The Hidden Nature of Complexity and the Software Defined Network Operator of the (very near) Future

$$\int_0^\infty \ln|S(i\omega)|d\omega = \int_0^\infty \ln\left|\frac{1}{1+L(i\omega)}\right| d\omega = \pi \sum Re(p_k) - \frac{\pi}{2}\lim_{s\to\infty} sL(s)$$

David Meyer
CTO and Chief Scientist, Brocade
Director, Advanced Technology Center, University of Oregon
Upperside NFV/SDN Summit 2014
dmm@{brocade.com,uoregon.edu,1-4-5.net,…}
http://www.1-4-5.net/~dmm/talks/hidden_nature_of_complexity.pdf

Agenda

- Too many words, too many slides ☺
  - This talk is about thinking about networking (and SDN) in new ways
- A Couple of Macro Trends
- SDN Context: Problem Space and Hypothesis
- Complexity, Layered Architectures, and SDN
- A Perhaps Controversial View
- Summary and Q&A if we have time

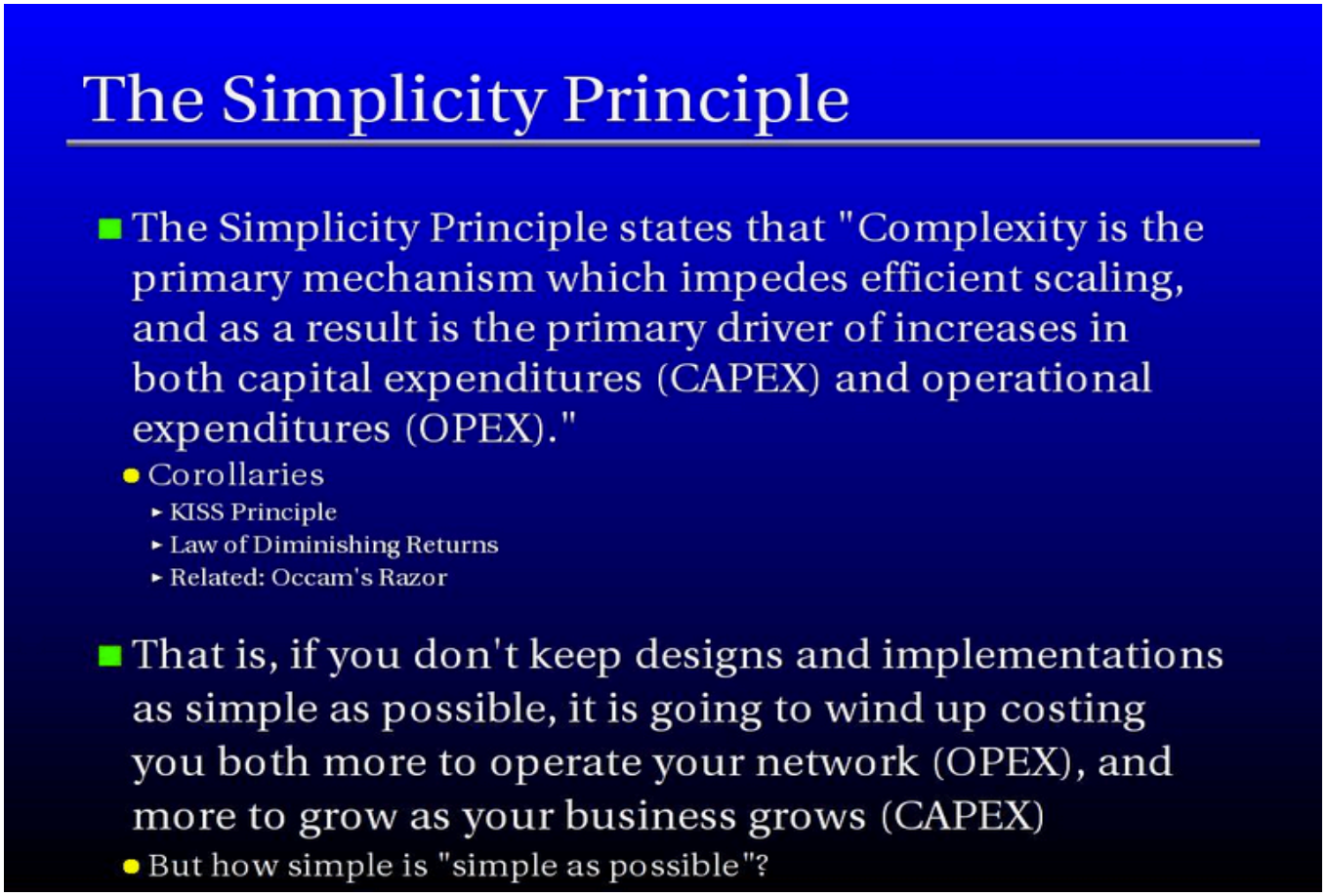
Danger Will Robinson!!!

Standard Disclaimer

This talk might be controversial/provocative (and perhaps a bit "sciencey")

A Long Time Ago in a Galaxy Far, Far Away…

(a first cut at trying to understand network complexity)



Mike O'Dell, by all accounts. See http://www.1-4-5.net/~dmm/talks/NANOG26/complexity_panel/

Sources of Complexity?
(note: not quite right…)

Sources of Complexity

- There are two well-known principles from non-linear systems theory that apply here, namely:
- Amplification
- Coupling

Amplification

Amplification Principle

- The Amplification Principle states that there are non-linearities which occur at large scale (which are not observed at smaller scales).
- COROLLARY: In many large networks, even small things can and do cause huge events.
  - In system-theoretic terms, in large systems such as these, even small perturbations on the input to a process can destabilize the system's output.
- Takeaway: As your networks grow, even things that once looked small can turn out to be big!

Coupling

Coupling Principle

- The Coupling Principle states that as things get larger, they often exhibit increased interdependence between components
- COROLLARY: The more events that simultaneously occur, the larger the likelihood that two or more will interact. This phenomenon has also been termed "unforeseen feature interaction" [Doyle and Willinger]
- Coupling can be either horizontal (within a protocol layer) or vertical (between layers)

Premise/Goal/Context of this Talk

- Clearly systems approaches to biology, medicine, engineering, and neuroscience face converging challenges

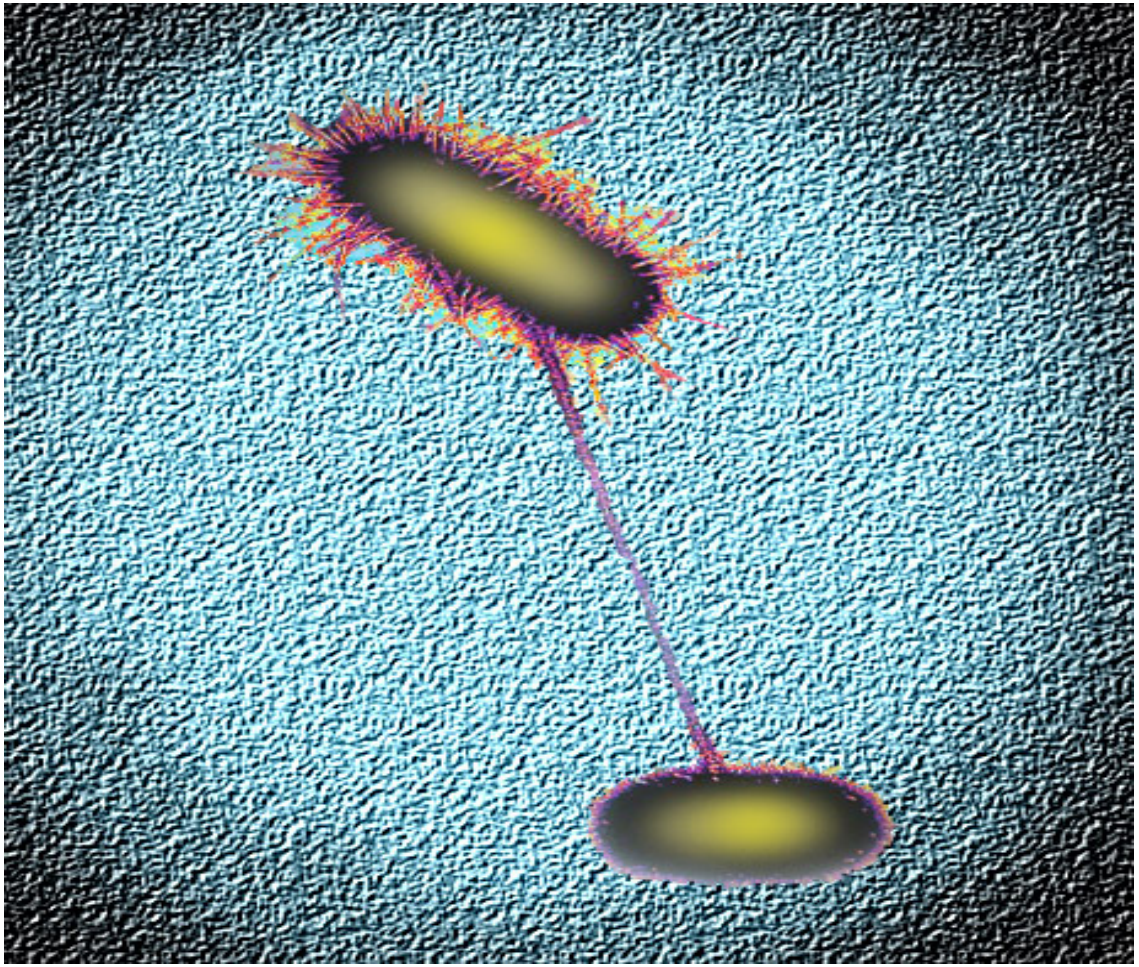
- Why?
  - Because modern science, technology, and culture create dauntingly complex but similar and overlapping problems in these domains → Convergent Evolution (a topic in and of itself)
  - SDN has given us a new tool with which to understand/experiment with network architectures

- Goal: To convince you that we are at a point in the history of network technology at which integrated theories (and methods) applicable to all complex networked systems, including the Internet/SDN, are needed, and that the approach we should take is to concentrate on the organizational principles of complex systems.

- High Level Ideas/Hard Problems
  - Provably hard tradeoffs
    - Initially Speed vs. Generality
  - Layering
    - Layering is a candidate *universal architecture* that seems to give us the ability engineer the speed/flexibility tradeoff
  - Horizontal Transfer (H*T)
    - HGT, HAT
  - BTW, why are there (necessarily) hard tradeoffs?
    - Top down requirements coupled with bottom up constraints (HW) → hard tradeoffs
    - Turing , Shannon and Bode, …

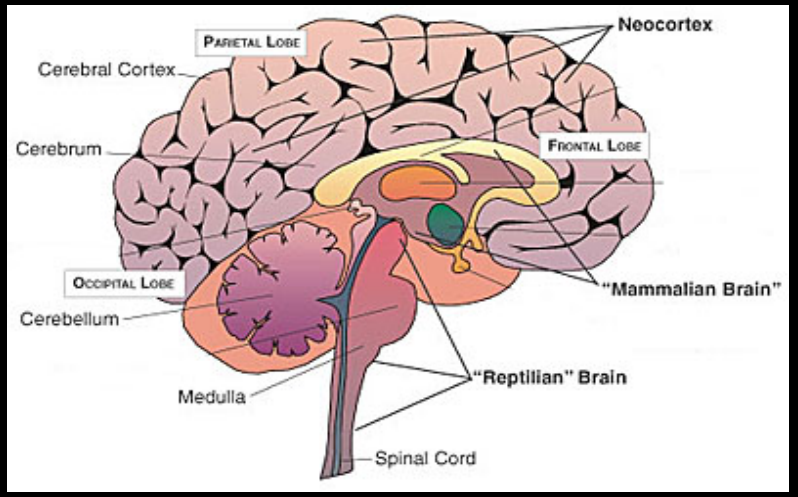
- And how is all of this connected to SDN and the Internet?

A Couple of Macro Trends



Trend: The Evolution of Intelligence

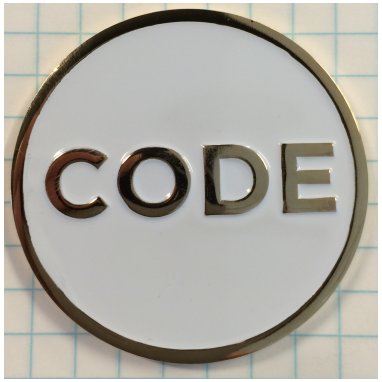
R-complex (Reptilian Brain) to Neocortex → Hardware to Software

HARDWARE	SOFTWARE
	

- Key Architectural Features of Scalable/Evolvable Systems
  - Turing, Bode, and Shannon
  - **RYF-Complexity (behavior)**
  - **Layered Architecture**
  - **Bowties and Hourglasses**
  - **Horizontal Transfer (H*T)**
  - Protocol Based Architectures

Once you have HW
its all about code…

Trend: Open Source (hardware *and* software)



- ***Community building*** **is a core Open Source objective**
- ***Code*** **is the coin of the realm**
- ***Engineering systems*** **are as important as artifacts**

Putting this all together →

Trend: Engineering artifacts are *no longer* the source of sustainable advantage and/or innovation

What you build isn't as important as how you build it

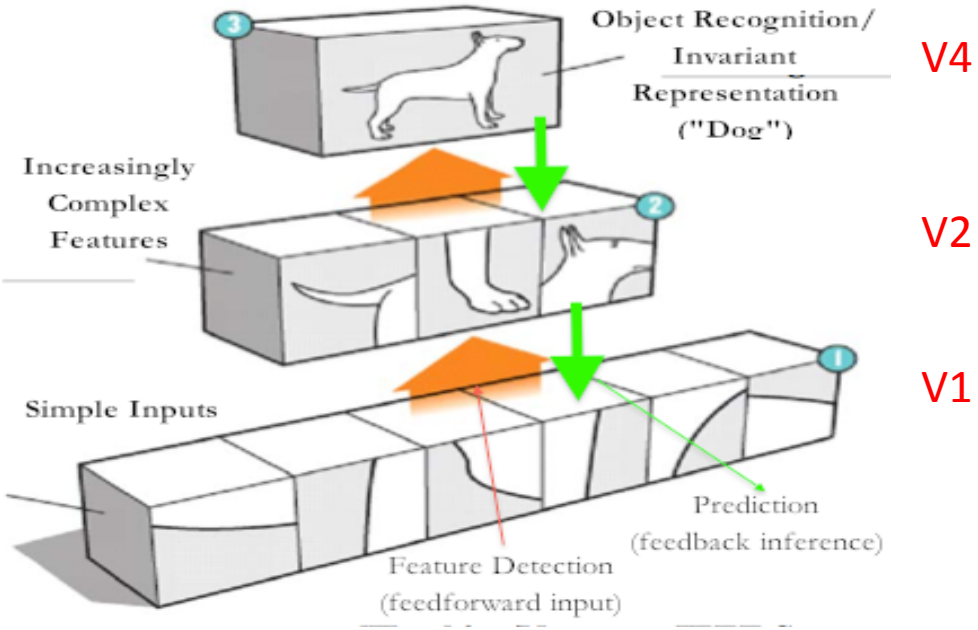
Perhaps surprisingly, the "hyper-scale" and open source communities have taught us that actual artifacts (in our case network applications as well as HW/SW) are ephemeral entities and that the only source of sustainable advantage/innovation consists of

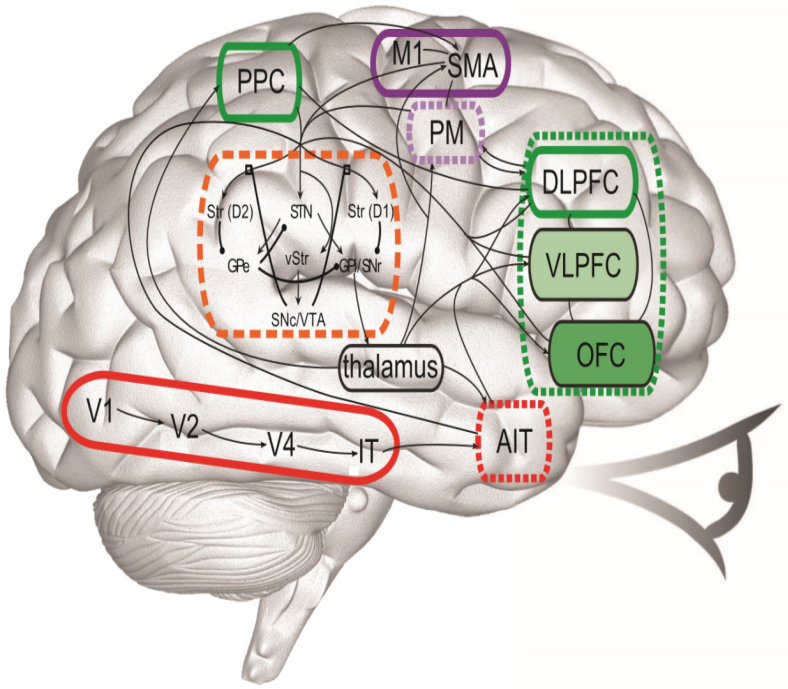
- Engineering Systems[1]
- Culture
- People/Process
- Multi-disciplinary Approaches

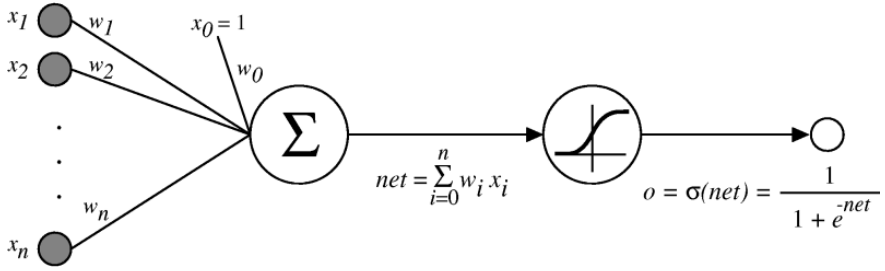
http://en.wikipedia.org/wiki/Aeroelasticity - Flutter

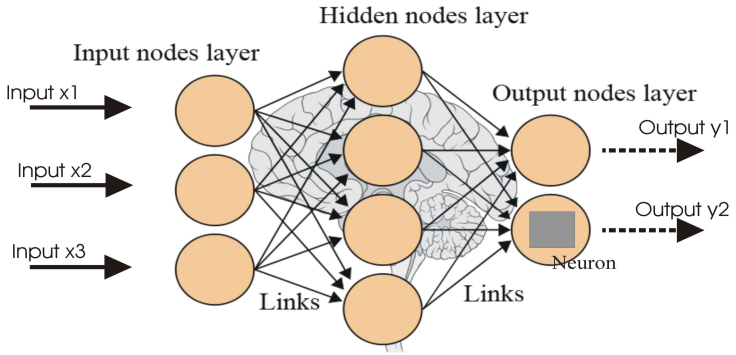
[1] Note that our *Engineering Systems* evolve using the same mechanisms that are used to build artifacts. This is architecturally analogous to Horizontal Gene Transfer (HGT) and the acquisition of anti-bacterial resistance in the bacteria biosphere; the same mechanisms used to create the artifact (plasmid) are used to evolve the "Engineering System" (transcriptional network). Consider: Horizontal Application Transfer?

Coming to a Network Near You…
Trend: Deep Learning









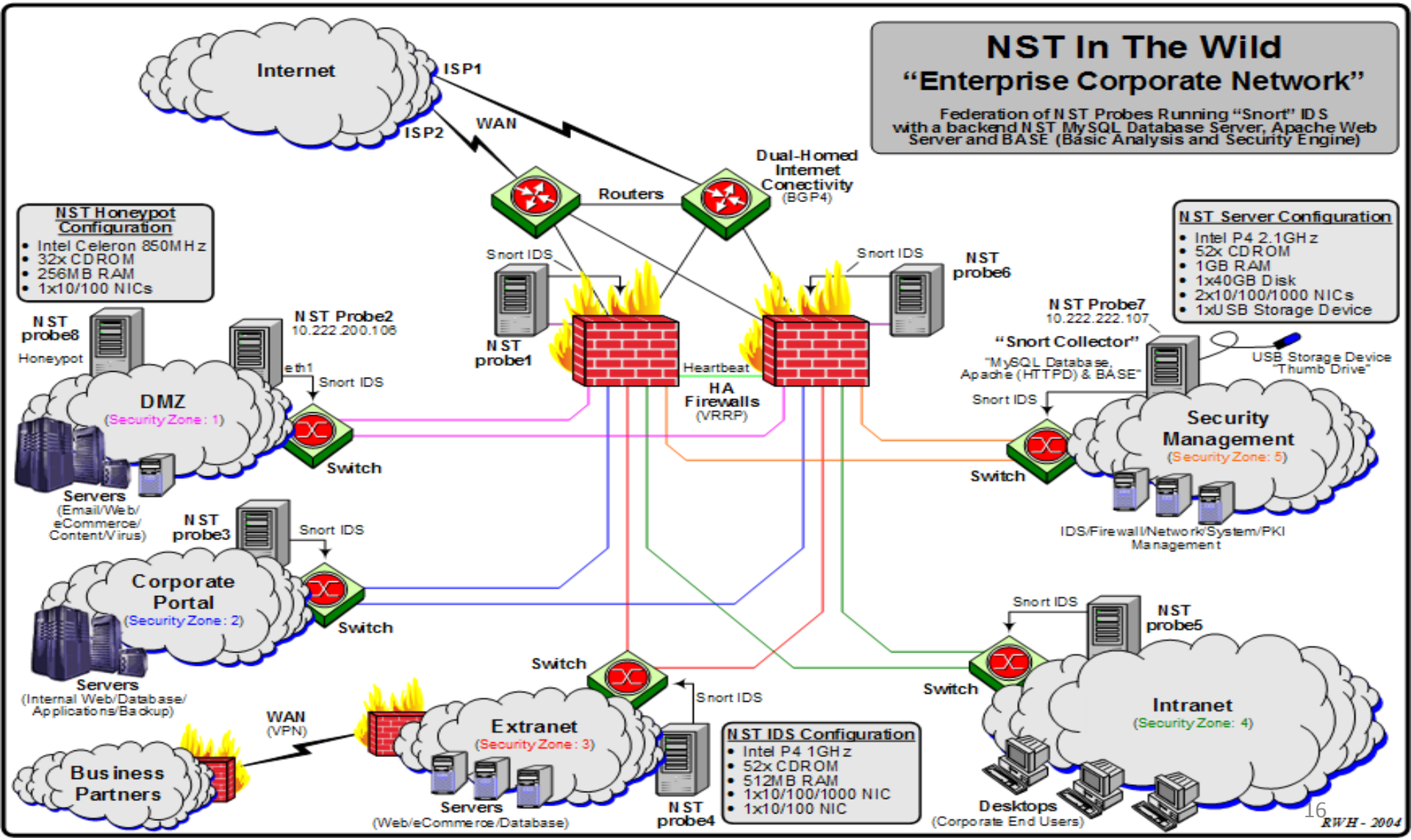
Agenda

- ~~A Couple of Macro Trends~~
- SDN Context: Problem Space and Hypothesis
- Complexity, Layered Architectures, and SDN
- A Perhaps Controversial View
- Summary and Q&A if we have time

Oh Yeah, This Talk Was Supposed To Have Something To Do With SDN

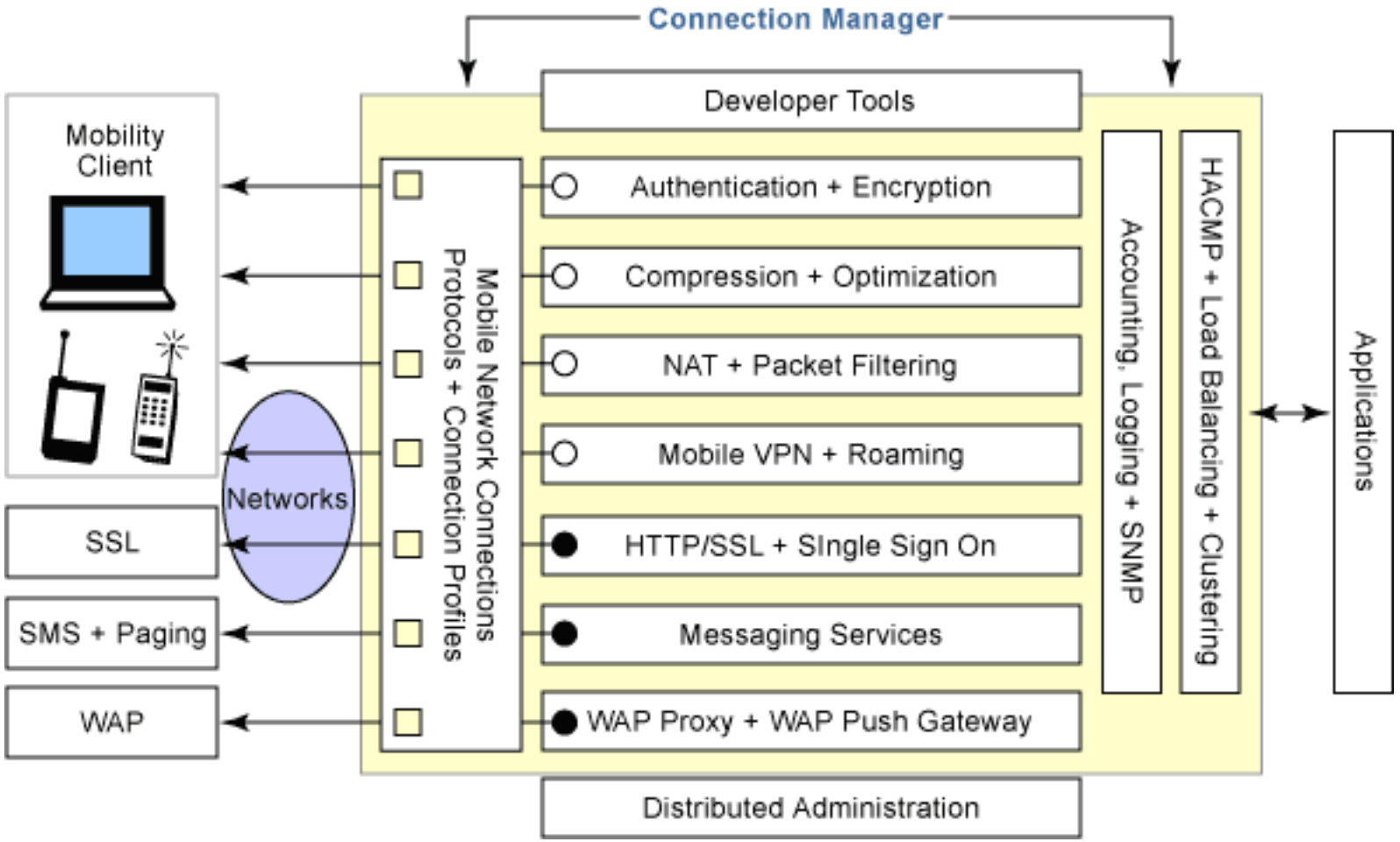
- Well then, what *was* the SDN problem space?
- Network architects, engineers and operators are being presented with the following challenge:
  - ***Provide state of the art evolvable network infrastructure and services while minimizing TCO***
  - → better, faster, cheaper, choose 3?
  - Evolvability and scalability? Horizontal Application Transfer?
- ***SDN Hypothesis***: It is *the lack of ability to innovate in the underlying network* coupled with the lack of proper network abstractions results in the inability to keep pace with user requirements and to keep TCO under control.
  - Is this true? Hold that question…
- *Note future uncertain*: Can't *"skate to where the puck is going to be"* because curve is unknowable (this is a consequence, as we will see, of the "software world" coupled with Moore's law and open-loop control).
  - That is, there is quite a bit of new research that suggests that such uncertainty is inevitable
- So given this hypothesis, what was the problem?

Maybe this is the problem?



Or This?

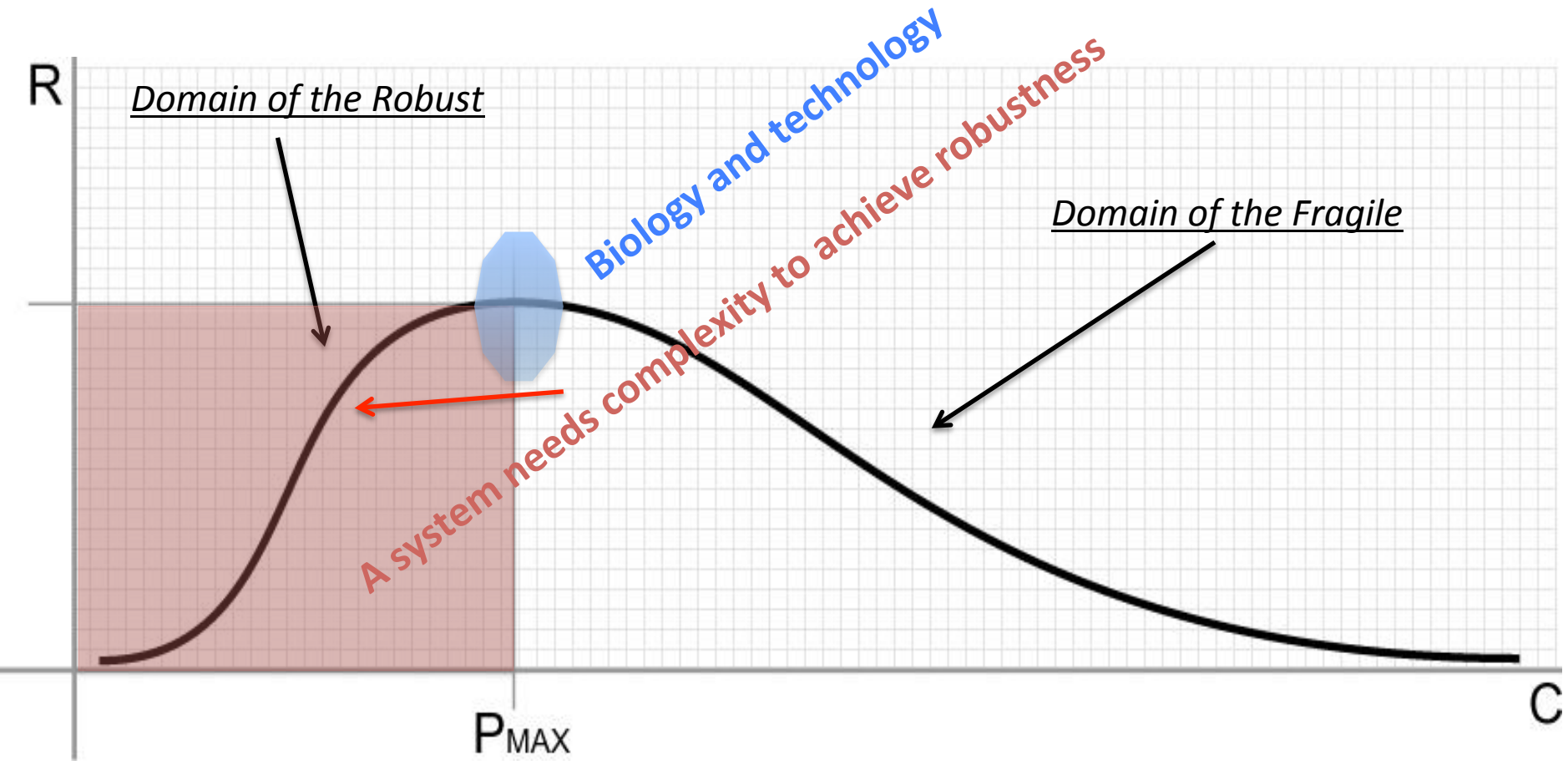
(Note Layering)



Many protocols, many touch points, few open interfaces or abstractions,..

Network is Robust *and* Fragile → The network is RYF-complex

BTW, Complexity Isn't Inherently "Bad"



Increasing number of policies, protocols, configurations and interactions (well, and code)

Agenda

- ~~A Couple of Macro Trends~~
- ~~SDN Context: Problem Space and Hypothesis~~
- Complexity, Layered Architectures, and SDN
- A Perhaps Controversial View
- Summary and Q&A if we have time

Connecting Complexity, Design, and Robustness

"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

Robustness is a Generalized System Feature

- **Scalability** is robustness to changes to the size and complexity of a system as a whole
- **Evolvability** is robustness of lineages to large changes on various (usually long) time scales
- Other system features cast as robustness
  - **Reliability** is robustness to component failures
  - **Efficiency** is robustness to resource scarcity
  - **Modularity** is robustness to component rearrangements
- Not surprisingly, these are the same features we're seeking from the network

Just so we're all talking about the same things – a few definitions

- **Robustness** is the preservation of a certain property in the presence of uncertainty in components or the environment
  - Systems Biology: Biological systems are robust if their important functions are insensitive to the *naturally occurring variations* in their parameters
    - Limits the number of designs that can actually work in the real environment
    - Examples: Negative autoregulation and exact adaptation in bacterial chemotaxis

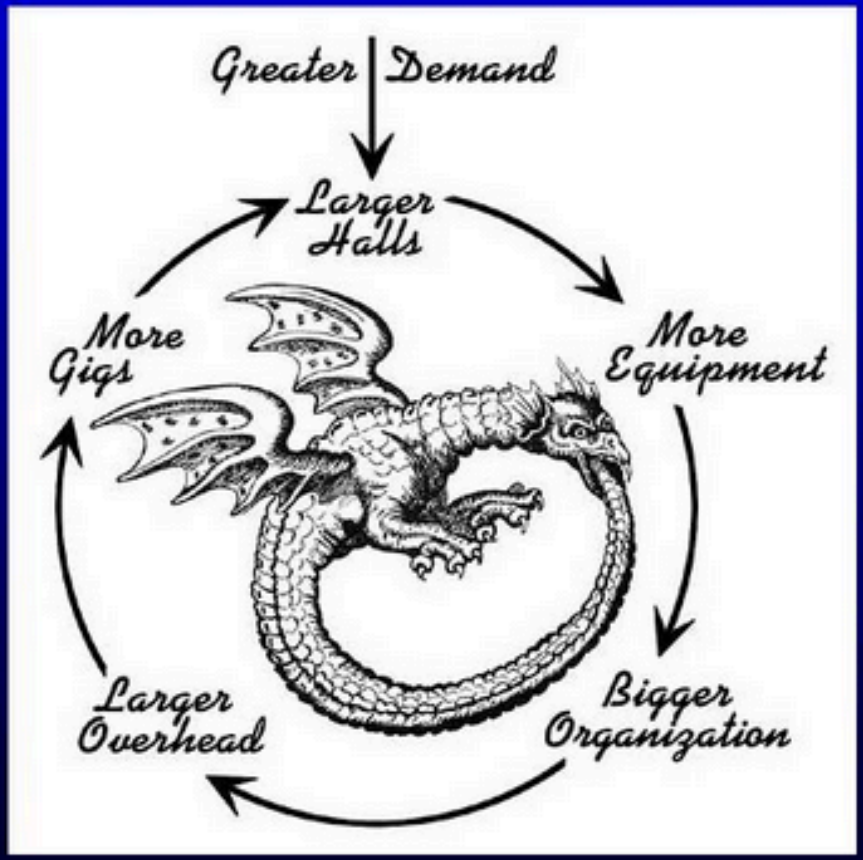
- **Fragility** is the opposite of robustness
  - Both need to be specified in terms of a system, a property and a set of perturbations

- A system can have a *property* that is *robust* to one set of perturbations and yet *fragile* for a *different property* and/or perturbation → the system is ***Robust Yet Fragile***
  - Or the system may collapse if it experiences perturbations above a certain threshold (K-fragile)

- For example, a possible ***RYF tradeoff*** is that a system with high efficiency (i.e., using minimal system resources) might be unreliable (i.e., fragile to component failure) or hard to evolve
  - Another example: HSRP (VRRP) provides robustness to failure of a router/interface, but introduces fragilities in the protocol/implementation
  - Complexity/Robustness Spirals

- **Summary**: Software, and SDN in particular, creates all kinds of RYF tradeoffs

A "Well Known" C/R Spiral



The creature is called an "Ouroboros" (which means "devouring its tail"). See http://www.dragon.org/chris/ouroboros.html (URL courtesy Sean Doran)

RYF Behavior is found everywhere

Robust	*Yet Fragile*
☺ Metabolism	☹ Obesity, diabetes
☺ Regeneration & repair	☹ Cancer
☺ Immune/inflammation	☹ AutoImmune/Inflame
☺ Microbe symbionts	☹ Parasites, infection
☺ Neuro-endocrine	☹ Addiction, psychosis,…
🗐 Complex societies	☠ Epidemics, war,…
🗐 Advanced technologies	💣 Disasters, global &!%$#
🗐 Risk “management”	💣 Obfuscate, amplify,…

Accident or necessity?

Robust

☺ Metabolism
☺ Regenerati
☺ Healing wo

Fragile

☹ Obesity, diabetes
une/Inflame

☹ Fat accumulation
☹ Insulin resistance
☹ Proliferation
☹ Inflammation

Same Mechanisms

- Fragility ← Hijacking, side effects, unintended…
  - DDOS, reflection, spoofing, …
- Of mechanisms evolved for robustness
- Complexity ← ***control***, robust/fragile tradeoffs
- Math: robust/fragile constraints (“conservation laws”)

Both

Accident or necessity?

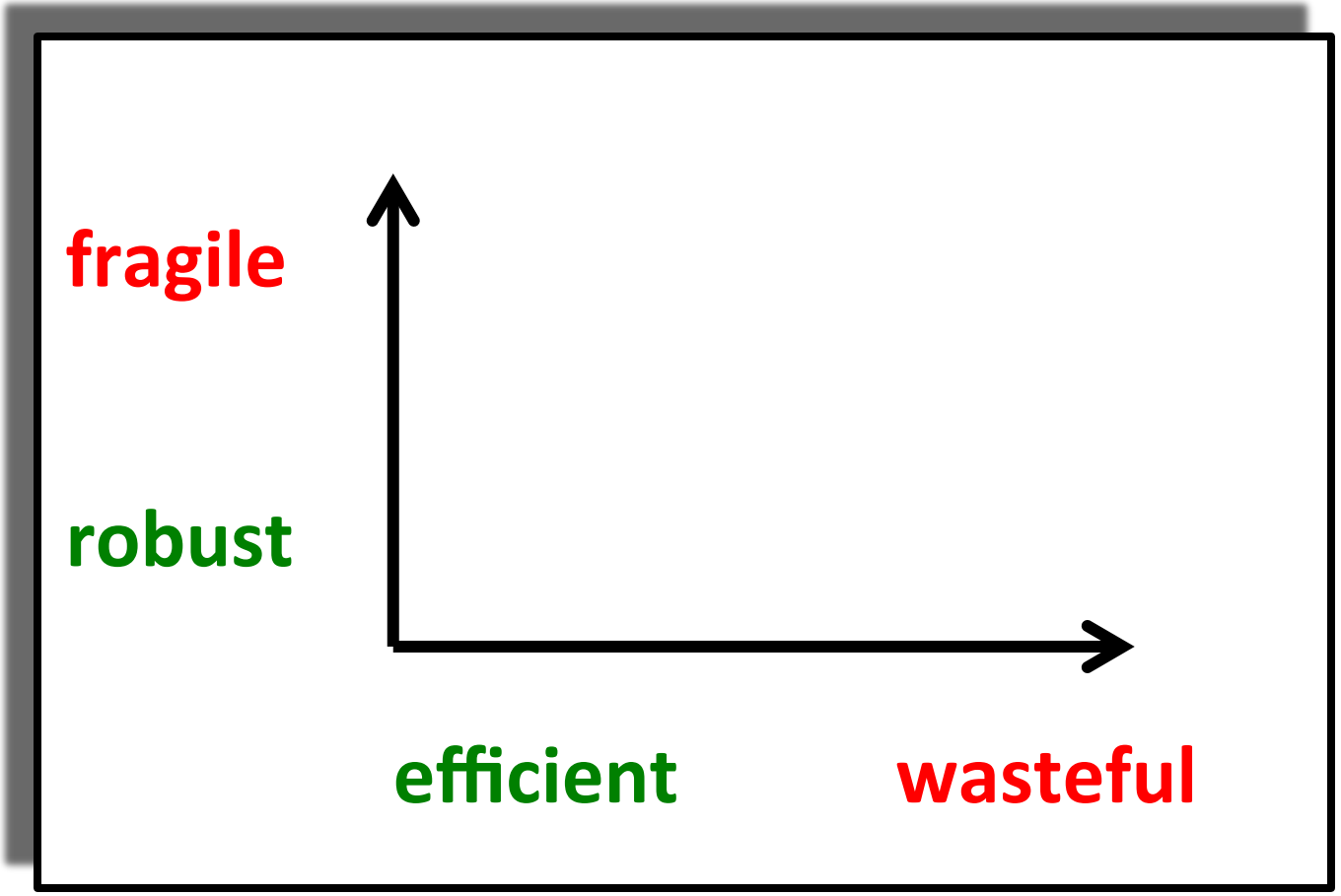
Original slide courtesy John Doyle

Summary: Understanding RYF is ***The*** Challenge

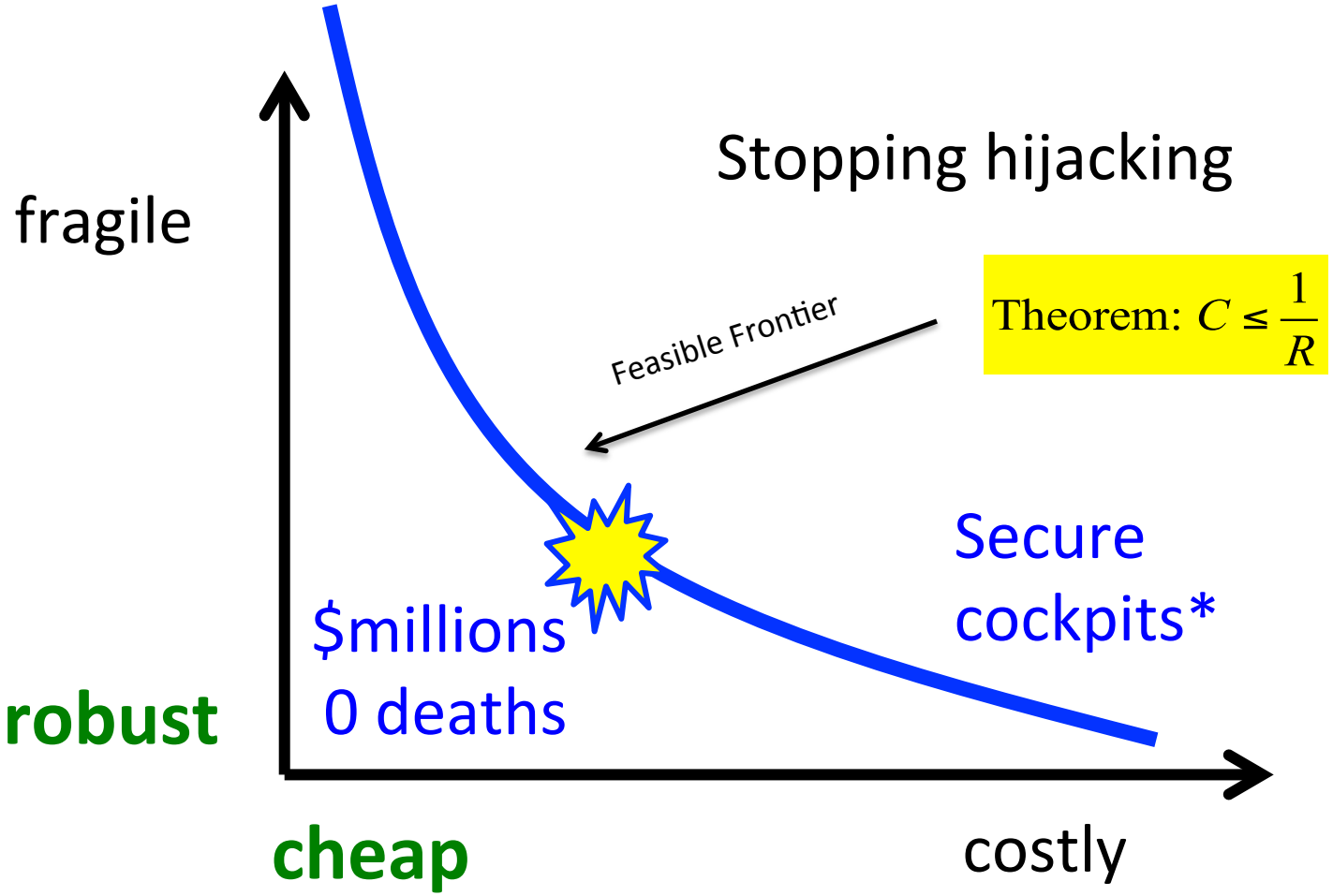
- It turns out that managing/understanding RYF behavior is ***the most essential challenge*** in technology, society, politics, ecosystems, medicine, etc. This means…
  - Understanding *Universal Architectural Principles*
  - Managing spiraling complexity/fragility
  - Not predicting what is likely or typical
    - But rather understanding what is catastrophic (fat tailed)
  - → ***understanding the hidden nature of complexity***

- BTW, it is much easier to create the robust features than it is to prevent the fragilities
  - With, as mentioned, poorly understood "conservation laws"

BTW, can we tell this story in a "Low Dimensional" space?



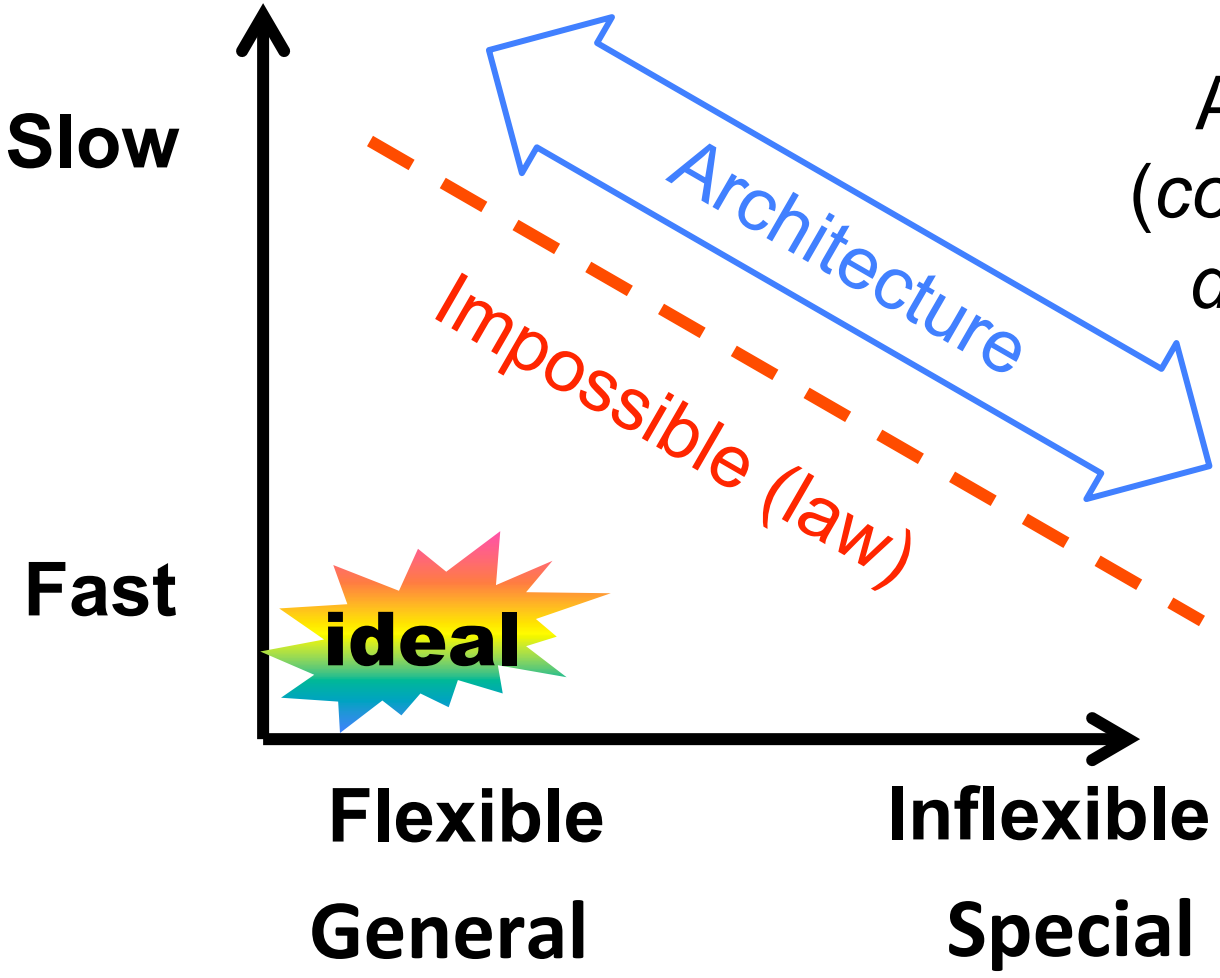
Example: Airline Security Architectures



* do cheap things engineers recommend

Universal Laws and Architectures (Turing)

Layering, Formal Systems, Hard Tradeoffs

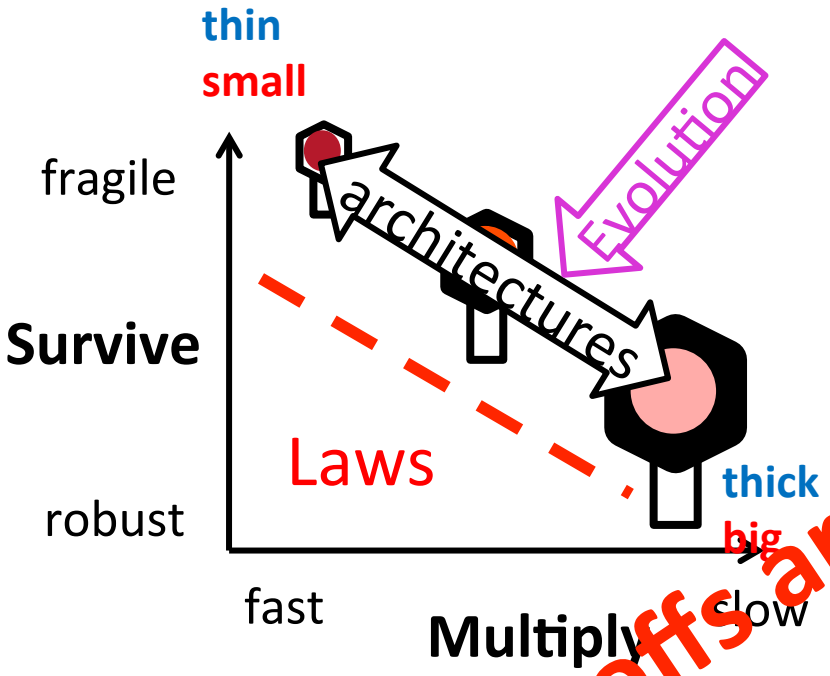


Architecture (*constraints that deconstrain*)

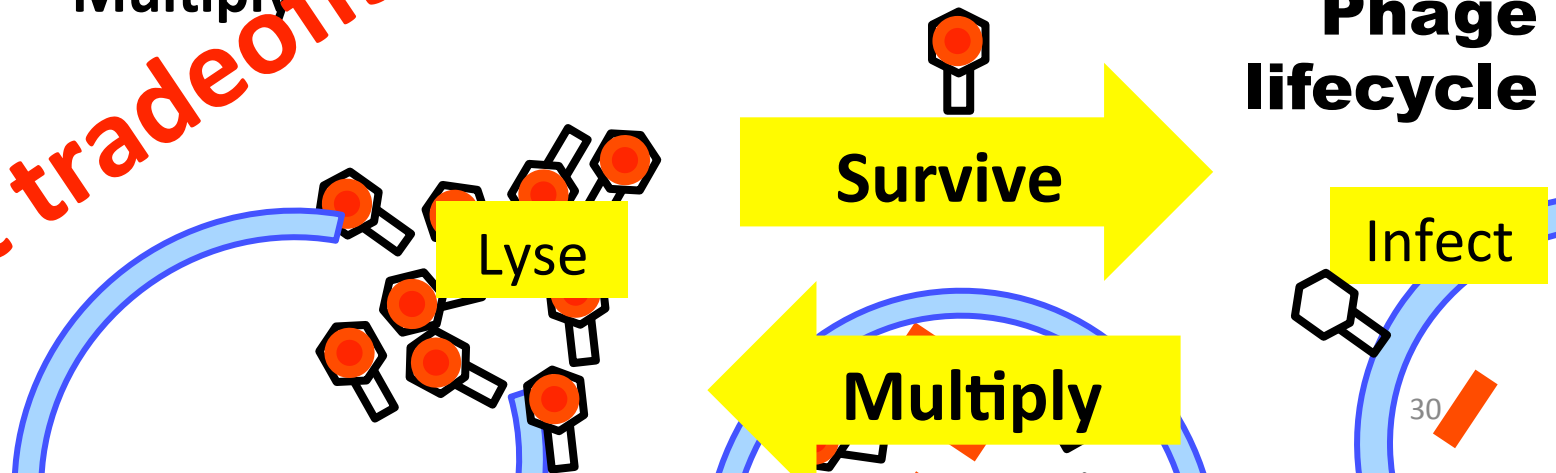
Original slide courtesy John Doyle

So What is Universal?

- *Laws, constraints, tradeoffs*
  - Robust/fragile
  - Efficient/wasteful
  - Fast/slow
  - Flexible/inflexible
- *Architecture*
- *Hijacking, parasitism, predation*



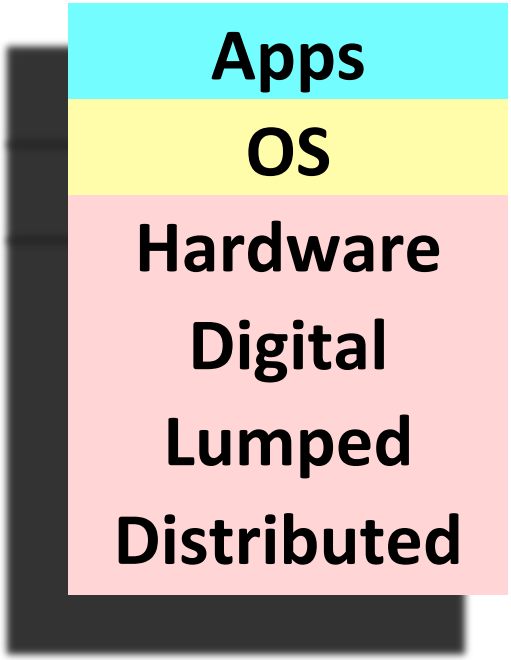
What tradeoffs are we making with SDN?

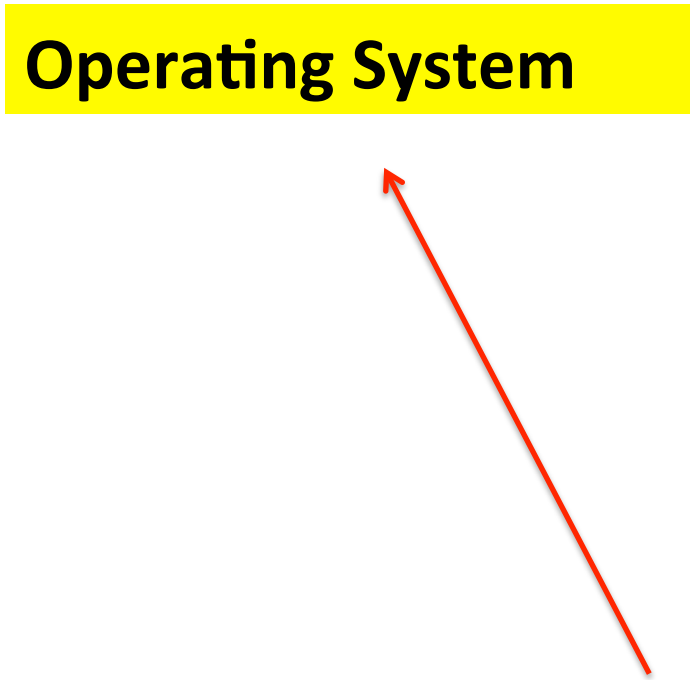


Architectures

- What we have learned is that there are ***universal architectural building blocks*** found in systems that *scale* and are *evolvable*. These include
  - **Layered Architectures**
  - **Bowties and Hourglasses**
  - **Horizontal Transfer (H*T)**
  - Protocol Based Architectures
  - Massively distributed with *robust* control loops

Layered architectures 101





Apps

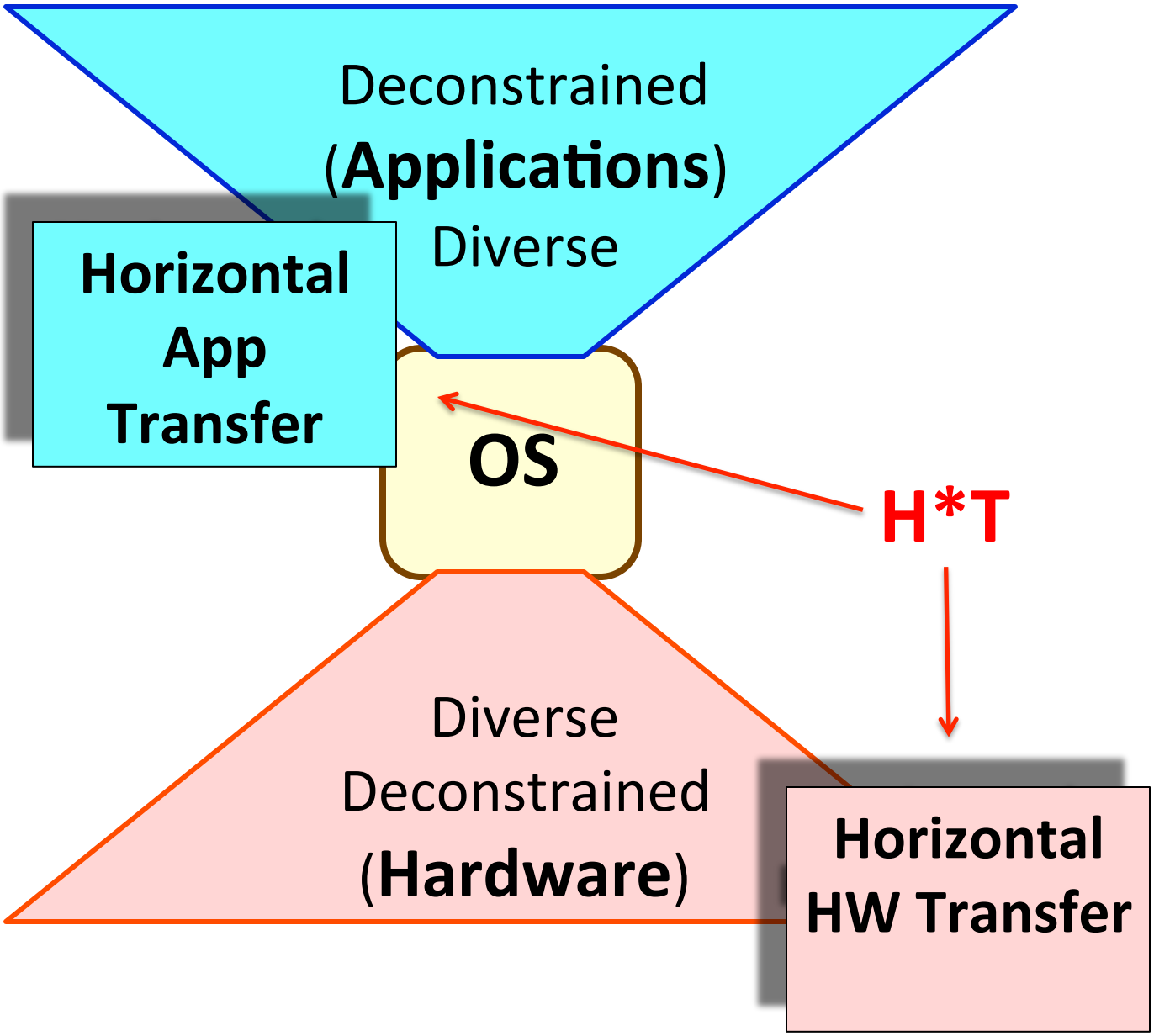
OS

HW

SDN Controller/Network OS?

Slide courtesy John Doyle

Layered architectures



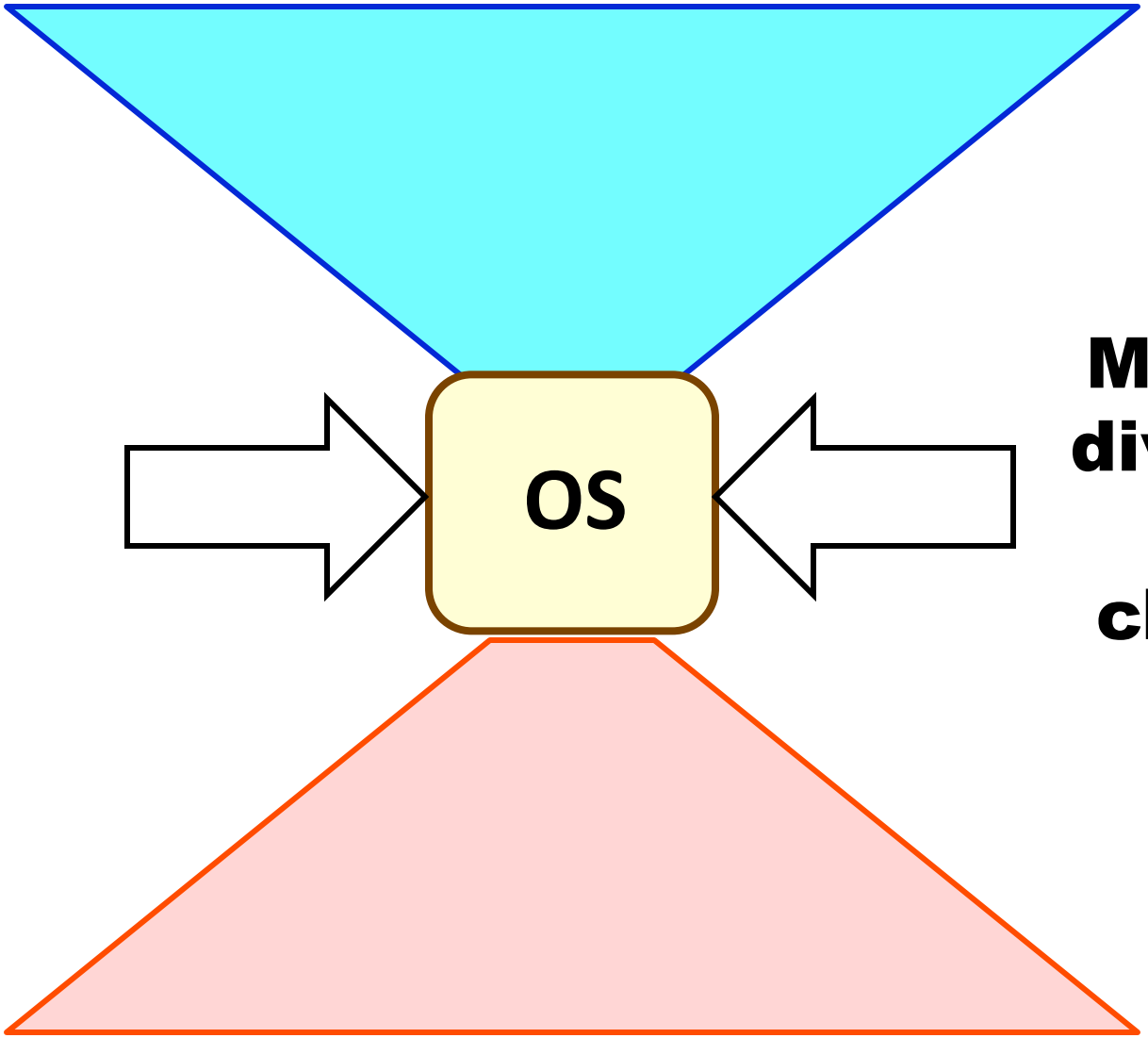
Apps

OS

HW

Slide courtesy John Doyle

Layered architectures



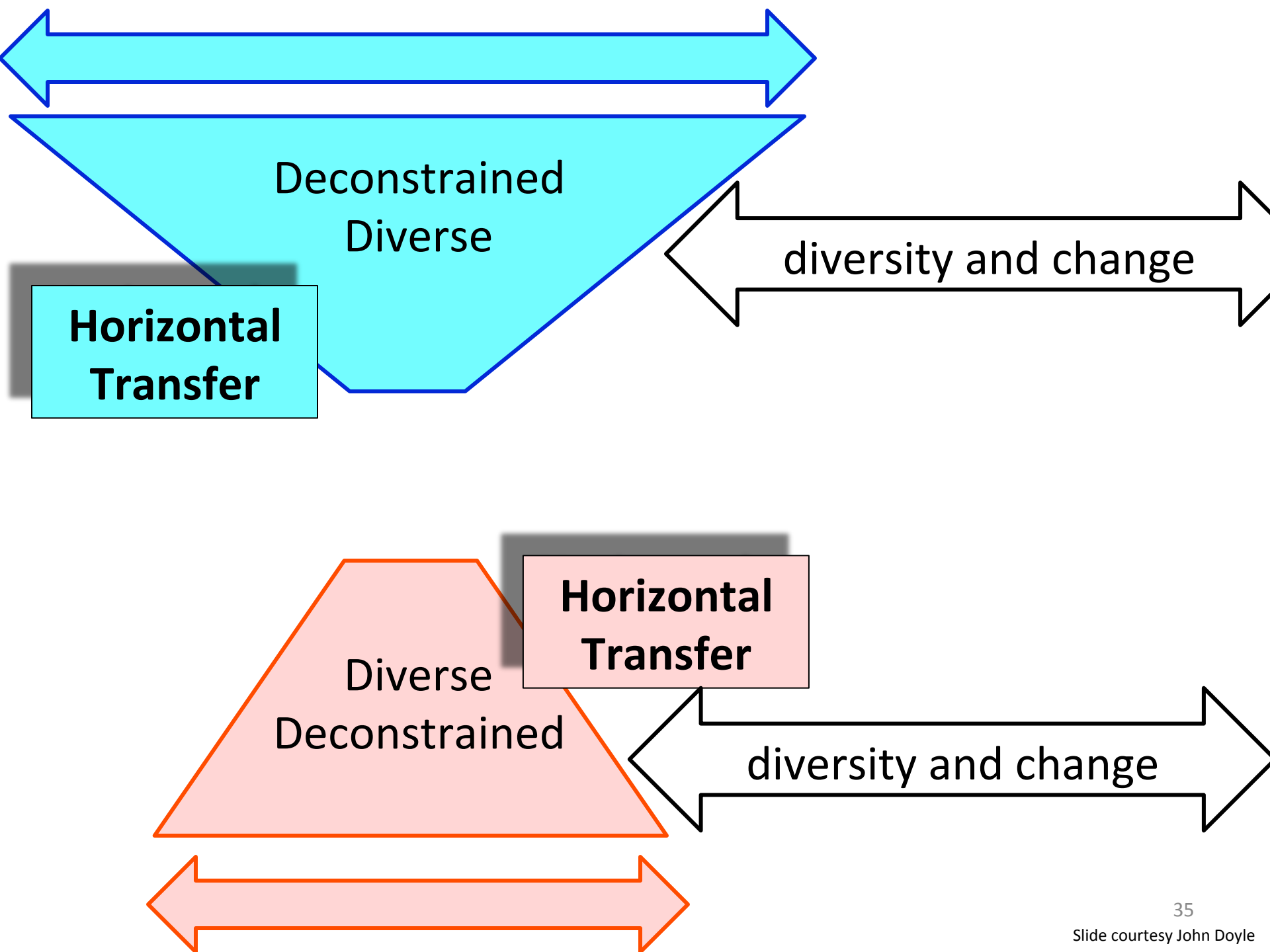
Minimal diversity and change

Apps

OS

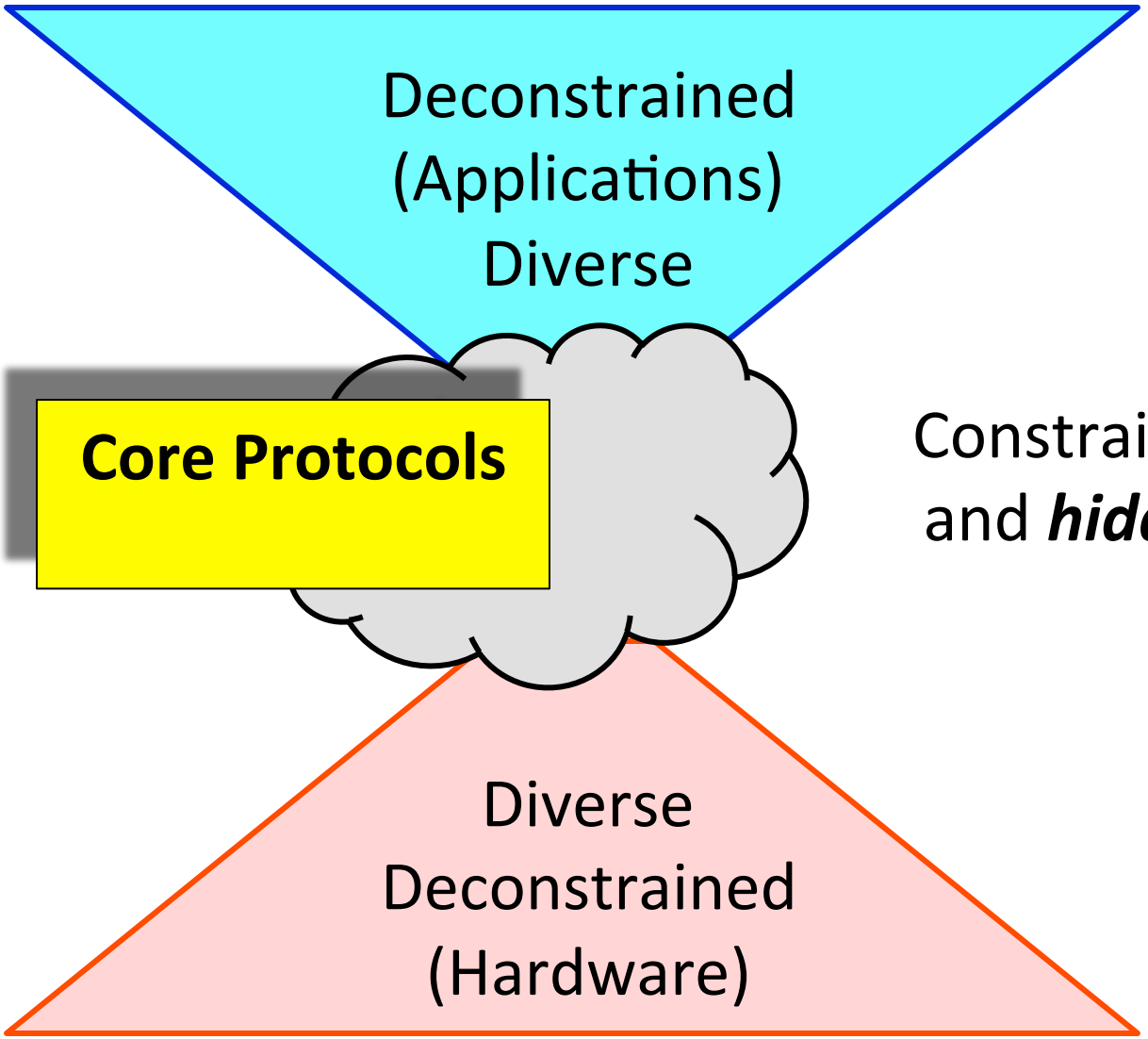
HW

Slide courtesy John Doyle



Slide courtesy John Doyle

Layered architectures



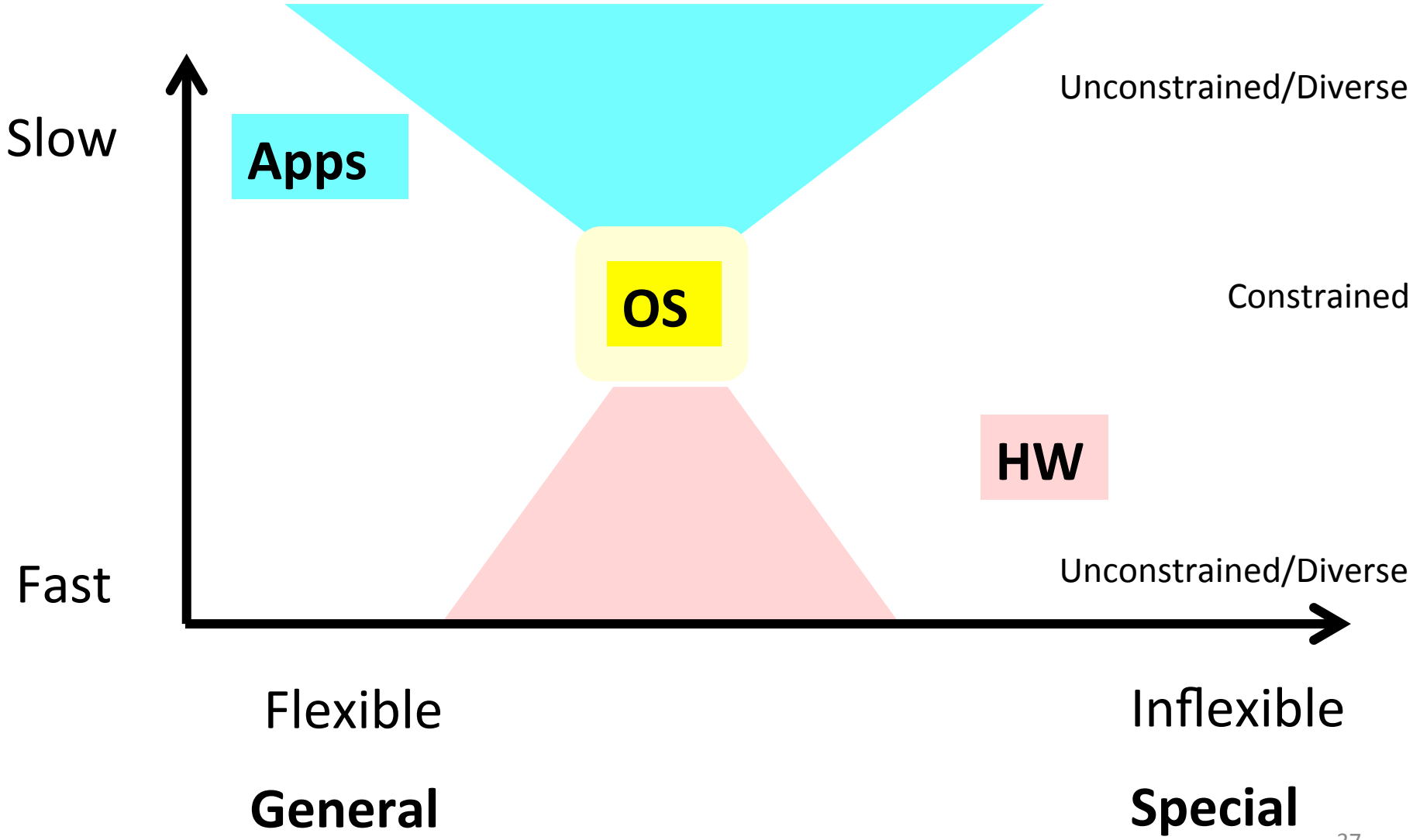
Apps

OS

HW

Slide courtesy John Doyle

Overlaying Tradeoffs



Slide courtesy John Doyle

Layering is a universal architecture

Layered Bacteria

Apps

Slow
Cheap

HGT

DNA repair

Mutation

DNA replication

Transcription

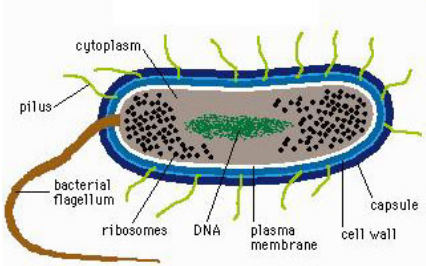
Translation

Metabolism

Signal…

OS

HW



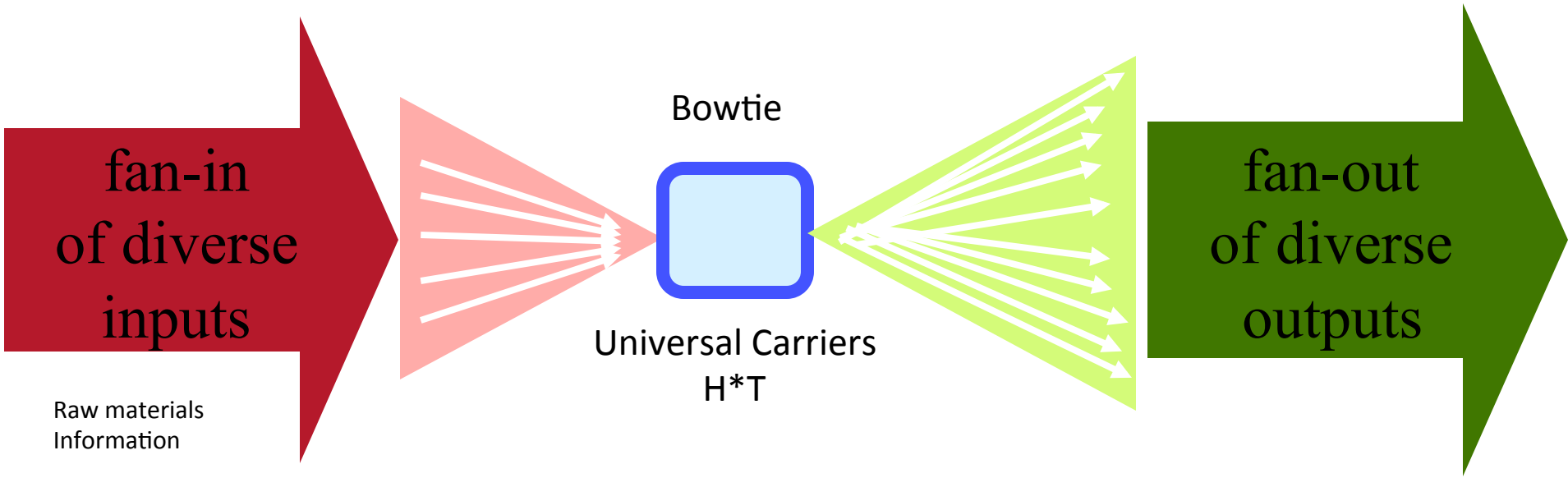
Fast
Costly

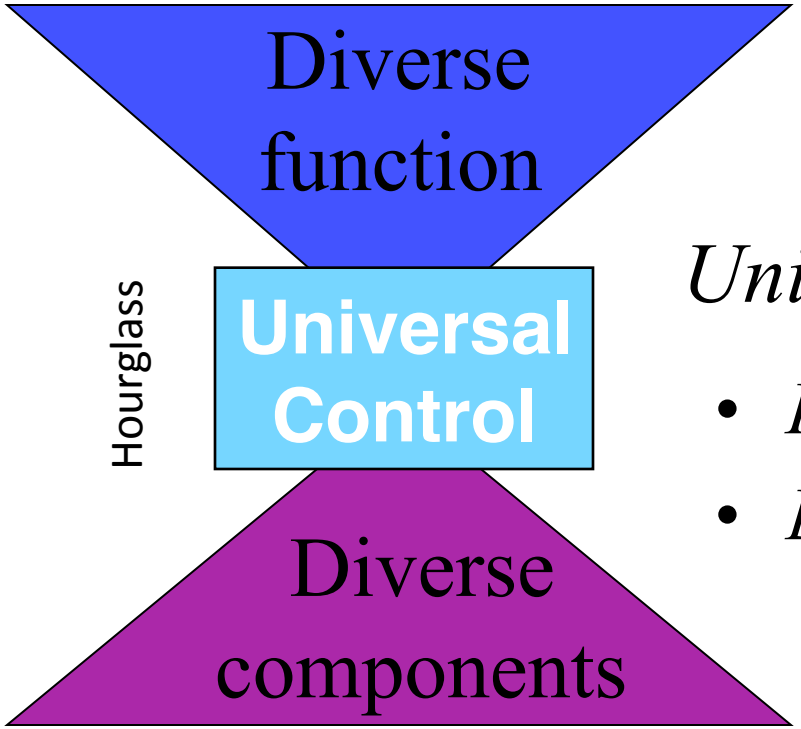
Flexible

General

Inflexible

Special





Universal Architectural Principles

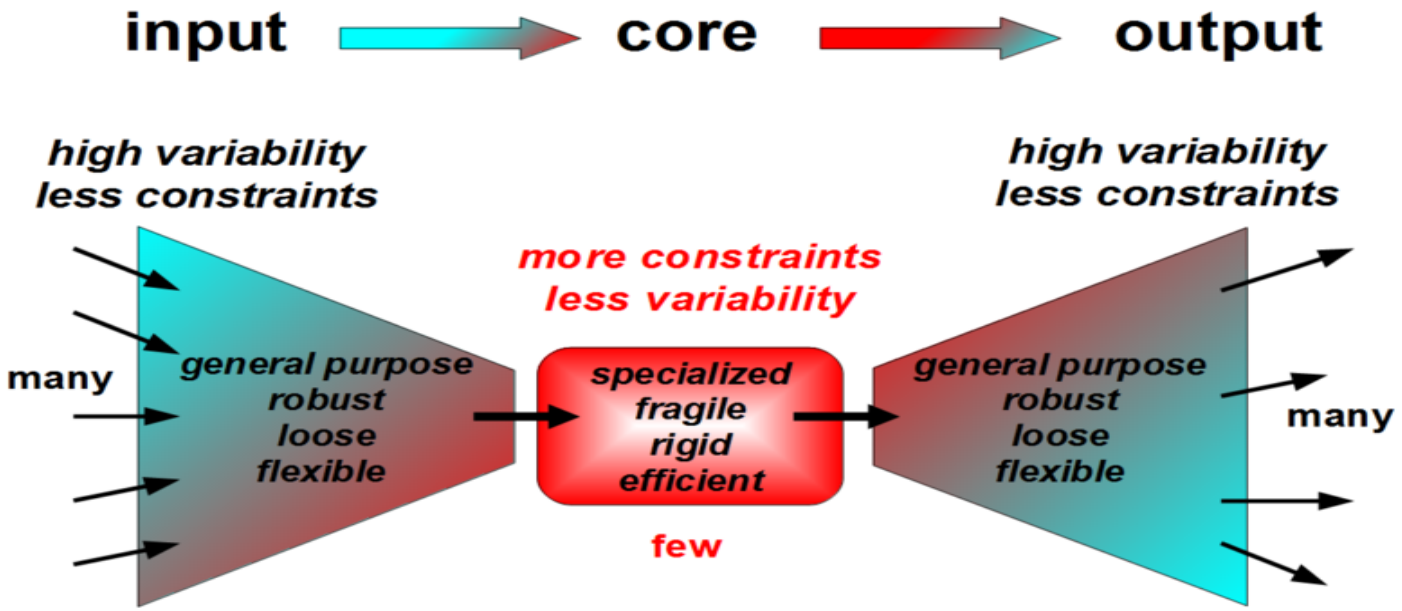
- *Hourglasses* for layering of control
- *Bowties* for flows within layers

Slide courtesy John Doyle

Bowties 101

Constraints that Deconstrain

Schematic of a "Layer"



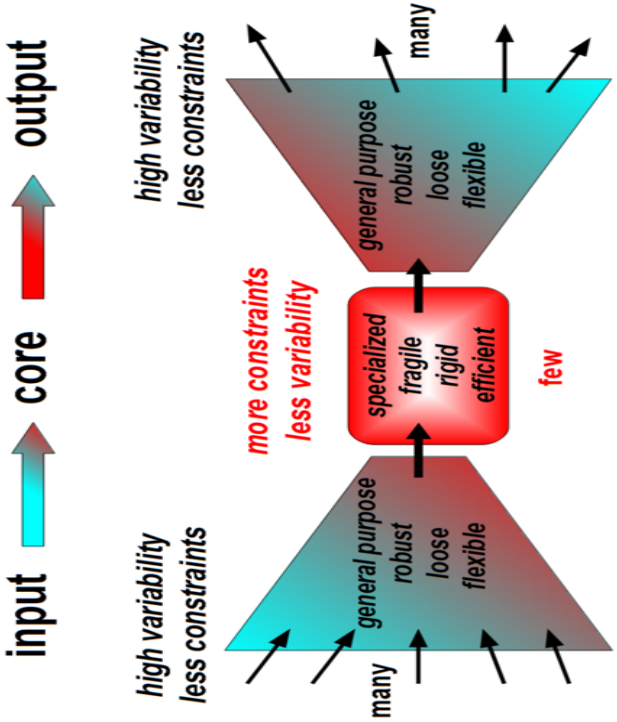
For example, the reactions and metabolites of core metabolism, e.g., *ATP metabolism*, Krebs/Citric Acid Cycle, … form a "metabolic knot". That is, ATP is a ***Universal Carrier*** for cellular energy.

1. Processes L-1 information and/or raw material flows into a "standardized" format (the L+1 abstraction)
2. Provides plug-and-play modularity for the layer above
3. Provides robustness but at the same time fragile to attacks against/using the standardized interface
4. H*T

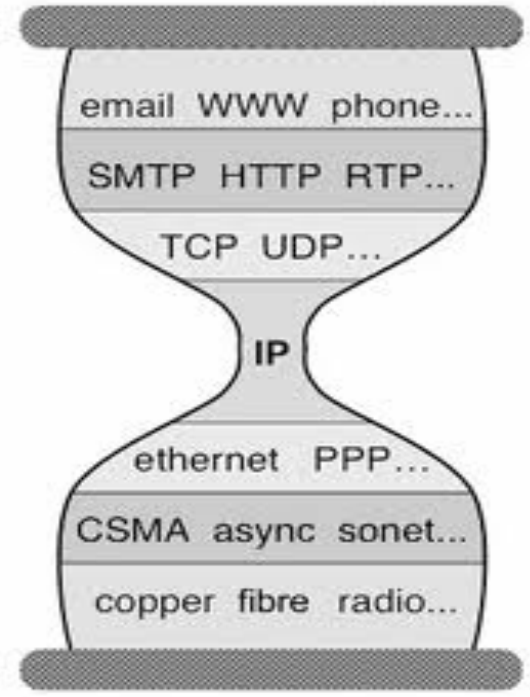
See Kirschner M., and Gerhart J., "Evolvability", Proc Natl Acad Sci USA , 95:8420–8427, 1998.

But Wait a Second

(Can we apply this to the Internet?)



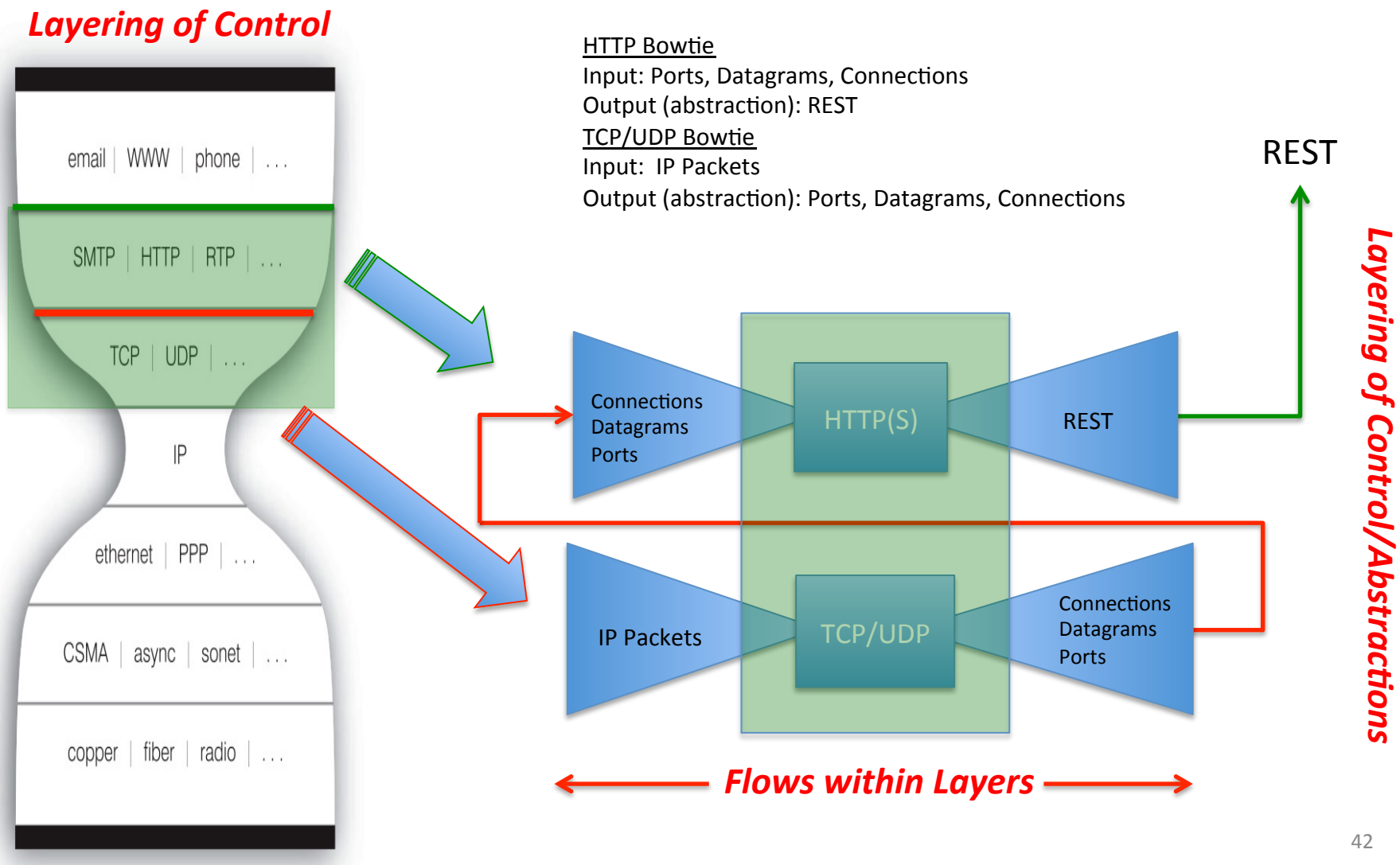
Bowtie Architecture



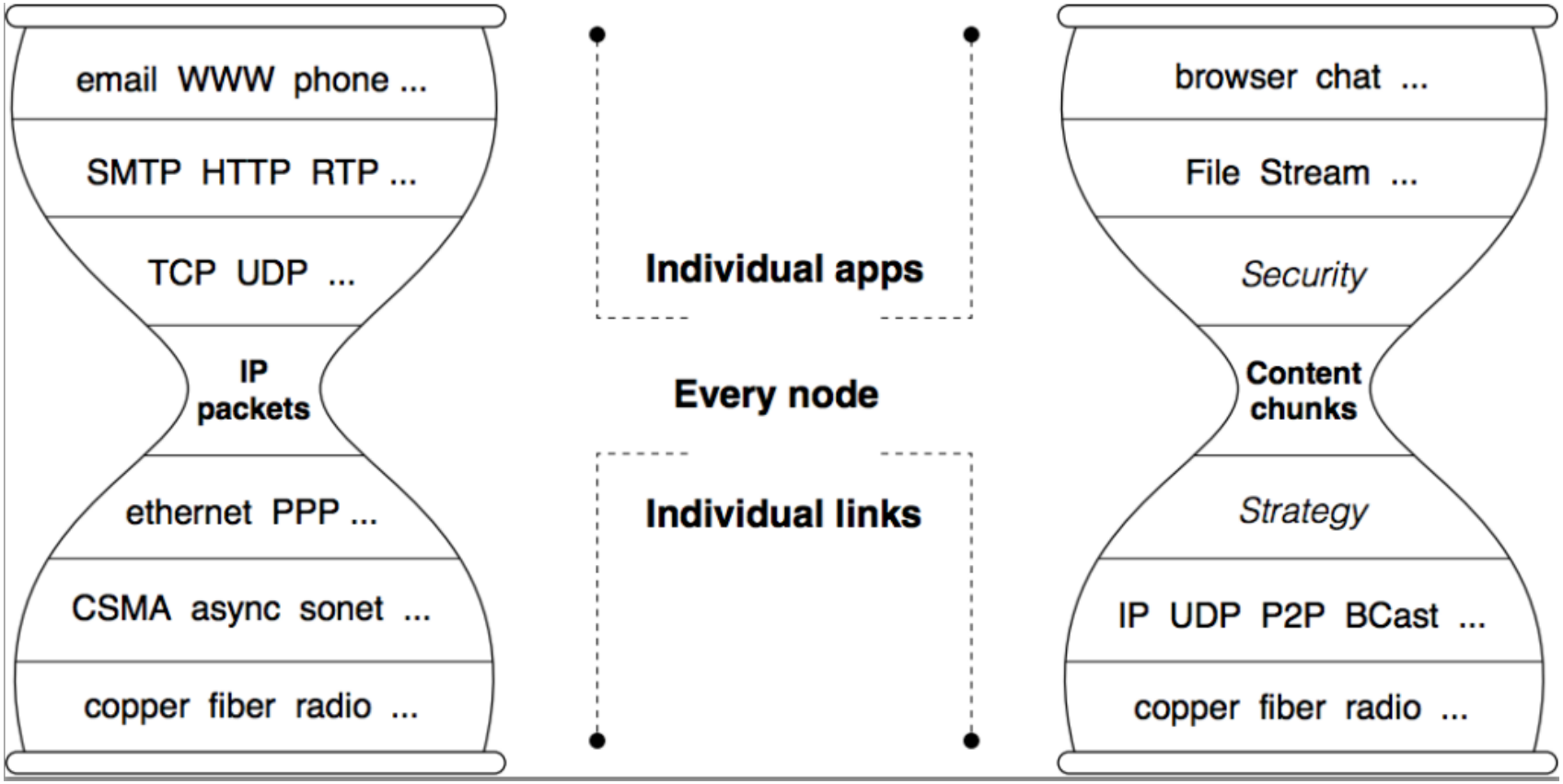
Hourglass Architecture

The *Protocol Hourglass* idea appears to have originated with Steve Deering. See Deering, S., "Watching the Waist of the Protocol Hourglass", IETF 51, 2001, http://www.iab.org/wp-content/IAB-uploads/2011/03/hourglass-london-ietf.pdf. See also Akhshabi, S. and C. Dovrolis, "The Evolution of Layered Protocol Stacks Leads to an Hourglass-Shaped Architecture", http://conferences.sigcomm.org/sigcomm/2011/papers/sigcomm/p206.pdf.

The Nested Bowtie/Hourglass Architecture of the Internet

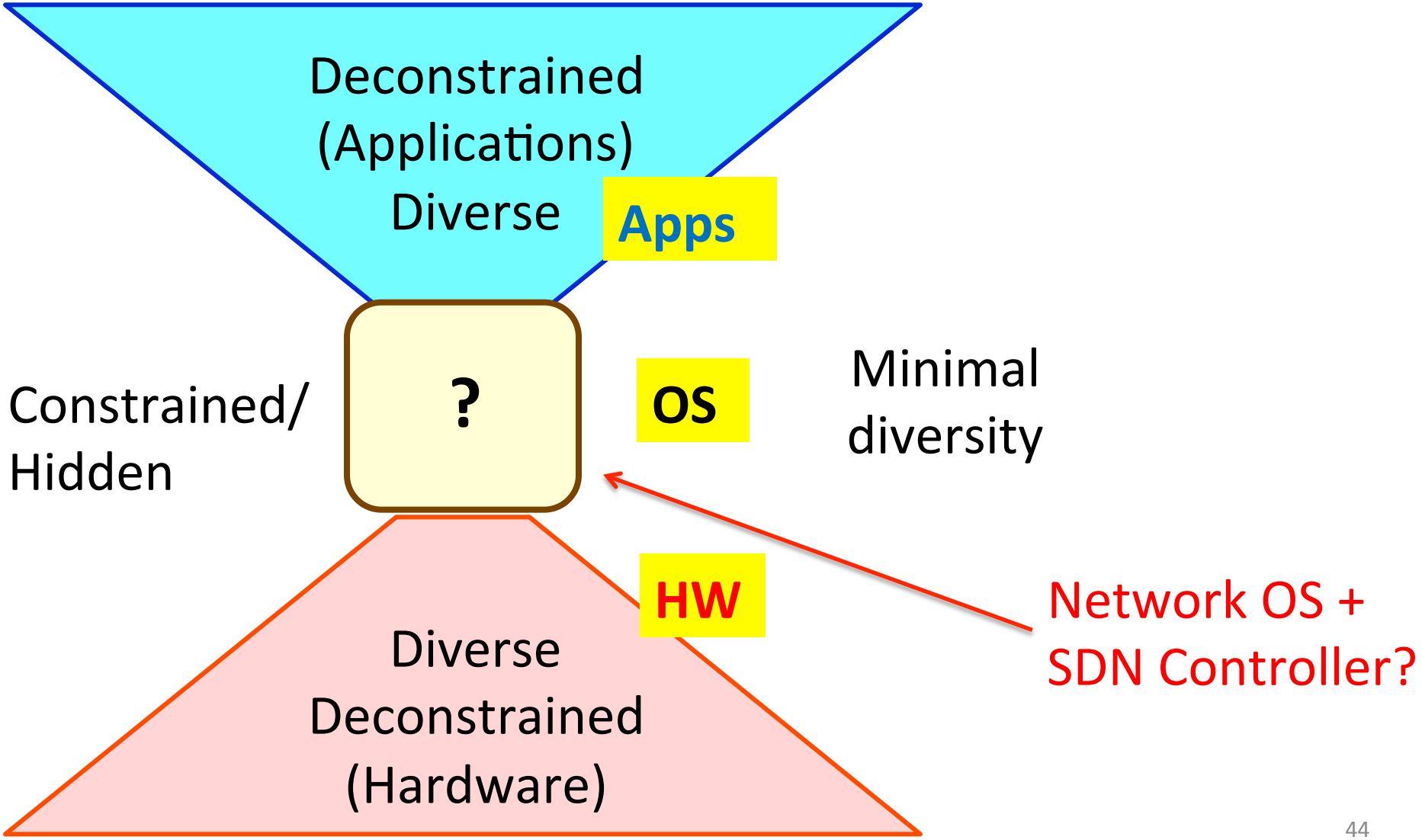


NDN Hourglass



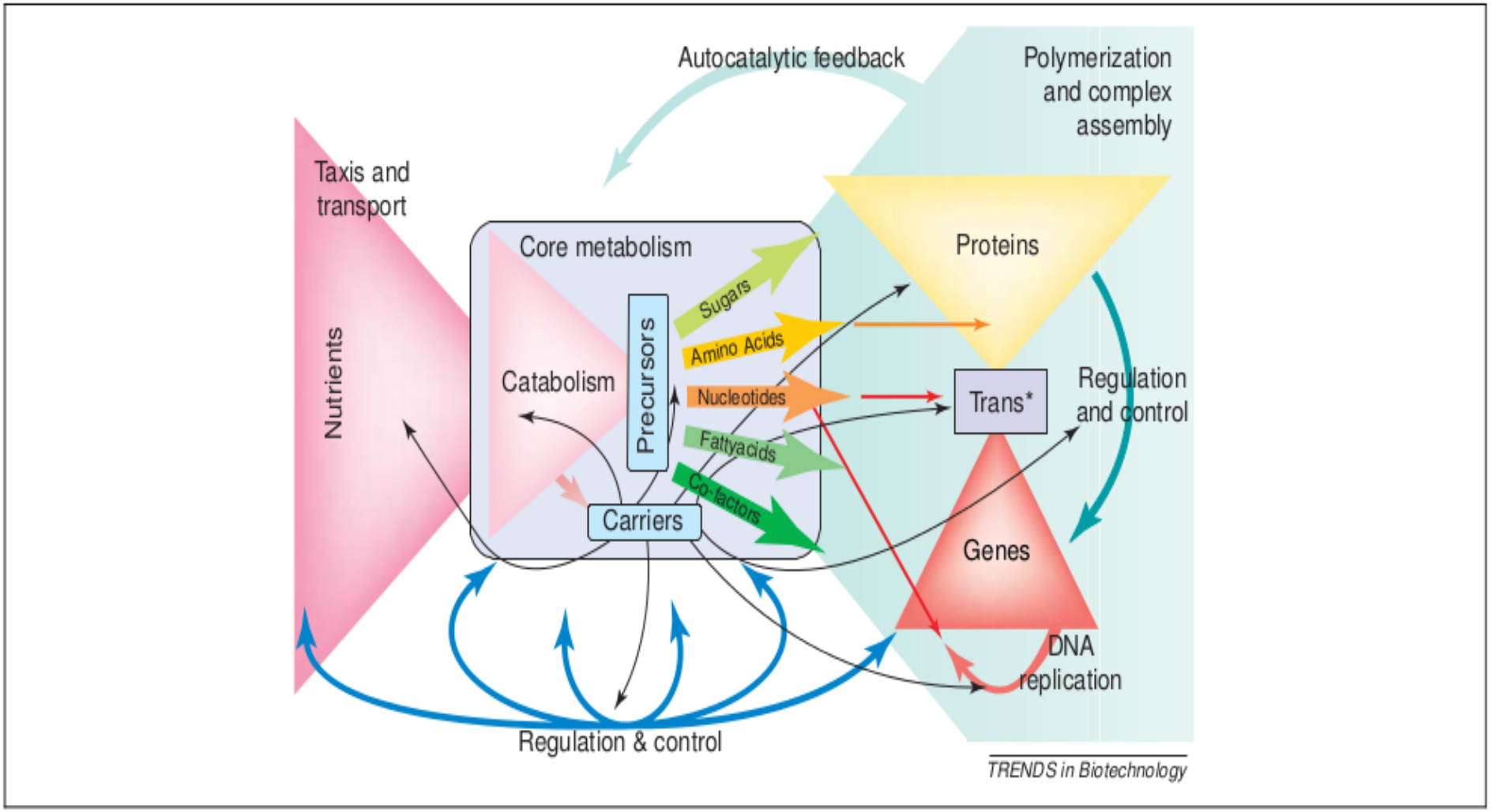
See Named Data Networking, http://named-data.net/

Layered architectures make robustness and evolvability *compatible*



Of Course, in Practice Things are More Complicated

The Nested Bowtie/Hourglass Architecture of Metabolism



See Csete, M. and J. Doyle, "Bowties, metabolism and disease", TRENDS in Biotechnology, Vol. 22 No. 9, Sept 2004

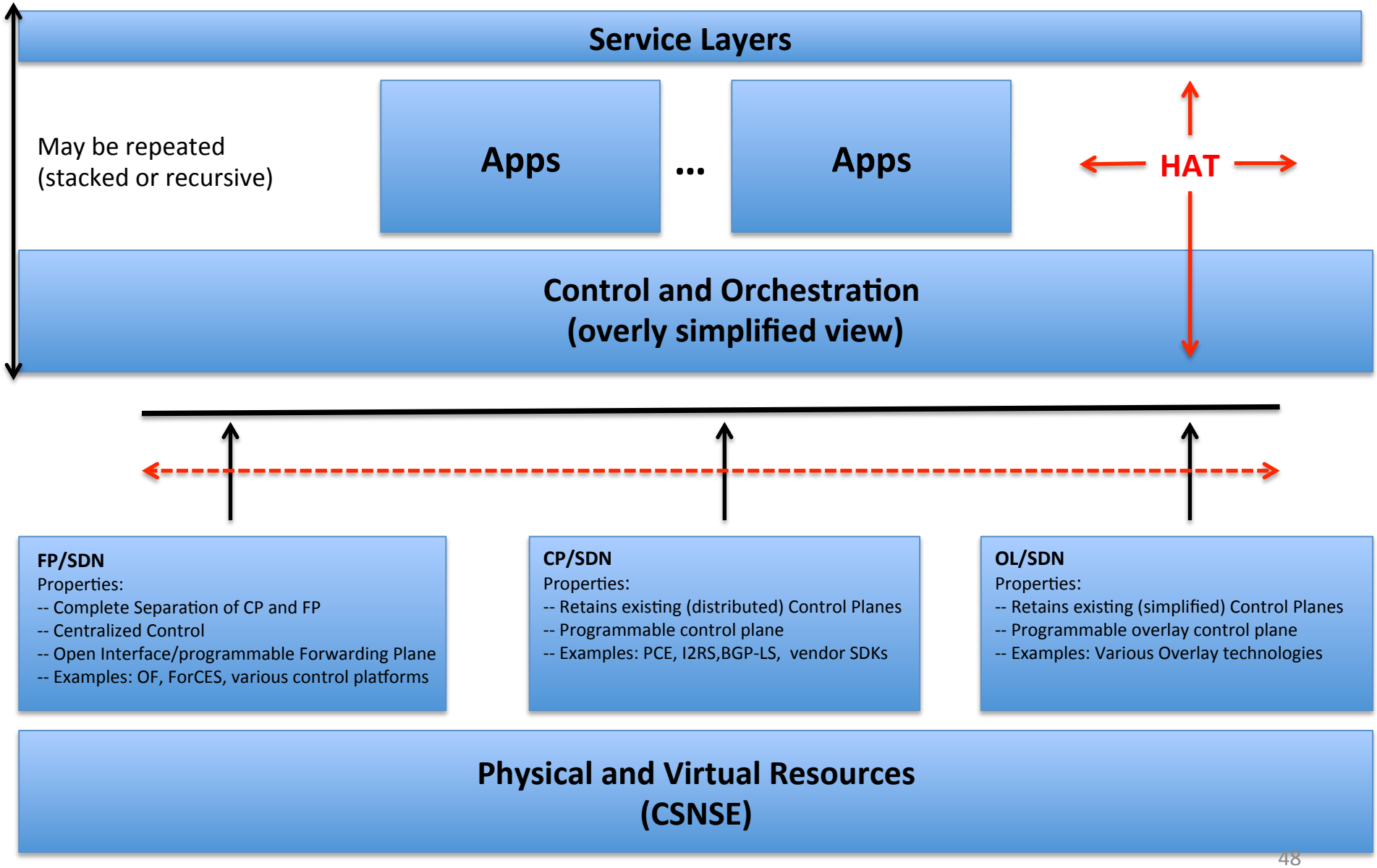
Agenda

- ~~A Couple of Macro Trends~~
- ~~SDN Context: Problem Space and Hypothesis~~
- ~~Complexity, Layered Architectures, and SDN~~
- A Perhaps Controversial View
- Summary and Q&A if we have time

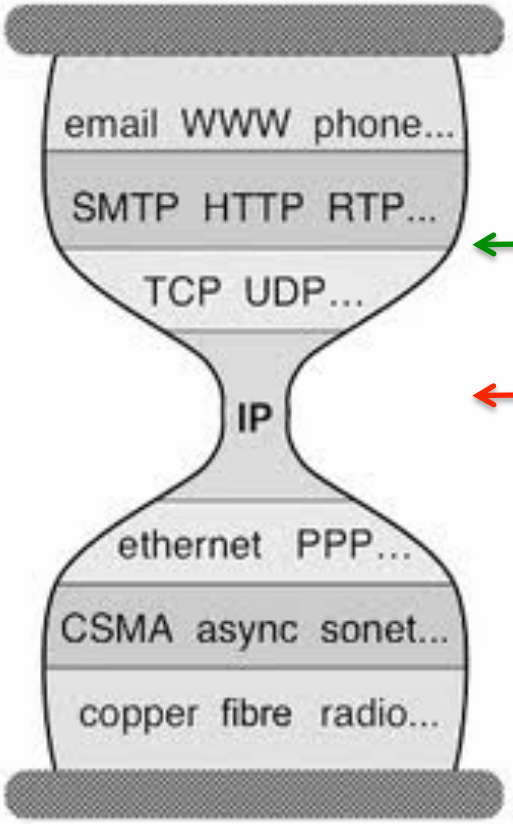
OF/SDN is One Point in a Larger Design Space

- But not the only one
- The larger space includes
  - Compute, Storage, and Network Programmability
  - Security and Energy
- My model: "SDN continuum"
  - http://www.ietf.org/id/draft-haleplidis-sdnrg-layer-terminology-04.txt

A Simplified View of the *SDN Continuum*



Bowties/Hourglasses?



Open Source is a wildcard

OL/SDN

CP/SDN

OF/SDN?

- OF/SDN lower dimensional model makes more sense
- CP/SDN makes existing control planes programmable
- OL/SDN is an application *from the perspective of the Internet's waist*

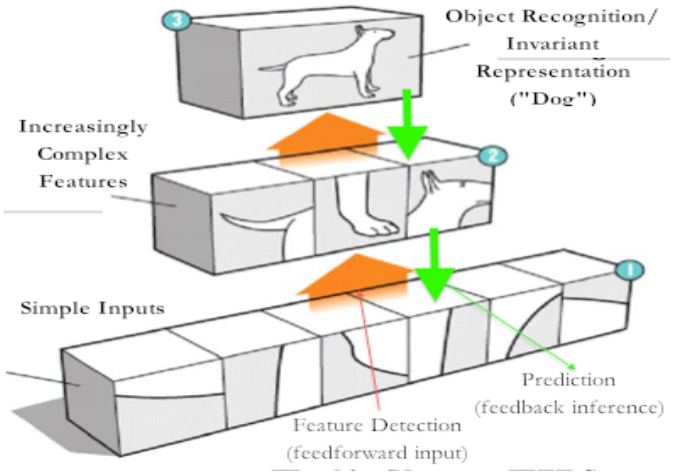
Agenda

- ~~A Couple of Macro Trends~~
- ~~SDN Context: Problem Space and Hypothesis~~
- ~~Complexity, Layered Architectures, and SDN~~
- ~~A Perhaps Controversial View~~
- Summary and Q&A if we have time

So The Future: Where's it All Going?

But More Seriously….

- Current Events
  - ONF: Evolving OpenFlow with things like Table Typing Patterns (TTPs)
  - IETF: Model Driven Everything (I2RS, …)
  - Everyone else (ETSI NFV, Cablelabs, …)
  - Open Source/*Everything*
    - http://www.opendaylight.org
    - http://www.openstack.org
    - http://opencompute.org/
  - Rise of AI (see my other talk on Deep Learning)

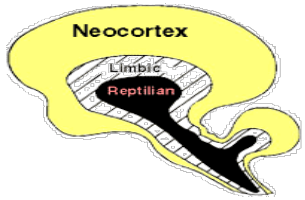


- High Order Bit:
  - System(s) we're building are inherently uncertain → cloudy crystal balls
  - Architect for change and rapid evolution: see Open Source dev methodologies for a clue
  - **Increasing roles for s/w and programmability + Moore's law → volatility/uncertainty**
  - "Above the waist" characterized by uncertainty, e.g., http://spotcloud.com/

- Easy to see: NetOps -> DevOps
  - http://dtucker.co.uk/work/a-netops-to-devops-training-plan.html
  - http://www.slideshare.net/mestery/next-gennetworkengineerskills

But More Seriously….continued

- Conventional Technology Curves – S & F
  - Moore's Law and the reptilian brain
    - Someone eventually has to forward packets on the wire
  - 400G and 1.2 T in the "near" term
  - Silicon photonics, denser core count, ….
  - But also….10^3 core chips communicating via graphene nano-patch antennas
    - In the terahertz band



- The future is all about Software Ecosystems
  - Open Interfaces: Protocols, APIs, Code, Tool Chains
  - Open Control Platforms at every level
  - "Engineering Systems"

- Theoretical Frameworks
  - Requires data-driven multi-disciplinary approaches and
  - Building bridges between engineering and theory communities
    - http://www.1-4-5.net/~dmm/talks/sigcomm2014JDoyleMeyerTutorial.pdf
  - Where is this currently happening? Systems Biology

Where To From Here?

- Robust systems "might be" intrinsically hard to understand
  - RYF complexity is an inherent property of advanced technology
  - Software (e.g., SDN, NFV, Cloud, …) exacerbates the situation
  - And the Internet has reached an unprecedented level of complexity…

- Nonetheless, many of our goals for the Internet architecture revolve around how to achieve robustness…
  - which requires a deep understanding of the *necessary interplay between complexity and robustness, modularity, feedback, and fragility*[1]
    - which is neither accidental nor superficial
  - Rather, architecture arises from "designs" to cope with uncertainty in environment and components
  - The same "designs" make some protocols hard to evolve
  - Does SDN help or hurt, and can we build formal models that help us reason about "universal laws"?

- Understanding these universal architectural features will help us achieve the scalability and evolvability (operability, deployability, understandability) we're seeking from the Internet architecture today and going forward
  - Multi-disciplinary approaches provide a template of how we might go about this (e.g., Systems Biology)

[1] See Marie E. Csete and John C. Doyle, "Reverse Engineering of Biological Complexity", http://www.cds.caltech.edu/~doyle/wiki/images/0/05/ScienceOnlinePDF.pdf

Q&A

Thanks!