Massachusetts Institute of Technology
Sloan School of Management
Applications of System Dynamics
15.875

Prof. Jim Hines
Spring 2004

Guidelines for fifth project week and group presentation: Partial model of first hypothesis[1]

Purpose of fifth presentation: In your fifth week "on the job" you should begin modeling by choosing a hypothesis (i.e. a loop or two) and making a start on a stock and flow version as well as a start on the equations. In the breakout group presentation you will explain how you chose which hypothesis to model first, you will outline your strategy for converting this loop-hypothesis into a model and you will present a start on that task. As always we hope you will also present any insights that emerged during the week's effort.

Modeling Hypotheses. Most experienced SDers think of modeling as completing a series of "pieces" that will ultimately fit together. There are three ways of breaking a model into pieces: By loops (hypotheses), decisions and by sectors.

In this class we take the loop approach for deciding what to model and then a decision approach for deciding how to break the loop(s) up into separate views in Vensim. Loops always have dynamics; hence, by taking a loop-centric approach to our modeling, we will continually be adding to a model that has (potentially) interesting dynamics capable of yielding insights.

A sectoral approach, in contrast, divides the model into logical "blocks" – such as finance, marketing, customer, and production. It is possible and even likely that most hypotheses cut through some or all of the sectors. The sectors themselves may have loops, but they are often not the loops of most interest to the SDer and his client. Consequently, with the sectoral approach, examination of hypotheses and truly interesting dynamics must potentially await completion of the entire model when all sectors are finally linked together. Getting benefit from simulation is delayed until very late in the project. Further, SDers who finally "turn on" their model after six months or more of modeling sectors are turning on a very complicated model; they are likely to be bewildered by the resulting behavior. Finally, the objective of completing a loop provides guidance concerning the appropriate amount of detail and level of aggregation; without this guidance modelers tend to produce sectors that are needlessly detailed and disaggregate, thereby complicating the task of gaining insight from the model. In contrast, modelers who build their models hypothesis-by-hypothesis are continually

[1] Prepared by Jim Hines 1998. Revised March, 1999; July, 1999, March 2000, April 2004.

simulating and understanding complete-loop dynamics and their models tend toward an amount of detail that is appropriate to achieving insight and understanding... Hypothesis-oriented modelers' understanding grows along with their model.

Choosing which hypothesis to work on first. In order of *decreasing* importance the hypothesis you choose to model first should be:

1. Easy to model
2. Central to the loop diagram
3. Interesting or important to you or your client

Easy is the most important consideration. As the model progresses decisions concerning vocabulary, aggregation, detail, and basic architecture gradually get worked out. As these questions become increasingly settled a modeler is in a position to tackle increasingly difficult formulation issues. Further, as modeling progresses understanding of the system also increases which again increases the ability of the modeler to model the *rest* of the system. In brief, as modeling continues, further modeling becomes easier. To take advantage of this process, you want deal with the most obvious formulations first (when you need all the help you can get) and save the most difficult ones for last, when you'll be more prepared to tackle them.

Easy hypotheses are those that

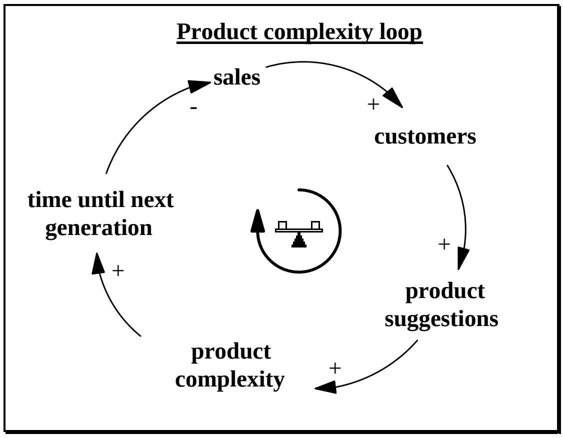
- you (or your client) understand thoroughly already
- are comprised of molecules you already know
- are small

A consulting challenge may arise here. Clients often want to start with the part of the model they understand the least. However, modeling what is poorly understood is almost certainly going to be difficult (it takes a great deal of knowledge of the actual system to create a good model). Fortunately, clients will usually listen to a modeler's well-reasoned opinion on what is really a question of modeling technique. In arguing for "your" starting place over another, its best to list for the merits of the structure you want to start with as opposed to listing the demerits of the client's preferred starting place or the shortcomings of his arguments. Still, for your own self confidence, it's good to know that the client's reasons for wanting to model the unknown are not as sound as he probably thinks. The reason most clients will focus on the piece of the organization they know least is that they hope to strengthen their greatest vulnerability. However their greatest vulnerability is not ignorance about a particular piece of the business but ignorance about how the entire system behaves and why. So the route that results in the fastest progress on modeling the greatest amount of the system is actually the fastest way to shore up weakness. Starting with an area that is difficult will mean a slow start. And, because difficulty tends to evaporate as modeling goes on, starting with something difficult means wasting time and effort that could better be used at the end of the project to refine analysis or policy solutions.

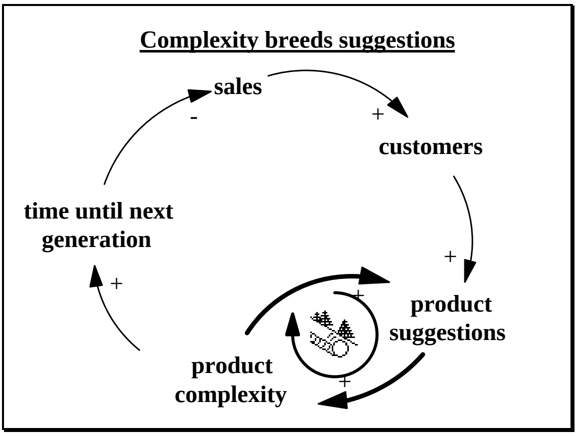
A **central** hypothesis (or loop) is one from which "hang" most of the other hypotheses (loops). The advantage of beginning your model with a central hypothesis is that (a) the work you do will be efficient in the sense that it will contribute to representing many other hypotheses, (b) you will not need to first model other (pre-requisite) loops, and (c) you will have many options for which hypothesis to model next. Further after finishing that hypothesis you will be able to choose from a larger number of additional hypotheses for your next modeling task. And this in turn, means that you can again choose a loop that is easy (and central and interesting).

An **interesting or important** hypothesis is usually one, which you think will generate a problematic behavior pattern. As mentioned above, some clients will focus on hypotheses that they know the least about. However, ignorance is *not* what we mean by "interesting".

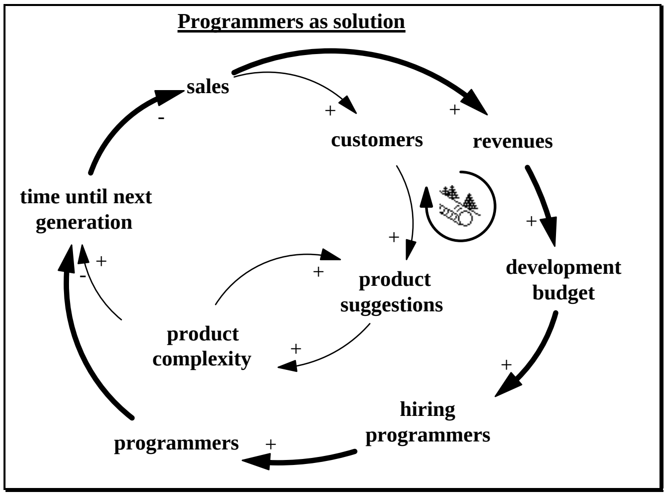
Example. For example say I had the following hypotheses about the problem with rising product development times.



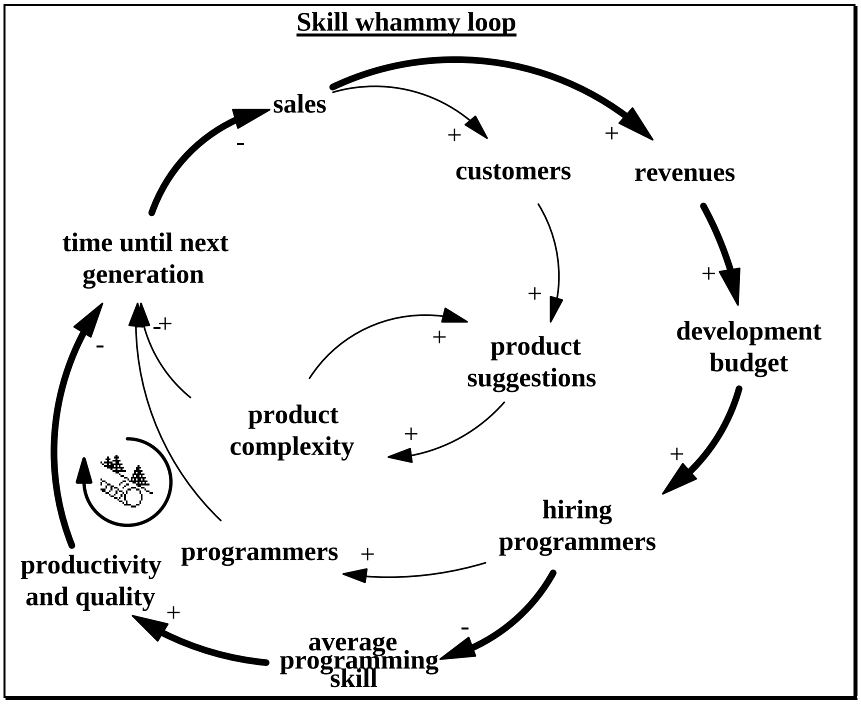
Product complexity loop



Complexity breeds suggestions



Programmers as solution



The *product complexity* hypothesis might seem like a good one to start with: The loop is the most central one and it is interesting in that it probably will generate the major concern – rising product development times. Unfortunately, it fails the most important criteria: I am not sure how to model it.

The *programmer as solution* hypothesis is not so interesting because it will probably generate *falling* product development times. On the other hand, this hypothesis is fairly central and, most importantly, I have a good idea how to model it.

The other two hypotheses are out of the running: The s*kill whammy* loop requires the programmer-as-solution loop as a prerequisite, and the *complexity breeds suggestions* loop is not central enough to focus on first. The best choice for me is the *programmer as solution* hypothesis.

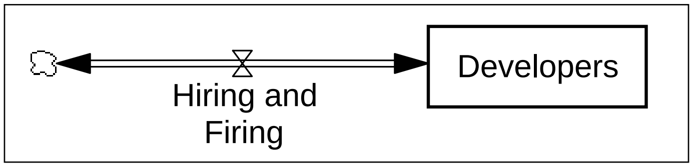
Strategy for converting a loop into a model. A good strategy is to start with what is easiest and most tangible and work towards what is hardest and most abstract. So, first identify any variables in the loop that obviously are levels – these will be levels that are actually physical accumulations (e.g. inventories) as apposed to more abstract accumulations (e.g. fatigue). Place those into Vensim. Then, hook up the physical levels in the obvious ways (e.g. if raw materials flow into work in process, hook up the stock of raw materials to the stock of work in process via a pipe) – that is, create any important cascades (chains) of physical stocks.

Next, think in terms of decisions controlling the valves (flows) and model each decision, beginning with the easiest and most tangible. Each decision can potentially go on a separate view, although if decisions are simple enough you can put several on the same view. To model a decision, start with molecules that you know. Then identify parts whose formulation you have some sense for, even though you do not have a molecule in mind. Finally, identify the "mysteries" – the parts that you have no idea how to model.

Note the recurring theme here: Start with what's easy and move to what's difficult. This principle applies at the broadest level of organization (at the hypothesis level) and at the smallest level (the decision level). Wherever you happen to be at the moment, try to find something easy to do next.

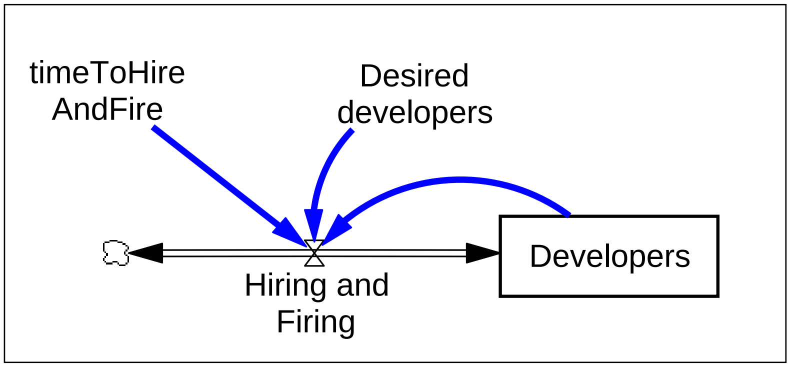
To continue with the above example, an obvious level in the programmer-as-solution loop is the stock of programmers. The development budget may be a stock, but if so it is more abstract and so we'll hold off for now.

Once you have spotted the obvious physical stocks, put it into the model:



Developers = INTEG(HiringAndFiring, initialDevelopers)
Units: people
initialDevelopers = 100
Units: people

Next, consider the flows. The only flow here is hiring and firing. How are you going to model that? An obvious molecule is to use is the *CloseGap* molecule, which will yield:

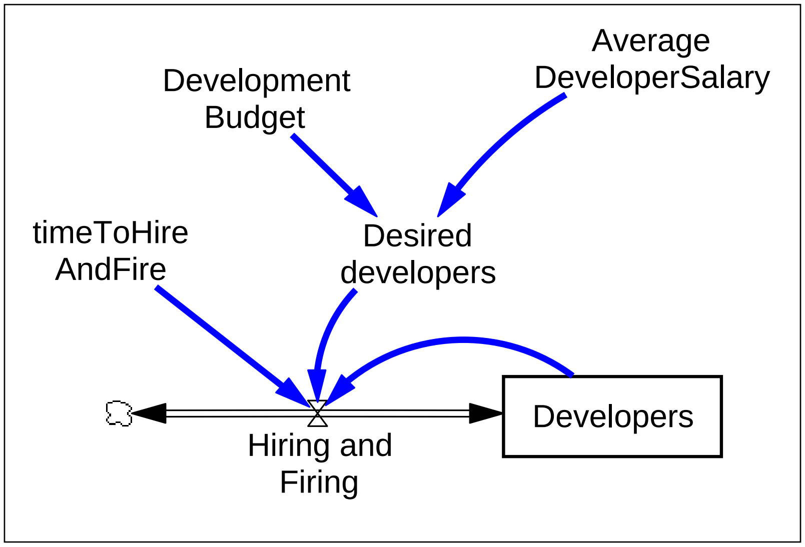


Hiring and Firing = (Desired developers-Developers)/timeToHireAndFire
Units: people/year
timeToHireAndFire=1
Units: year

Note the above structure is the *workforce* molecule. If you'd been planning on using the workforce molecule from the beginning, fine. If not, no problem – you've gotten to the same place by combining lower-level molecules. The fact that your work turns out to be another molecule is an added bonus, perhaps giving you confidence that your structure will work.

The obvious formulation to think about next is the one for *desiredDevelopers*. You may be thinking in the back of you mind that *timeToHireAndFire* may not be constant, but *timeToHireAndFire* is not part of any of your loops and a constant will certainly work until you have taken care of more pressing formulations.

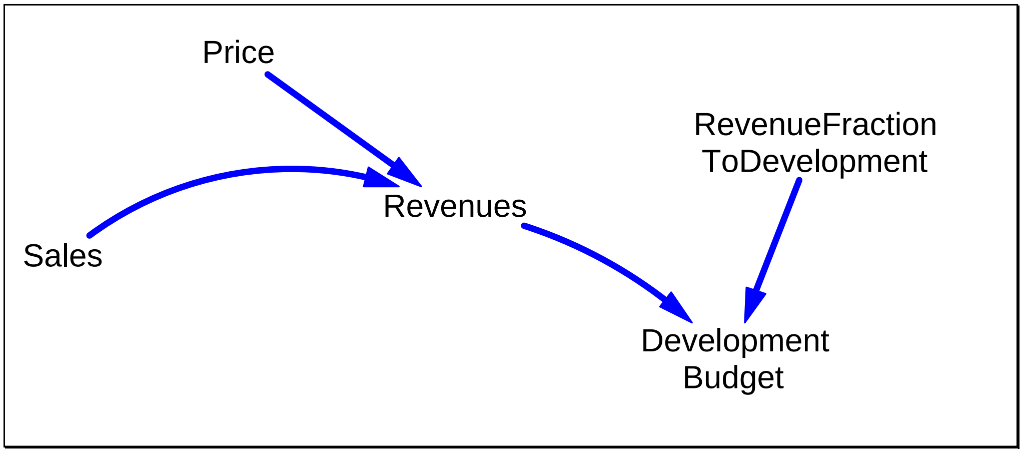
We know from the loop that the workforce is determined by how many people we can afford, so the *Workforce From Budget* molecule would make sense here. Note, that we could put this on a separate view, but things are still relatively simple, so we'll keep adding to the existing view.



Desired developers = *DevelopmentBudget/AverageDeveloperSalary*
Units: people
AverageDeveloperSalary=50000
Units: $/(year*person)

What should we formulate next? Well, *AverageDeveloperSalary* is not a constant in the real world, however its not part of any of our loops, so we'll keep it a simple constant at least until the time (if ever) when formulating a more complicated equation for *AverageDeveloperSalary* becomes the *next* most pressing thing to model – which it certainly is not now). On the other hand, Development budget is part of our loop and so that where we will turn our attention next.

No molecule comes to mind for the development budget, perhaps, I can see an easy formulation for it, given the fact that our loop says that as revenues go up, so does the development budget. If not, I'd stop here in terms of preparation for the class, and would talk to my client about it ASAP.

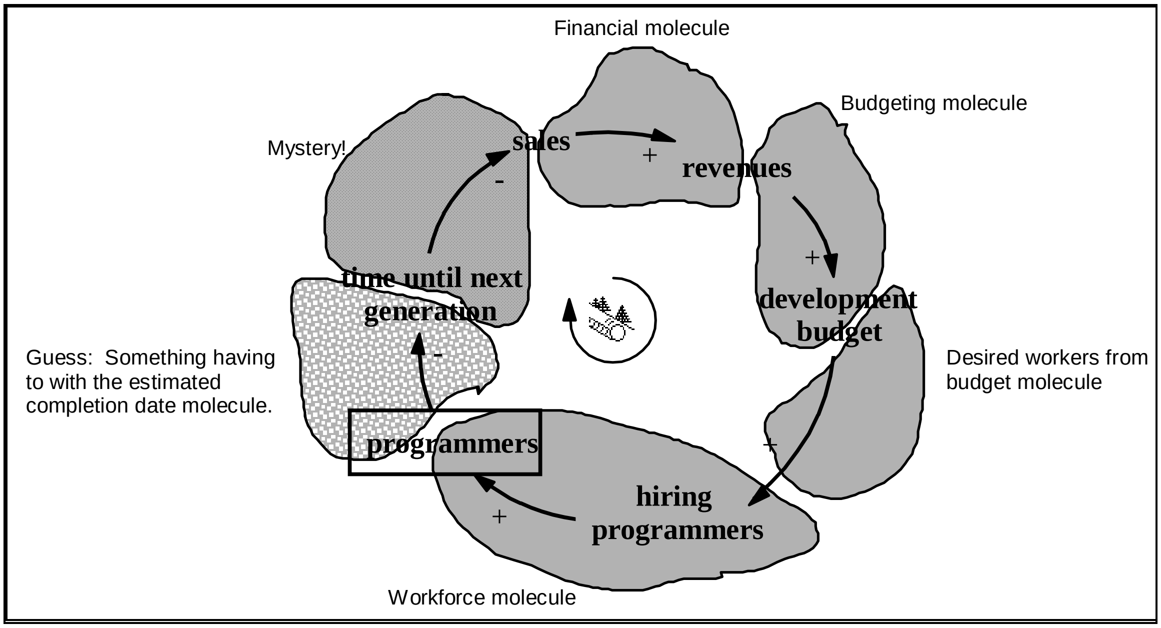


*DevelopmentBudget=Revenues*RevenueFractionToDevelopment*
Units: $/year
RevenueFractionToDevelopment=0.4
Units: fraction
*Revenues=Price*Sales*
Units: $/year
Price=100
Units: $/box

The next ting to model is obviously sales (price is not part of my loop, so I'll keep it a constant). The model says that sales go up as time until next generation goes down. Unfortunately, nothing comes readily to mind about how to model sales. I need to go back to my client and chat a bit about how sales come about in the company. For now I'll stop, knowing that I would be in good shape to give a nice presentation on this, the fifth handout.

The modeling process in brief. In brief the modeling process is always to work on the easiest thing if possible. So starting at the top, find the easiest hypothesis. If a few hypotheses tie for this honor, then choose the most central one. If we still have choice, choose the most important.

When deciding which hypothesis will be easiest, you'll want to think about whether or not you have a sense for modeling each "piece" of the hypothesis. Often this will mean identifying pieces for which a molecule comes to mind. It may be a good idea to actually draw the hypotheses and make notations about where different molecules *might* go. If we'd done this for the example above, we would have started with an even better picture of the loop to model. Our loop to model would have had the following additional annotations:


Financial molecule
Budgeting molecule
Mystery!
sales
+
revenues
-
+
time until next generation
development budget
Guess: Something having to with the estimated completion date molecule.
-
Desired workers from budget molecule
+
programmers
hiring programmers
+
Workforce molecule

Having this diagram would have made our task even easier. I recommend that your team actually annotate all of your hypotheses with molecules that come to mind and use this information in choosing which loop to start with. If few or no molecules come to mind, do NOT start flipping through the documentation searching, instead make a mental note to study the molecules a bit more when you are *not* modeling.

At any rate, begin the actual modeling process by identifying physical stocks and putting them into Vensim. Then concentrate on the rates of flow into and out of the molecule and begin "backing up" along the causal chain(s) of the loop(s) in your hypothesis.

Use Molecules and Views. You may be concerned that molecules may not adequately represent how your client company actually does things. This is a good concern, and as soon as you can you will want to run the structure by your client. However, with very little effort (and, hence, little at risk) you can start your model before talking to the client as long as you are working from molecules or easy structures.

Each view should fit onto the computer screen without scrolling. Think of each view as being organized around a particular decision: The first view above deals with deciding how many people to hire and fire (the view also includes some decisions "up the chain"). The second view deals with the budgeting decision.

On a more minor note: try not to split molecules across views. Other modelers will be able to understand your model faster and better if they can "chunk" it into molecules.

An important note: The process of using molecules is **NOT** one of flipping through the molecule documentation looking and looking for something you might be able to use. Flipping through the documentation can bring on an acute case of "molecule-itis". Sufferers of this dread disease see a molecule that might conceivably, by some stretch of the imagination, have something to do with the model at hand and so they hook it in. The result is a meandering model consisting of near-aimless strings of molecules attached one to the other. Just because a molecule *can* be hooked into a model doesn't mean it *should be.*

When modeling its best to rely on what is in your head and in your client's head. What comes easily to mind is usually appropriate. Further a molecule that comes to mind is usually a molecule that is understood. Understanding is important, because often you will need to modify (sometimes extensively) the chosen molecule. If you don't know or spot a molecule that is right on – no problem; you'll just need to invent something. System dynamics is all about creativity and invention.

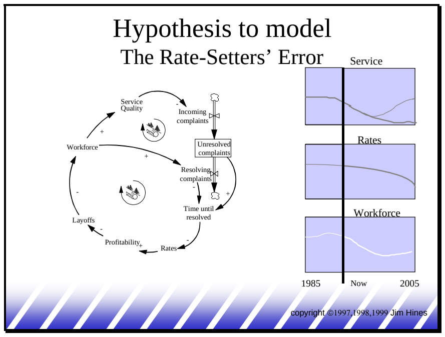
If you get stuck, go to your client, *not to the molecule documentation.* The most common reason for not being able to model something is simply not knowing how real people in the system actually behave. Simply calling your client and saying "let's talk a little bit more about hiring" will usually get you unstuck.

A start on a model. Remember you don't have to complete the model of your first loop this week. Completing the model of your first loop will come next week. For now, just model what you can, either making up the structure or using molecules.

Insights. As you model your loops, you will realize things about the loops that you hadn't realized before. Perhaps an important distinction will emerge, or you will realize that a concept has a time constant attached to it that is remarkably short or long. As always, record any insights and please share them with your breakout group.

Sample Slides

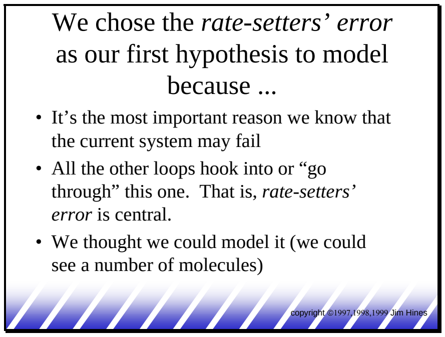
Slide 1



Hypothesis to model

The Rate-Setters' Error

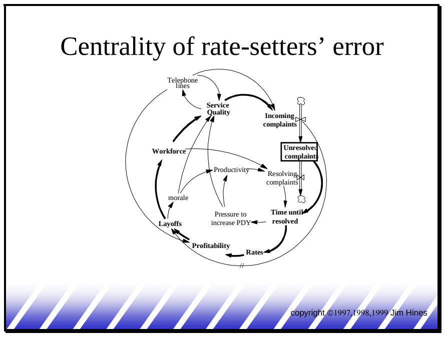
Slide 2



We chose the *rate-setters' error* as our first hypothesis to model because ...

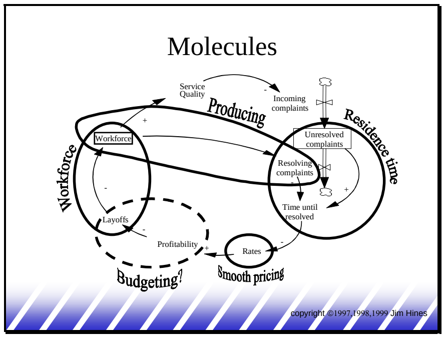
- It's the most important reason we know that the current system may fail
- All the other loops hook into or "go through" this one. That is, *rate-setters' error* is central.
- We thought we could model it (we could see a number of molecules)

Slide 3



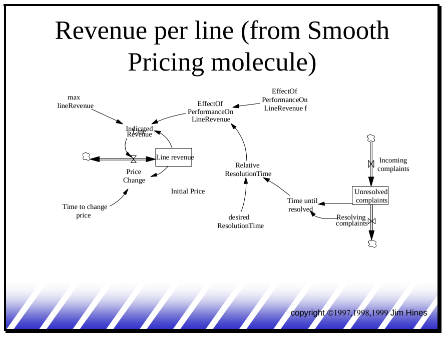
Centrality of rate-setters' error

Slide 4



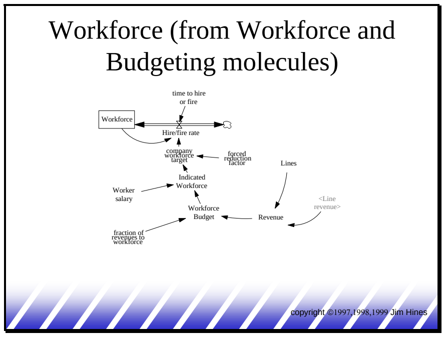
Molecules

Slide 5



Revenue per line (from Smooth Pricing molecule)

Slide 6



Workforce (from Workforce and Budgeting molecules)