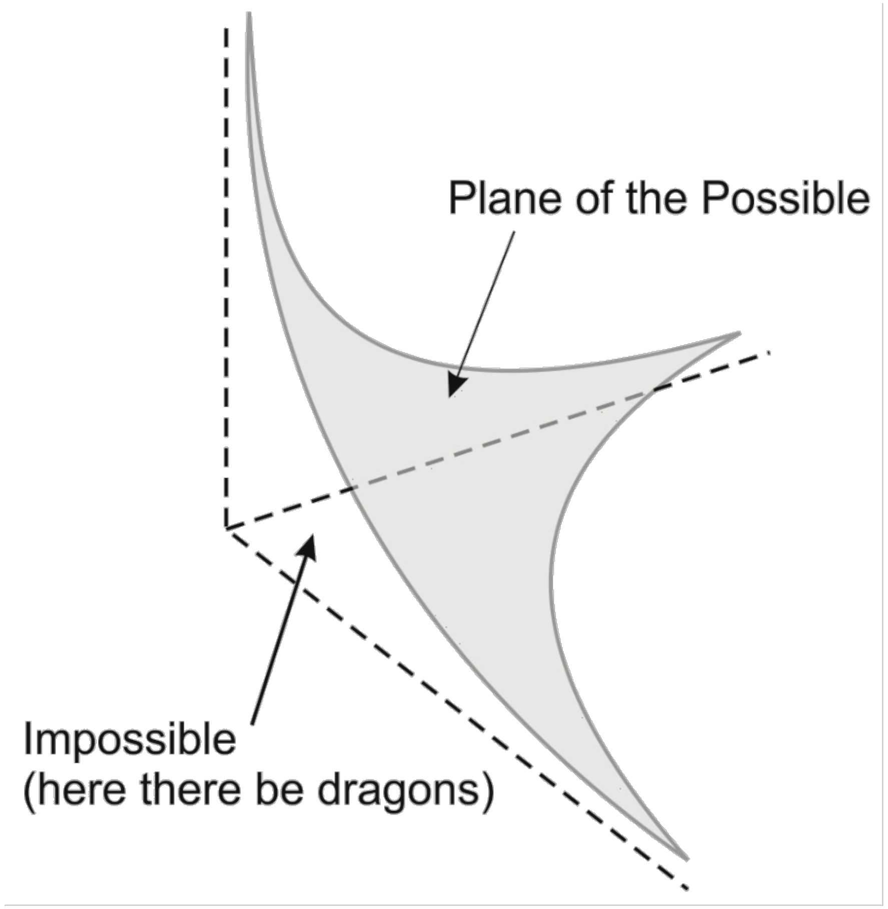
Engineer Versus Complexity

Russ White

Complexity is Impossible to “Solve”

- We cannot get to the lower left hand corner
- Instances –
  - Robust Yet Fragile (RYF)
  - Consistent/Accessible/ Partitionable (CAP)
  - Quality/Speed/Cost (QSX)
- *Just the way the world is built*



Complexity is Necessary

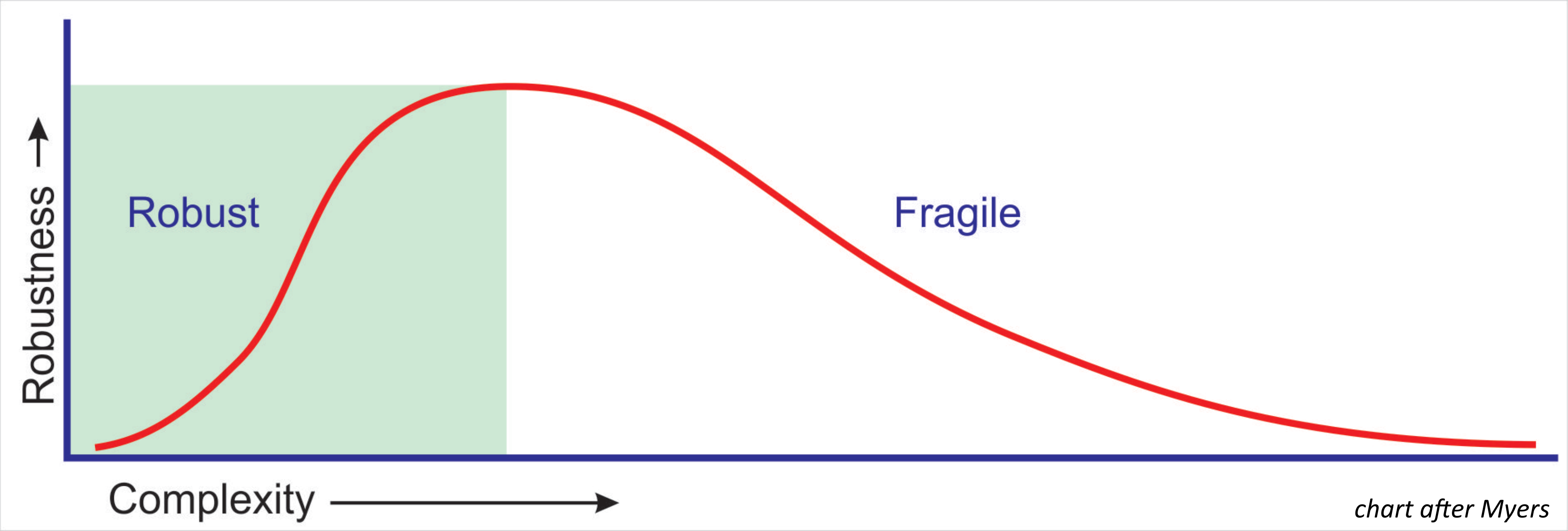


chart after Myers

"In our view, however, complexity is most succinctly discussed in terms of functionality and its robustness. Specifically, we argue that complexity in highly organized systems arises primarily from design strategies intended to create robustness to uncertainty in their environments and component parts."

See Alderson, D. and J. Doyle, "Contrasting Views of Complexity and Their Implications For Network---Centric Infrastructures", IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS, VOL. 40, NO. 4, JULY 2010

Abandon all Hope?

- No…
- Think through how to manage complexity in design and operations
- Think about how to approach complexity as an engineer

↑

We are here

Metamodel

(or process/technique)

to understand

(comprehend/process)

a complex system quickly

Developing the Metamodel

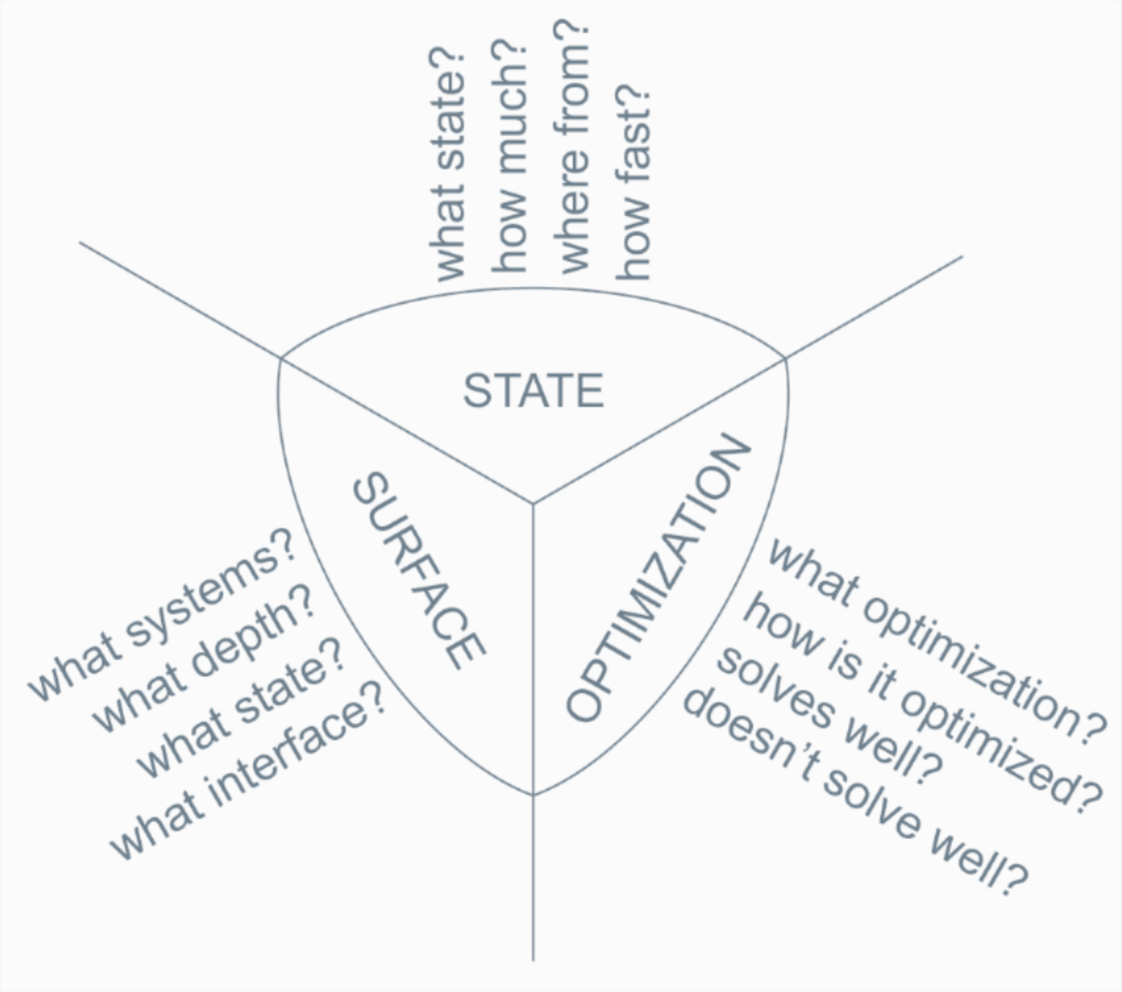
- Start with four common models/solutions
  - "Department of Defense" (DoD) model
    - *Origin of TCP/IP*
  - Recursive Internet Architecture (RINA) model
    - *http://csr.bu.edu/rina/*
  - Flow across a data center fabric
  - IP Fast Reroute
- Ask
  - Why?
  - What and how?
  - What is this like?

Why?

What problem am I solving?

Narrows my view – don't boil the ocean

What & How?



This is like?

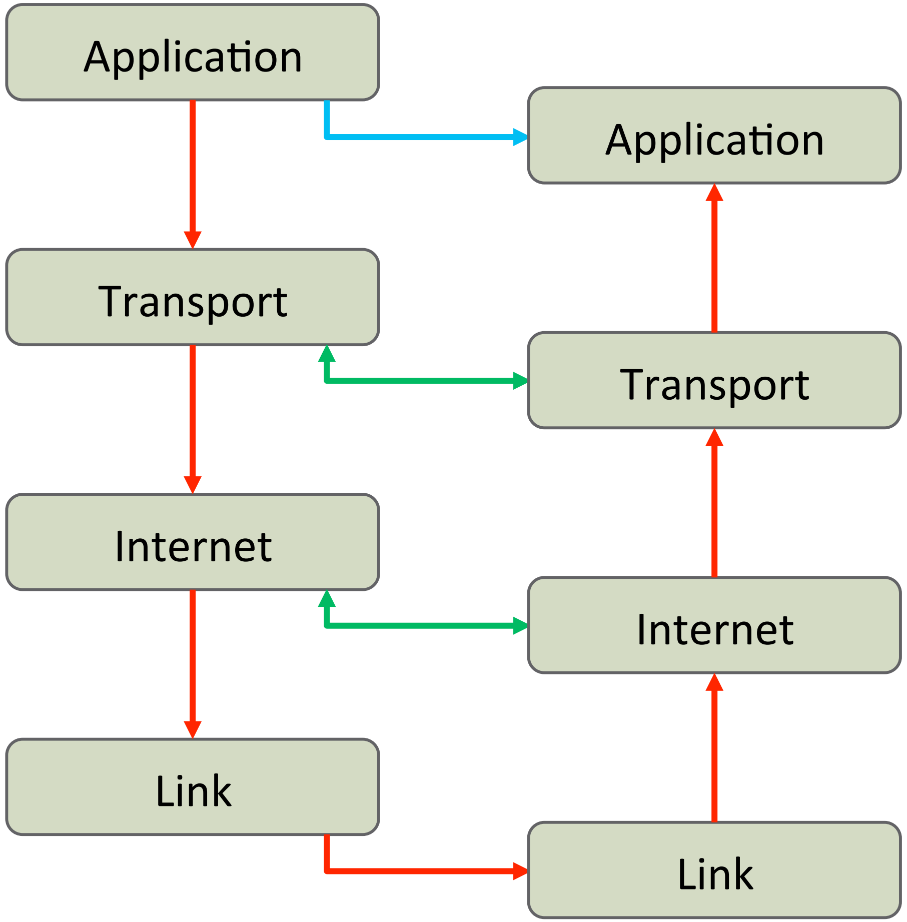
Has this problem been solved before?

How was it solved?

What are the problems with this solution?

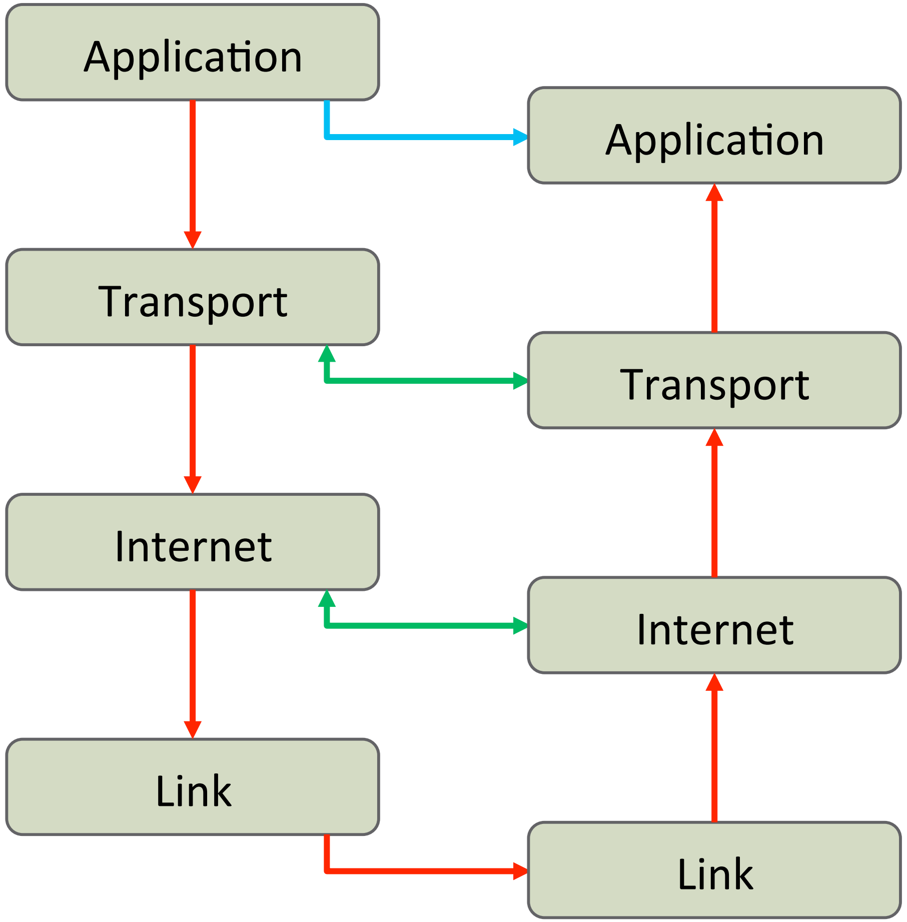
Layered Model Example (DoD)

- Move information between applications
- Carry information down the stack, across the wire, back up the stack
  - Split responsibilities up
  - Control amount of information managed by any given layer/application
- Carry information laterally between stack elements



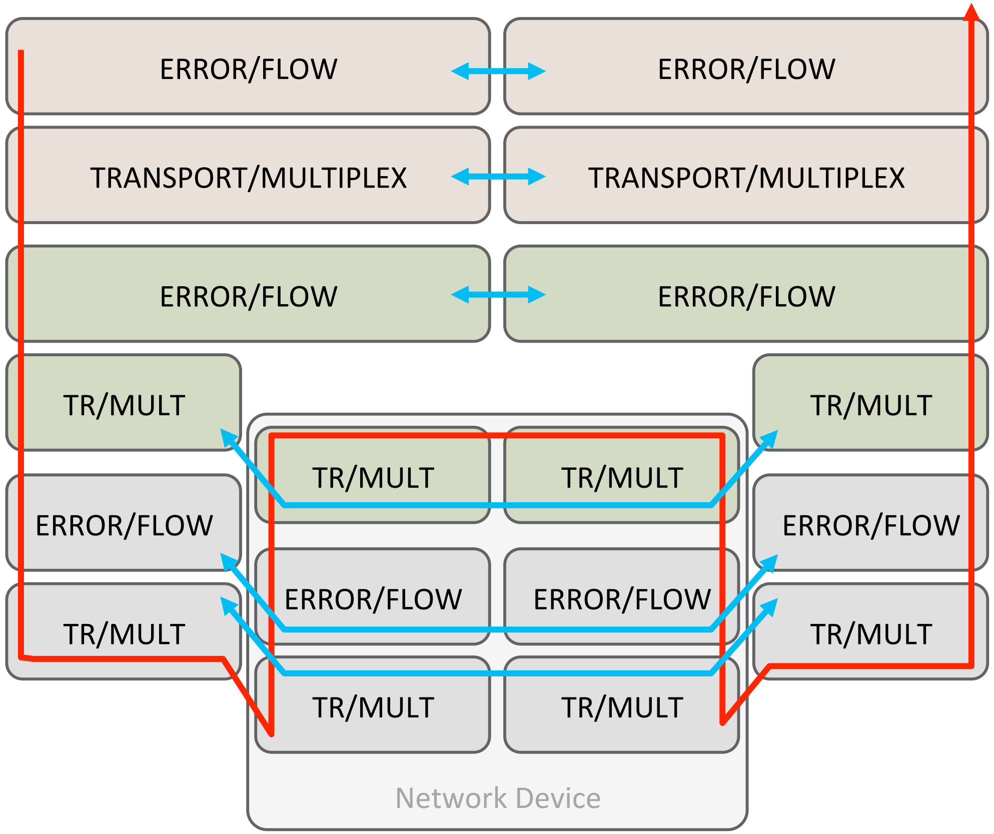
Layered Model Example (DoD)

- What questions does this ask/answer?
- Assigns responsibilities
  - Who knows what about each piece of data?
  - What functions does each process perform?
- Defines interactions
- Interaction surfaces
  - What is happening?



Layered Model Example (RINA)

- Ensure correct information flow between applications across the network
- What needs to be done to the information?
  - Group into natural divisions
  - Transport, Multiplex
  - Error handling, Flow control
- Who needs to do it?
  - Interface, host, application
  - Network device



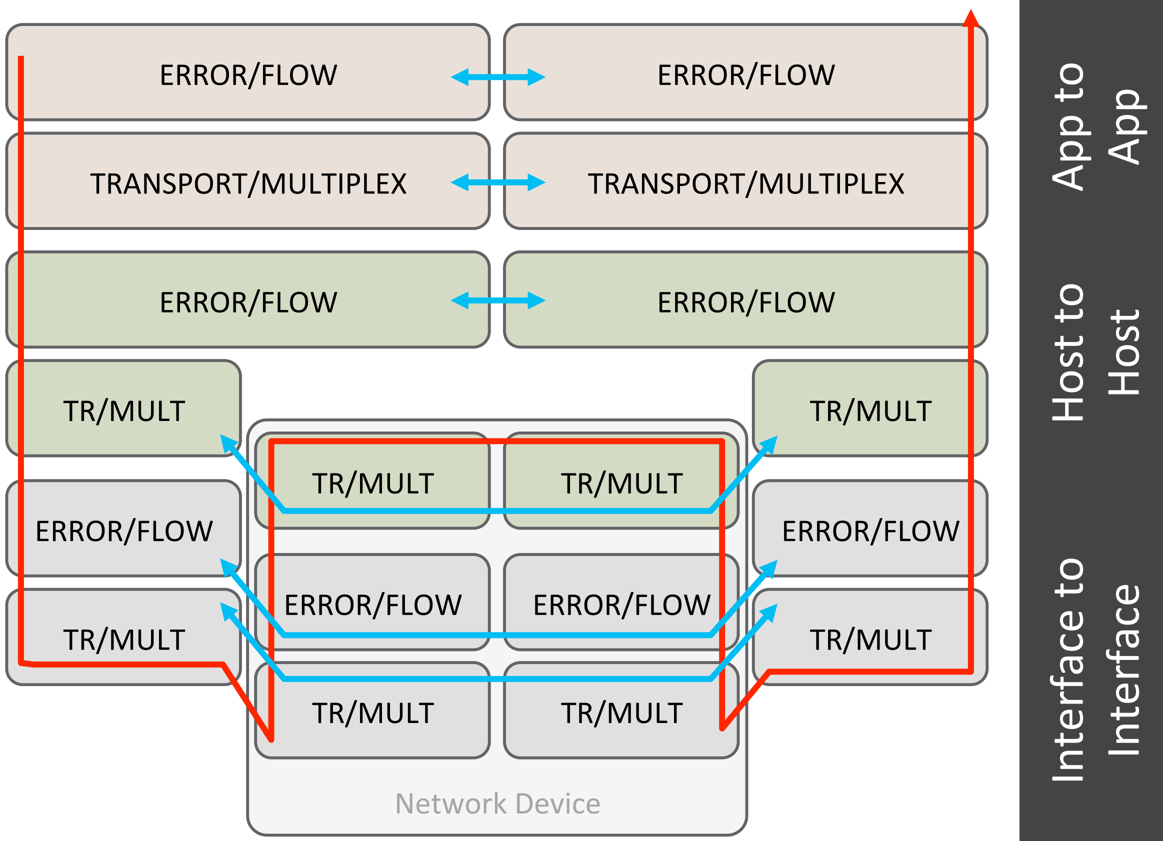
App to App

Host to Host

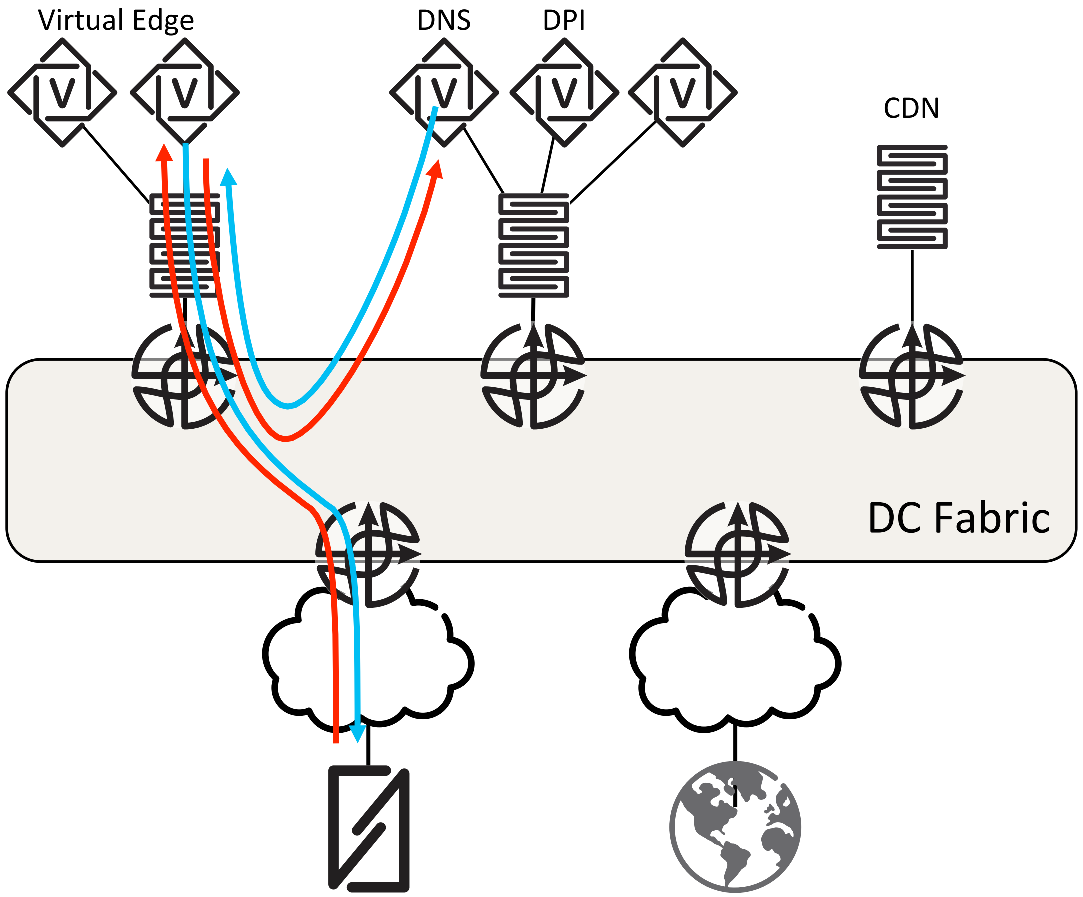
Interface to Interface

Layered Model Example (RINA)

- What questions does this ask/answer?
- Assign responsibilities
  - Who does what to the data?
  - How does each process accomplish it’s goals?
- Defines interactions
- *Interaction surfaces*
  - *How is the data being processed?*

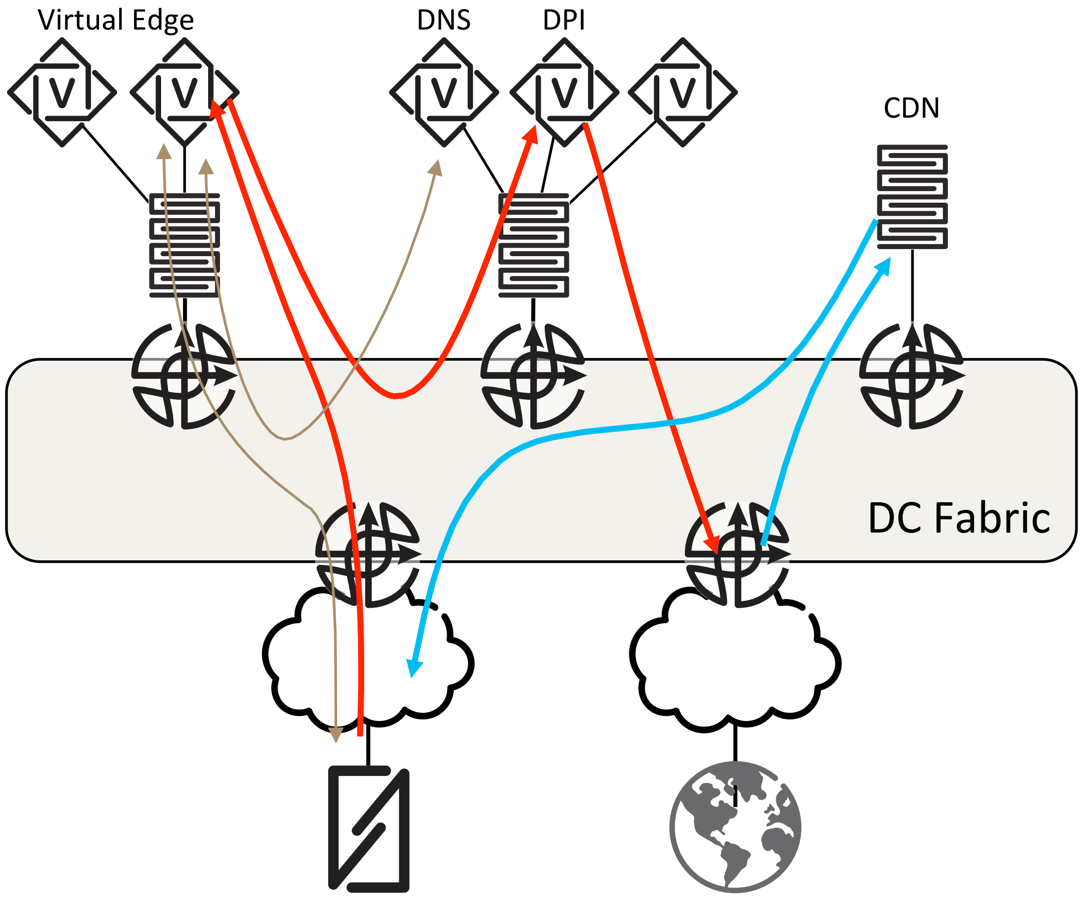


Data Flow Example



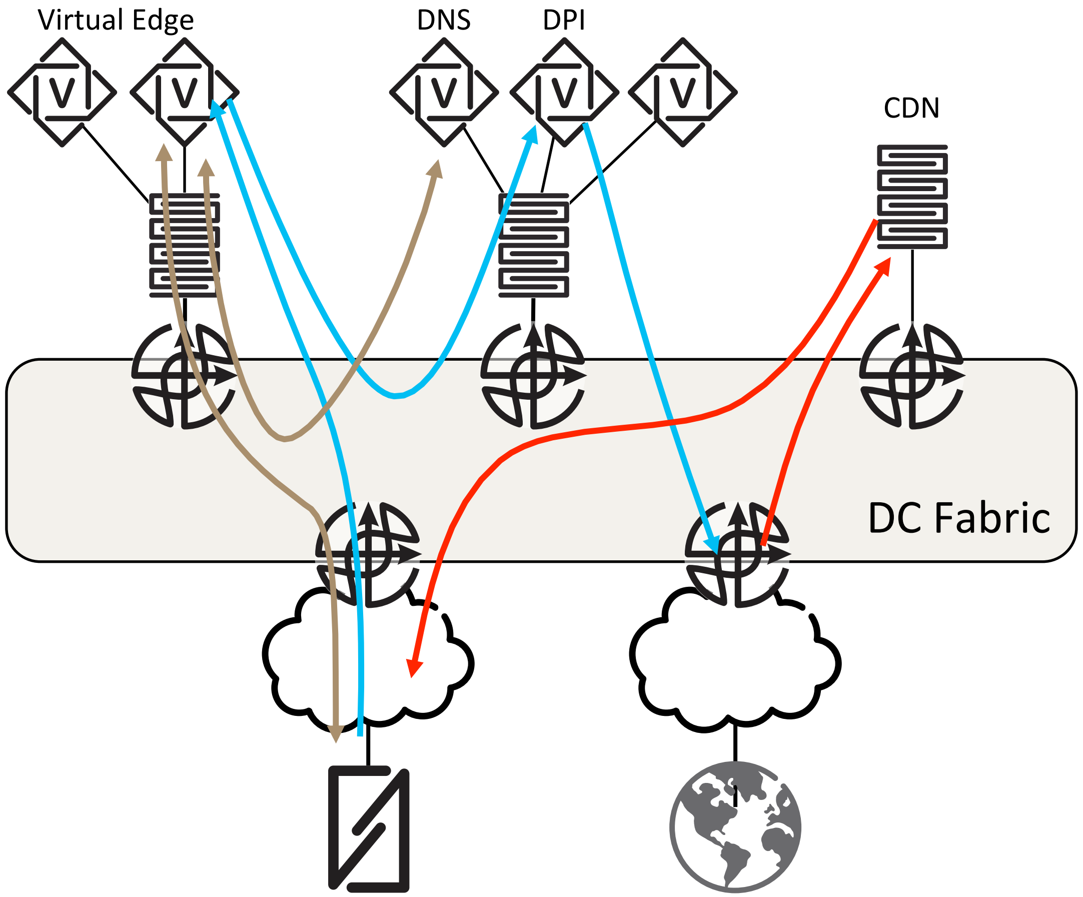
- Mobile device queries DNS server for service
- Packet must pass through virtual edge
  - OSS/BSS
  - Encryption
  - Other common edge services
- DNS server responds

Data Flow Example



- Mobile device initiates HTTP session with server
- Flow must pass through deep packet inspection (DPI) service
- Flow is redirected to a local CDN server for processing

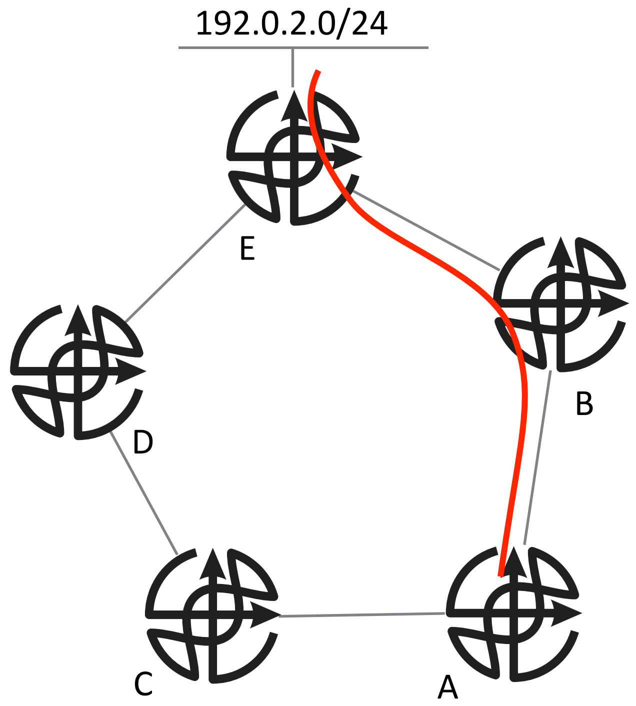
Data Flow Example



- What questions does this sort of analysis ask/answer?
- State
  - What state do I need to make each of these steps happen?
- Optimization
  - What's the tradeoff between the amount of state and optimal traffic flow?

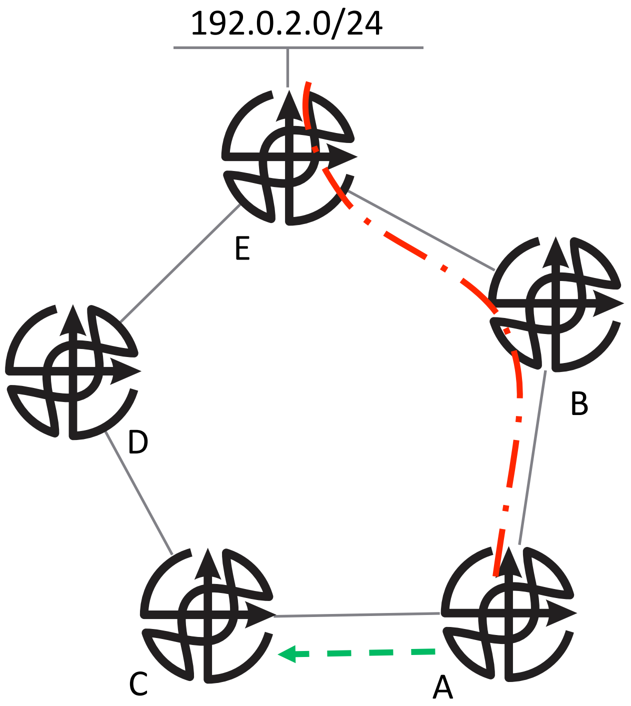
Control Plane Example

- Router A uses the path through B as its primary path to 192.0.2.0/24
- There is a path through C, but this path is blocked by the control plane
  - If A forwards traffic towards 192.0.2.0/24 to C, there is at least some chance that traffic will be reflected back to A, forming a routing loop
- We would like to be able to use C as an alternate path in the case of a link failure along A->B->E



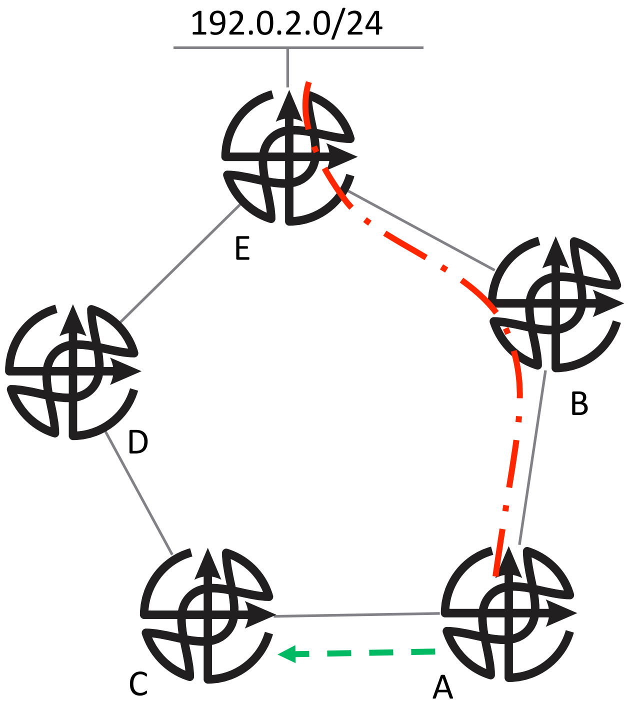
Control Plane Example

- Loop Free Alternates (LFAs)
  - A can compute the cost from C to determine if traffic forwarded to 192.0.2.0/24 will, in fact, be looped back to A
  - If not, then A can install the path through C as a backup path



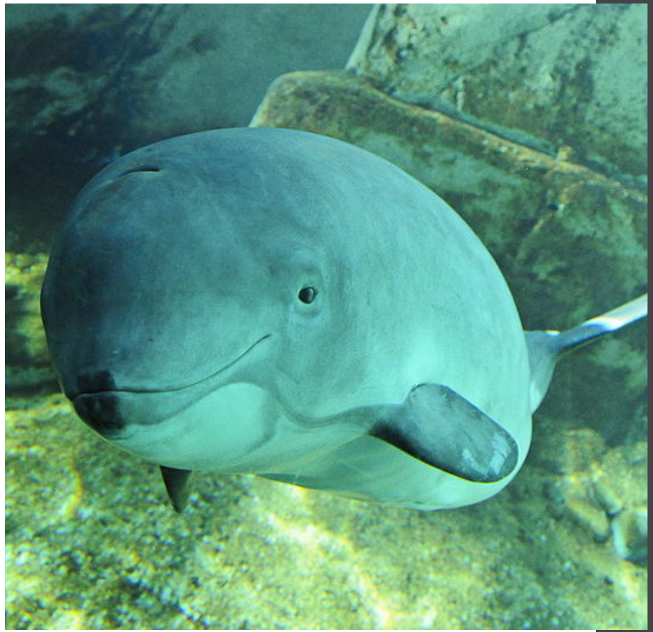
Control Plane Example

- What kinds of questions can we ask about this example?
- State
  - What state do I need to implement this?
  - How much state will be added?
  - Will this cause state to change more quickly or less?
- Optimization
  - What's the tradeoff between the amount of state and optimal traffic flow?



Developing the Metamodel

- What questions are common?
  - Questions about *state*
  - Questions about *surfaces*
  - Questions about *optimization*
- What other questions are important?
  - Why? What problem am I solving here?
  - Are there problems and/or solutions similar to this one?



The "Why" Question

- Acts as an abstractor
  - Only look at the stuff you need to understand the system
- Closes off rabbit trails
  - "Why does IS-IS elect a DIS?"
  - "That's a good question, but it's not related to fixing these slow DNS queries…"
- Connects the technical to the business
  - Unless you can explain why this problem is important to solve…
  - It's probably not!

What is this like?

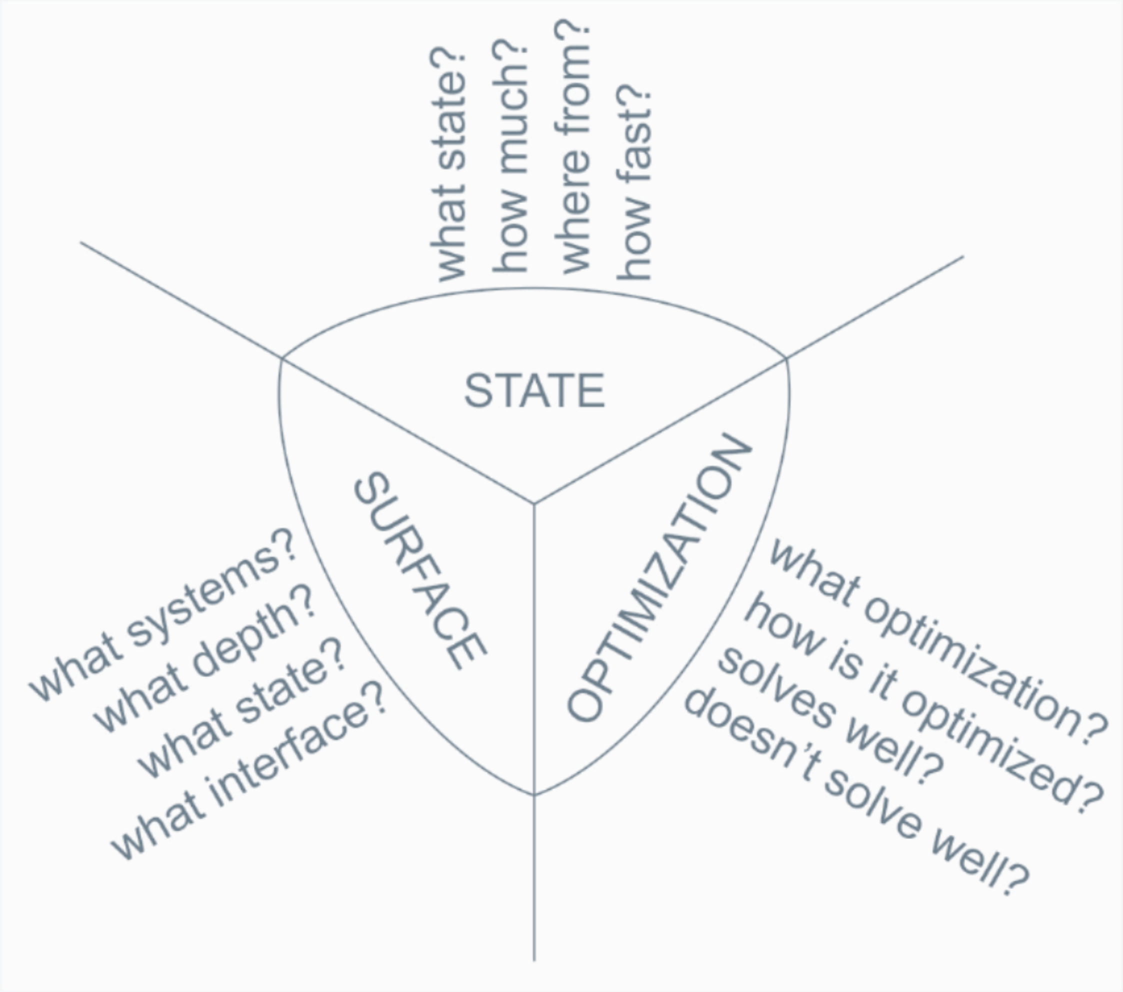
Common Problem	Common Solution	Common Problems with the Solution
Find the shortest path	SPF	Microloops, state too fast
	Path Vector	Slow convergence, inconsistent state
	Bellman-Ford	Slow convergence, feedback loops
	DUAL	Drops during convergence, feedback loops
Distribute Data	Flood	Possibly too dense, CAP, transmission errors
	Multicast	Complex control plane, CAP, transmission errors
	Unicast	Complex control plane, CAP, transmission errors
Reliable Data Transmission	Forward Correction	Carry unneeded state in many cases
	Detect/Retransmit	Slows down transmission

Why?

What problem am I solving?

Narrows my view – don't boil the ocean

What & How?



This is like?

Has this problem been solved before?

How was it solved?

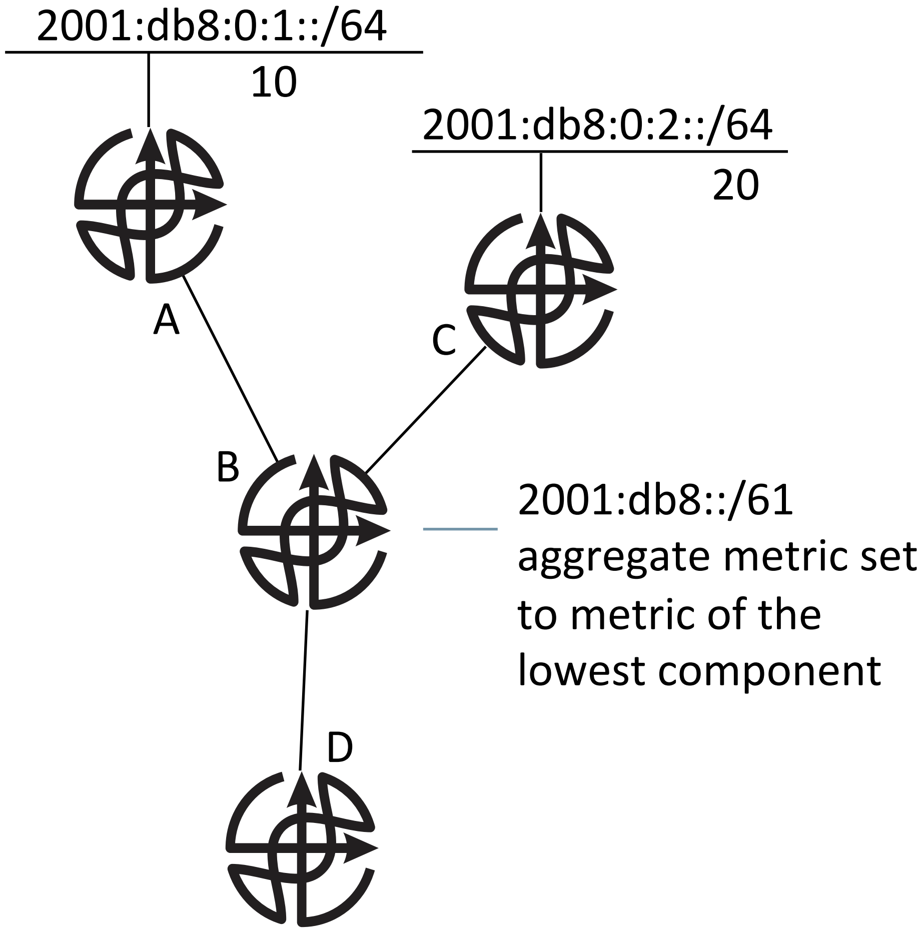
What are the problems with this solution?

Metamodel
to understand
a complex system quickly

Build an abstract model by asking the right questions

Models and Abstractions

- Abstractions Leak
- Aggregation
  - If the A->B link fails, the cost of the aggregate changes
- TCP retransmissions
- Embedded IP addresses



2001:db8::/61 aggregate metric set to metric of the lowest component

Models and Abstractions

- Abstractions hide things
  - You might not understand the entire system as well as you might by studying it “in detail”
  - But comprehending a complex system without abstractions often just leads to confusion
- More than one model will help expose different aspects of the system

Why?

State/Surface/Optimization

What?

How?

What is this like?

to find potential solutions and "know where to look"