# Thinking in Systems: Practical Applications

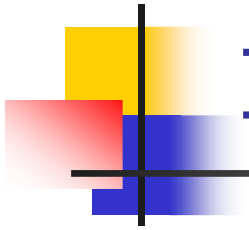
Jerry Thomas
Milwaukee School of Engineering
Contact information at http://decisionprocesstheory.com/

February 26, 2015

Northwestern Campus—Technical Institute

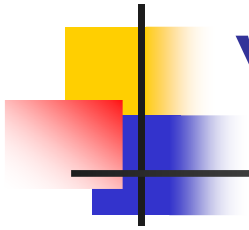
# Why Systems Thinking?

- Are systems dynamics tools of value in industrial applications? For example:
  - Manufacturing
  - Management Problems
  - Software Development
- What common tools are often used?
  - Non-systems Dynamics Statistical approach
- Are these the same?
  - NO



# Illustrative Story—what goes wrong

- Consider Large Software Projects: > 1 Million Lines of Code
- Experienced Staff
- Well Known/influential Customer
- Add 600,000 Lines each Release
- 2 year release cycle
- Project was 2 years late! Why?



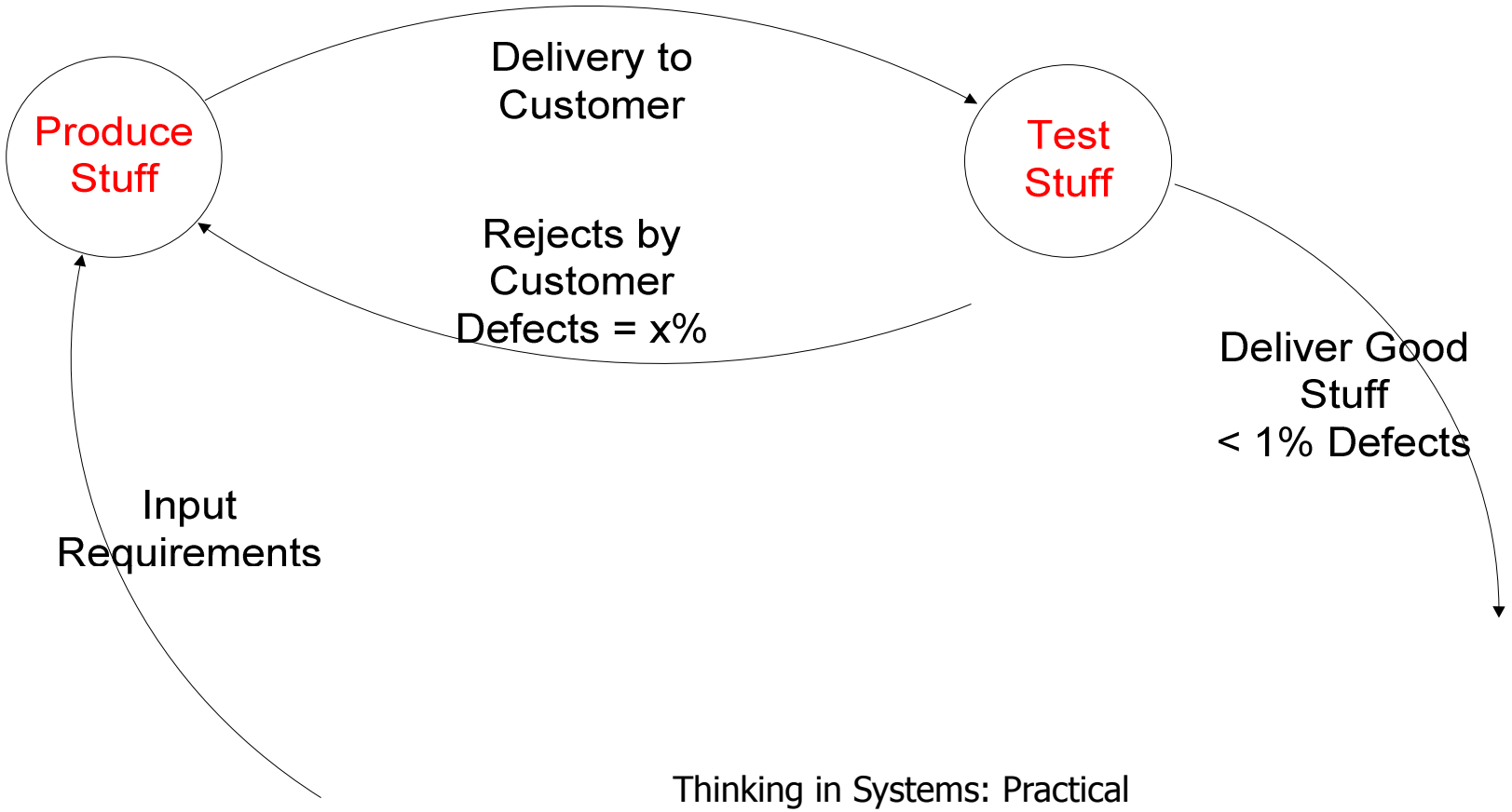
# "Statistical" Model Assumptions

- Customer expects to make money on product, which is "cutting edge"
  - Makes last minute changes
- Customer is willing to pay for product changes
- Customer increases size of project by 50% midstream, is willing to pay 50%
- Vendor expects to hire 50% additional staff for added work
- Conforms to statistical analysis

# Restatement of problem

- Will cost of project increase by only 50%?
- Will additional staff deliver almost on time?
- Statistical Model says yes
  - The vendor had an entire division to back up the "yes"
- Vendor and Customer made agreement based on "yes" analysis
- When project was significantly late, consultant hired

# Post mortem by Systems Dynamics Consultant—Executive Summary



# Case Study #1—Past History 10% Defect (basis of "yes")

| Cycle | Good | Bad | Time | On-Time? |
|---|---|---|---|---|
| 1 | 0.9 | 0.1 | 4 month | Yes |
| 2 | 0.99 | 0.01 | 4 month | Joint Testing |

# Case Study #2—50% Defect

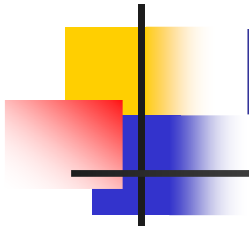
| Cycle | Good | Bad | Time | On-Time? |
|---|---|---|---|---|
| 1 | 0.50 | 0.50 | 4 month | Yes But |
| 2 | 0.75 | 0.25 | 4 month | No |
| 3 | 0.87 | 0.13 | 4 month | $ |
| 4 | 0.93 | 0.06 | 4 month | $$ |
| 5 | 0.96 | 0.03 | 4 month | $$$ |
| 6 | 0.98 | 0.02 | 4 month | $$$$ |
| 7 | 0.99 | 0.01 | 4 month | Joint Testing |

# Why did Defect 10% →50%

- Systems Thinking questions
  - Was Staff Experience the Same for New People?
  - Was Staff readily available?
  - What is a Defect?
  - Was Project Size really an Issue?
  - Was Cycle Time an Issue?
  - Did it Matter that this was a Software Project?
- Consequence of such dynamical effects
  - Increase delivery time of > 2 years and $$

# End of Story: many other examples

- For Complex Organizations, Change has unanticipated Side Effects
- In this example, Cycle Time and Defects were Key Levers
- Process, Control, Discipline Required
- However, Not Always Easy to Identify and Implement
- Late and unexpected—Never Good, Always Much More Expensive than Hoped!



# Note inherent conflicts

- In the above example
  - Customer does not want to pay more than what the features are worth to them
    - 50% more features means 50% more cost, which is what they have always paid
  - They want the delivery to be on an agreed schedule
  - Vendor wants to be paid for their work
  - Vendor has no direct control over delivery time given acceptance criteria

# Systems view—arbitration

- By agreeing to a systems thinking view
  - Both vendor and customer agree to the systems outcome
  - Ship builder story was what sold us on the consultant—The story is … (similar)
  - Both have to agree on the input data
    - This turns out to be easy in practice
- What is hard to accept at first:
  - Systems output is not always intuitive
  - Often disagrees with the "statistical" view
- This is like arbitration!
  - Must agree on a common and common sense framework

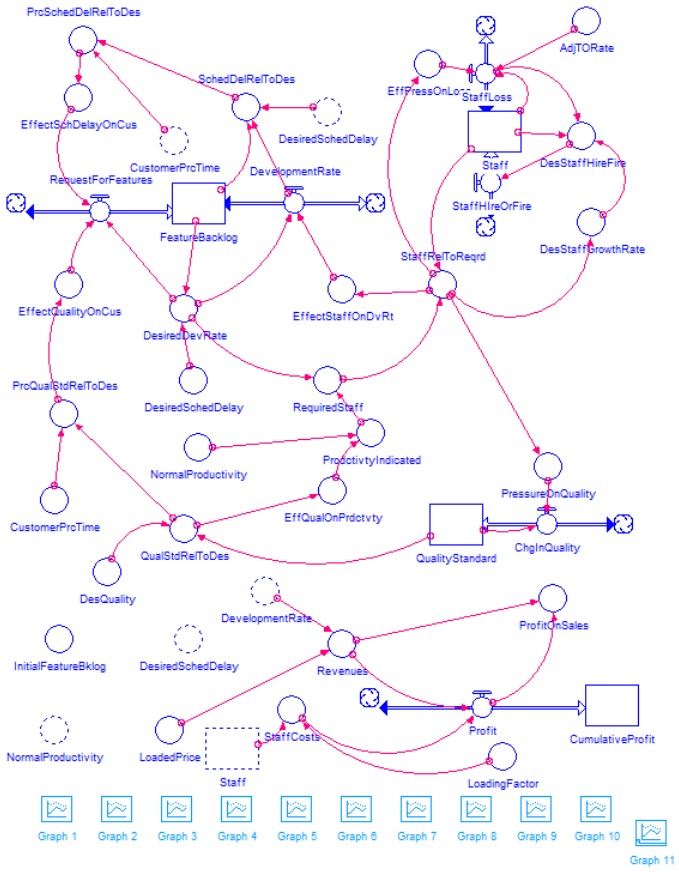
# Tools to help gain agreement

- The above argument appears qualitative
  - That is misleading
  - The consultant spent significant effort uncovering facts from the organization—E.g.
    - How many people were on the project
    - How much did the staff size grow
    - When did the delivery really occur
- To go quantitative—need tools, organization buy-in
- Note that some managers prefer qualitative arguments, because nobody is accountable

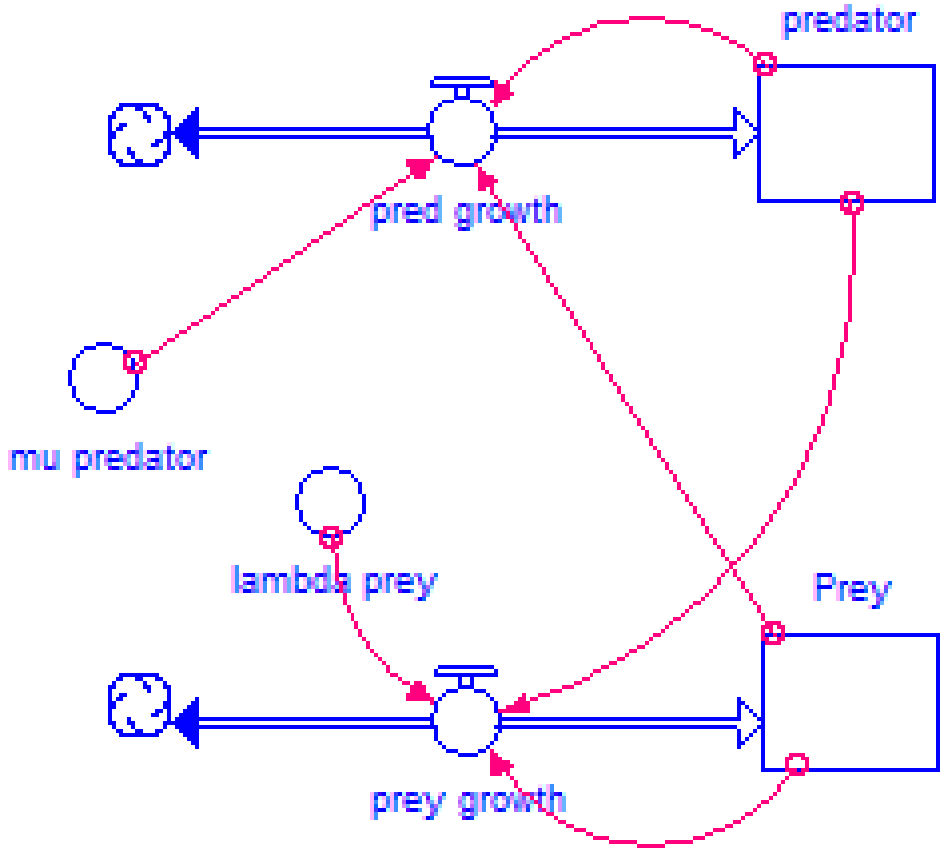
# System Dynamic Tools

- iThink is one company that provides "spreadsheet" tools to solve system problems
- Helps get agreement on two levels
  - Picture level: generate the "Thinking in Systems" picture
  - Quantitative: add the formulas that capture the quantitative organizational insights

# Project example



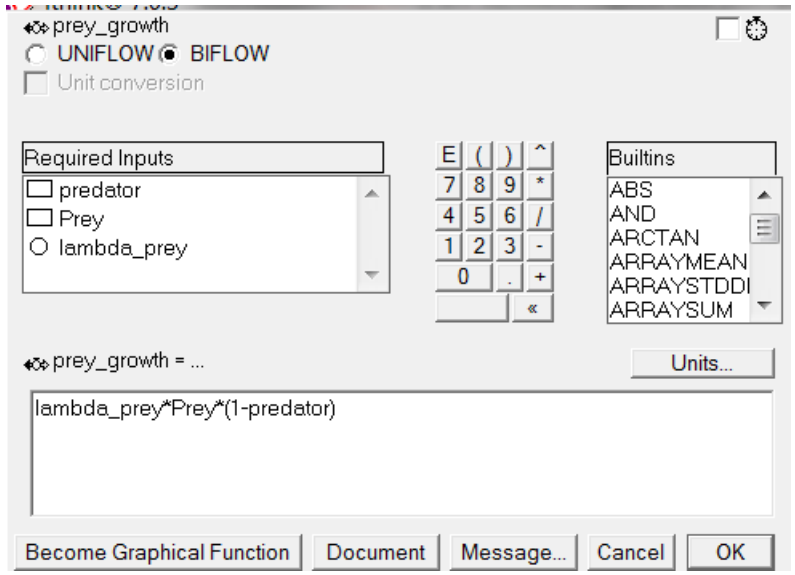
# Simpler model—picture only



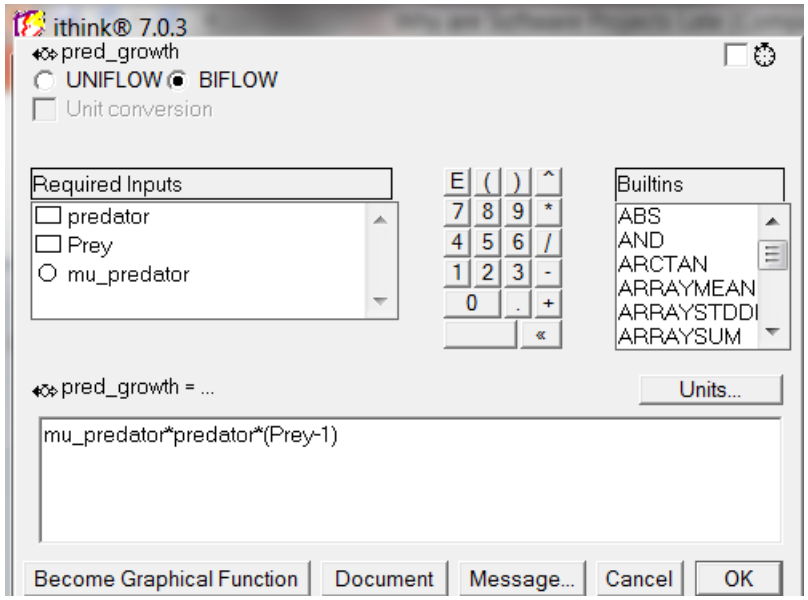
# Qualitative

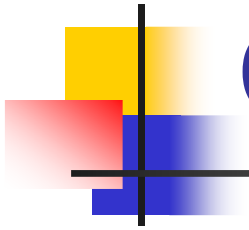
- Predators grow by eating prey
- Prey die when eaten
- Each population grows proportionate to its current population modulated by its birth and death rates

# Quantitative: Prey



# Quantitative: Predator

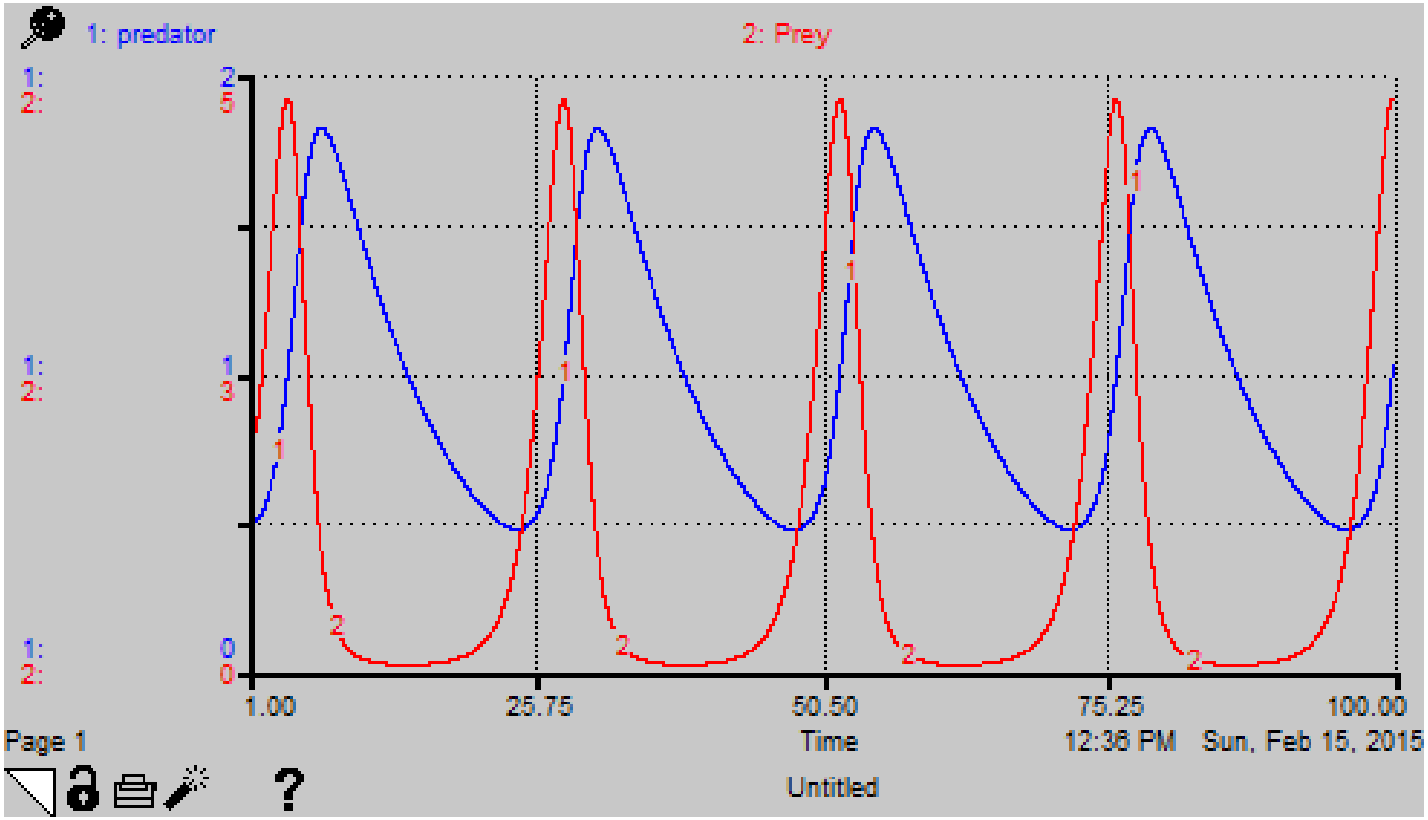




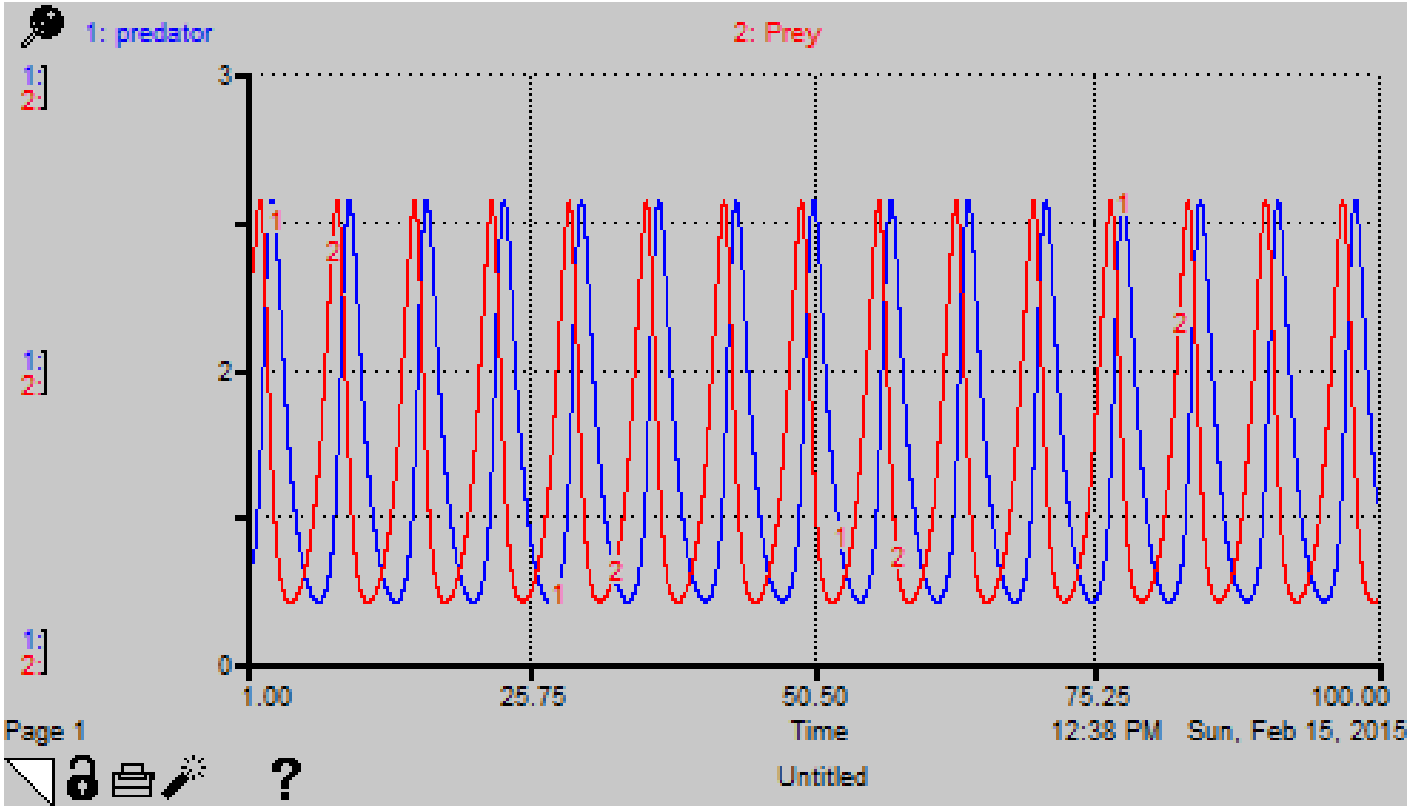
# Composite non-linear effects

- Deming said, as I recall, linear effects are easy to visualize.
  - Hence the growth and death of predator and prey are easy to understand
  - So is understanding why experienced software developers have higher productivity and produce fewer errors
- What is not easy is to visualize how these effects interact into values of interest

# Low predator rate

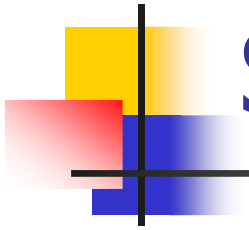


# High predator rate



# Note:

- For quantitative changes in one rate
  - Qualitative differences in populations
  - Qualitative differences in structure
- This says something important about applications where systems thinking is relevant.
  - Time structures depend on the details, not just on the picture
  - Analogous statements can be made about network connectivity: http://decisionprocesstheory.com/



# Sustainability

- You have been learning this to better understand sustainable structures
- Such structures are by their nature, interconnected structures in time and space
- The devil is in the details
  - It is all details
  - So get the details right