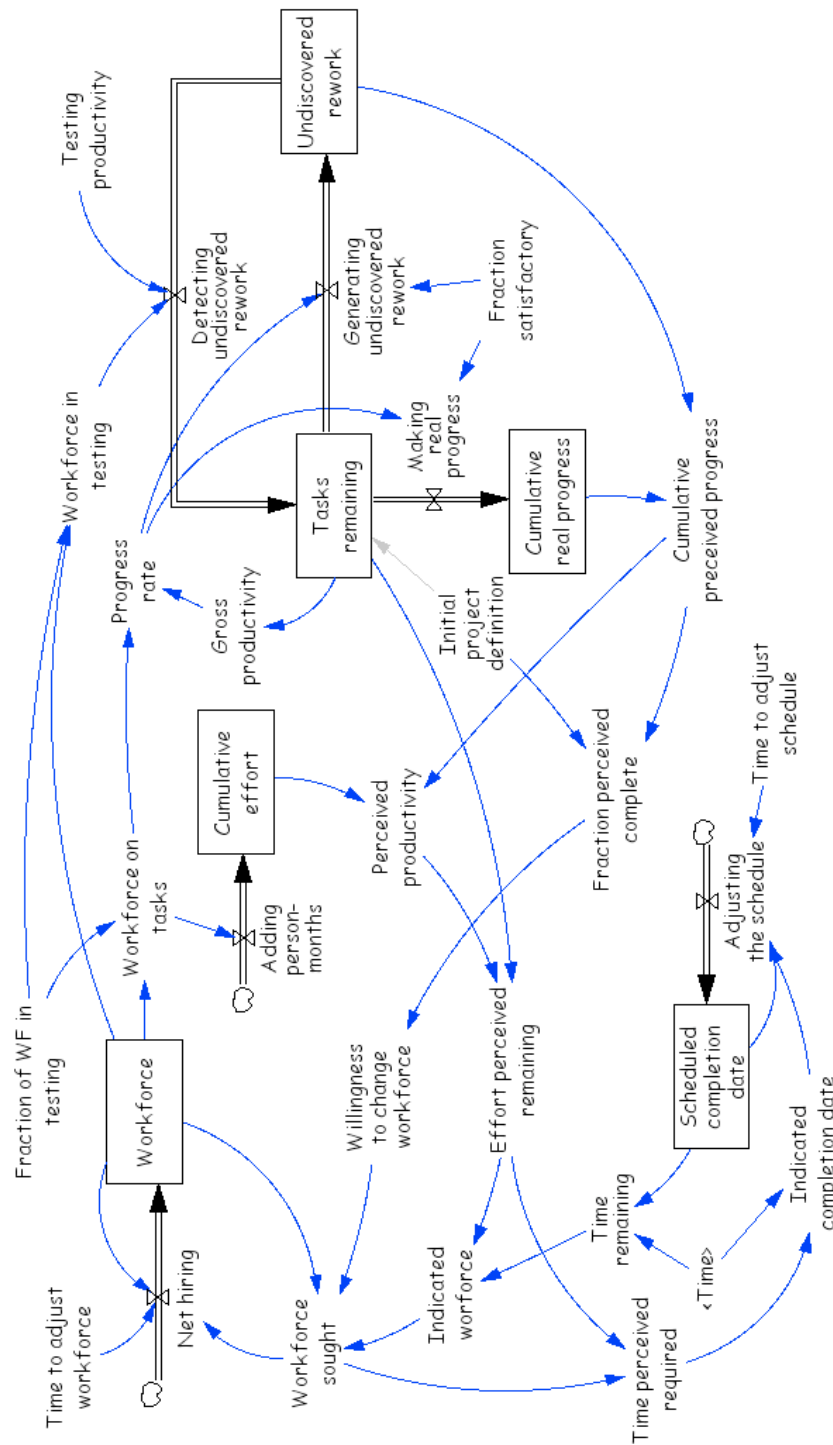
The Task

We must reformulate the detection of rework so that it is done by people. The plan is to separate the project workforce into two groups, Workers on Tasks and Workers in Testing. Workers on Tasks make progress, with some productivity, much as in the Richardson and Pugh model. Workers in Testing discover rework, with some Testing Productivity, measured in rework tasks discovered per person per month. We will approach this problem in three steps.

- 1) The model downloaded from the class web site is the Richardson and Pugh model with a corrected formulation for perceived productivity, but with the flawed formulation for detection of undiscovered rework. Your first task is to reformulate the model to look like the figure on page 3. You will create new *constants* for the Testing Productivity (tasks/person/month) and the Fraction of the Workforce in Testing (dimensionless). From these, and the diagram on page 2, you should be able to formulate the other new parts of the model and simulate it.

Experiment with various values for these two parameters to learn about how the model behaves with them as constants. You should be able to get a wide variety of behaviors, including making Undiscovered Rework go negative!

What you learn here should tell you something about how these two parameters need to be variables embedded in feedback loops, rather than constants. Hand in a couple of typical simulations, which show something revealing, and comment briefly on what you learned from your simulations.



2) Now reformulate the Testing Productivity to be a function of Undiscovered Rework.

[Your experiments in step (1) should tell you why this must be to prevent Undiscovered Rework from going negative. If not, go back to step (1). If still no insight, contact me.] Specifically, formulate Testing Productivity as a constant Normal Testing Productivity (tasks/person/month) multiplied by an Effect of Undiscovered Rework Remaining (a dimensionless graphical function). Create the equation for that Effect as an auxiliary LOOKUP function of Undiscovered Rework. [Since the number of tasks in the project is supposed to be 1200, you could let the x-axis of this graphical function range from 0 to 1200. But since Undiscovered Rework never gets that high, you could formulate the x-axis of this function to go from, say, 0 to 300.] You will know for sure what one of the points on this graphical function has to be to prevent Undiscovered Rework from going negative, but the rest of the graph is less well determined. Use your intuition.

And by the way, you are not to add or change anything else in the model at this point!

Normal Testing Productivity represents the number of flawed tasks a worker would discover per month. Pick a value for that parameter that makes sense to you, given the rest of the conditions in the model, in particular given that the gross productivity for *completing* tasks is set to be 1 task/person/month. (Would people be more productive at finding flaws or less?) Then set the numbers in the graphical function for the Effect of Undiscovered Rework Remaining, remembering that the “normal” (no effect) value for that multiplier would be 1.

Set the constant Fraction of Workforce in Testing to 0.1. Then make several simulations testing various *plausible* assumptions about the value of Normal Testing Productivity and the shape a values of the points in the graphical function for the Effect of Undiscovered Rework Remaining. [You can use Vensim’s “Set” button for these reruns.] Experiment until your formulation for Testing Productivity works plausibly in all conditions. (If it doesn’t, rework the constant and the graphical function until it does!)

Pick your favorite constant for Normal Testing Productivity, and your favorite graphical function for the Effect of Undiscovered Rework Remaining, and put them permanently into the model structure. Then simulate for a range of values in the constant Fraction of Workers in Testing. Hand in a couple of typical simulations, documented with the parameter values you assumed and experimented with and a Vensim graph of the graphical function you settled on for the Effect of Undiscovered Rework Remaining so we can follow what you did¹. Comment briefly on what you learned from your simulations.

3) With these simulations and observations you should be motivated to make the Fraction of Workers in Testing to be a variable, rather than a constant. So reformulate the Fraction of Workers in Testing to be a variable, computed [probably using a graphical function] from some quantity available in the model that you think real actors in the real system would use to guide

¹ The easiest way to get a nice little graph of this graphical function is to create a strip graph for Detection of Undiscovered Rework. You’ll see a graph for Testing Productivity. Copy and paste it into the file you hand in.

their policy of the fraction of workers they'd put in testing. You are formulating an endogenous *policy* for allocating workers to testing.

(And by the way, you are not to add or change anything else in the model at this point!)

Simulate your model with various assumptions in the functions for the Fraction of Workers in Testing and the Testing Productivity. Hand in some typical simulations, documented with Vensim graphs of the graphical functions so we can follow what you did, and comment on what you learn from your simulations.

[Be a scientist here: Vary only one thing at a time. Pick one graphical function for Testing Productivity and vary the function for the Fraction of Workers in Testing to test various policies actors in the system might use.

Then, if you want, pick a different function for Testing Productivity and repeat your same experiments with the function for the Fraction of Workers in Testing.]

4) Finally, comment briefly on what you are able to learn from this exercise about the problem the model is addressing – the problem of controlling overruns in large projects inherently affected by rework. If your simulations suggest new questions, be sure to mention those, since uncovering questions is a vital part of learning.