Architectural Musings on SDN
(“and now for something completely different...”)

David Meyer
CTO and Chief Scientist, Brocade
Director, Advanced Technology Center, University of Oregon
I2 Member Meeting 2013
dmm@{brocade.com,uoregon.edu,1-4-5.net,...}
Agenda

• Architectural Features that Enable System Scalability and Evolvability

• A Quick Tour Through the SDN Design Space

• Panel
Danger Will Robinson!!!

This talk is intended to be controversial/provocative (and maybe a bit “sciencey”)

What are Scalability and Evolvability?

- **Scalability** is robustness to changes to the size and complexity of a system as a whole.

- **Evolvability** is robustness of lineages to changes on 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.
OK, Fine. But What is Robustness?

• **Definition:** A property of a system is robust if it is invariant with respect to a set of perturbations, up to some limit.

• **Fragility** is the opposite of robustness
  – If you're fragile you depend on 2nd order effects (acceleration) and the curve is concave
  – Catch me later if you’d like to chat further about this...

• 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 (RYF-complex)** [0]
  – Or the system may collapse if it experiences perturbations above a certain threshold (K-fragile)

• 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

What this curve is telling us is that a system needs complexity to achieve robustness (wrt some feature to some perturbation), but like everything else, too much of a good thing....
Ok, but what is *Complexity*?

“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.” [AldersonDoyle2010]
BTW, This Might Be Obvious But...

- Networks are incredibly general and expressive structures

- Networks are extremely common in nature
  - Immune systems, energy metabolism, transportation systems, *Internet*, macro economies, forest ecology, the main sequence (stellar evolution), galactic structures, ....

- So it comes as no surprise that we study, for example, biological systems in our attempts to get a deeper understanding of complexity and the architectures that provide for scalability, evolvability, and the like

- Ok, this is cool, but what are the key architectural takeaways from this work for us?
  - where us \in \{ops, engineering, architects ...\}
Key Architectural Takeaways

• What we have learned is that there are fundamental architectural building blocks found in systems that scale and are evolvable. These include

  – Bowtie architectures

  – RYF complexity

  – Massively distributed, robust control loops
    • Contrast optimal control loops and hop-by-hop control

  – Highly layered
    • But with layer violations

  – Protocol Based Architectures (PBAs)

  – Degeneracy
Bowties 101

Constraints that Deconstrain

For example, the reactions and metabolites of core metabolism, e.g., ATP metabolism, Krebs/Citric Acid cycle signaling networks, ...

But Wait a Second
Anything Look Familiar?

Bowtie Architecture

Hourglass Architecture
The SDN Design Space

Service Layers

May be repeated (stacked or recursive)

Apps ...

Apps

Control and Orchestration
(overly simplified view)

DP/SDN
Properties:
-- Complete Separation of CP and DP
-- ("Logically") Centralized Control
-- Open Interface/programmable Data Plane
-- Examples: OF, ForCES, various control platforms

CP/SDN
Properties:
-- Retains existing (distributed) Control Planes
-- Programmable control plane
-- Network aware applications
  Explicitly "not" e.g., learning switch
-- Examples: PCE, I2RS, vendor SDKs

OL/SDN
Properties:
-- Retains existing (simplified) Control Planes
-- Programmable overlay control plane
-- May use OF to program vSwitches
-- Examples: Various Overlay technologies

Physical and Virtual Resources
(CSN)
• **OF/SDN** proposes a new architectural waist (not exactly sure where)
• **CP/SDN** makes existing control planes programmable
• **OL/SDN** is an application *from the perspective of the Internet’s waist*

---

*Open Loop Control + s/w + Moore’s Law → Randomness, Uncertainty, and Volatility*
Finally...What I Hope To Achieve

I hope to convince you that uncertainty and volatility are the “coin of the realm” of the future, why this is the case, how SDN (and the rise of software in general) is accelerating this effect, and finally, what we might do to take advantage of it.¹

¹ s/take advantage of/survive/ -- @smd
Q&A

Thanks!