NC STATE UNIVERSITY

Net SILOs: An Architecture to Enable Software Defined Optics

George N. Rouskas

Department of Computer Science

North Carolina State University

http://net-silos.net/

Joint work with: Ilia Baldine (RENCI), Rudra Dutta (NCSU), Dan Stevenson (RTI), Anjing Wang (NCSU), Manoj Vellala (NCSU)

Outline

- Context: The Clean-Slate Debate
- Motivation: Software Defined Optics
- SILO Network Architecture: The Story So Far
- Summary and Future Directions

Context (1)

The Internet is broken!

Context (1)

The Internet is broken! (has ossified / reached an impasse)

Context (1)

The Internet is broken!

Security is a mess: it is hard to

- identify users
- prevent them from causing harm
- hold them accountable

Context (1)

The Internet is broken!

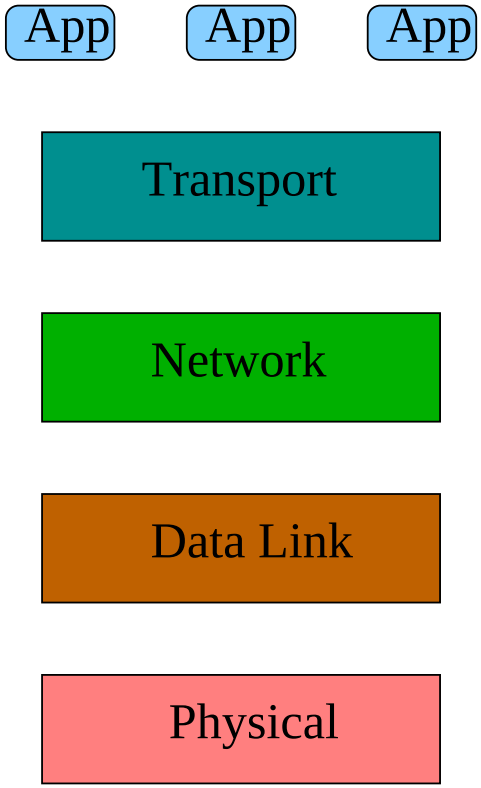
Middleboxes violate end-to-end principle:

- firewalls
- NAT
- proxies

Context (1)

The Internet is broken!

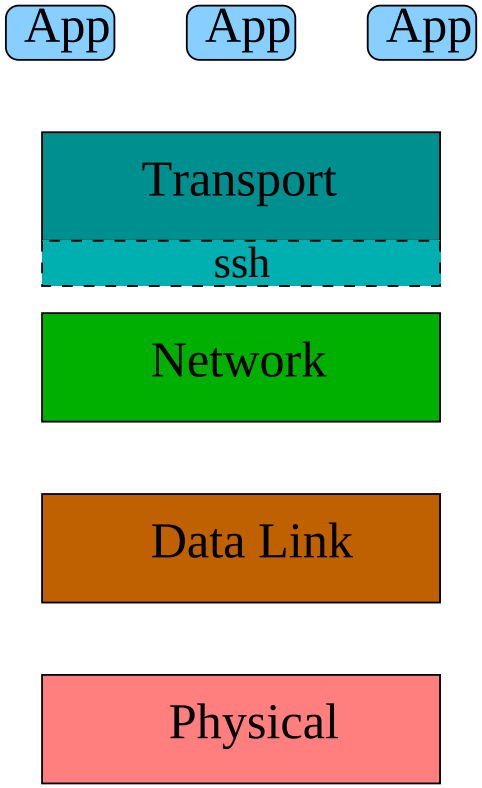
Fixed layer architecture is outdated



Context (1)

The Internet is broken!

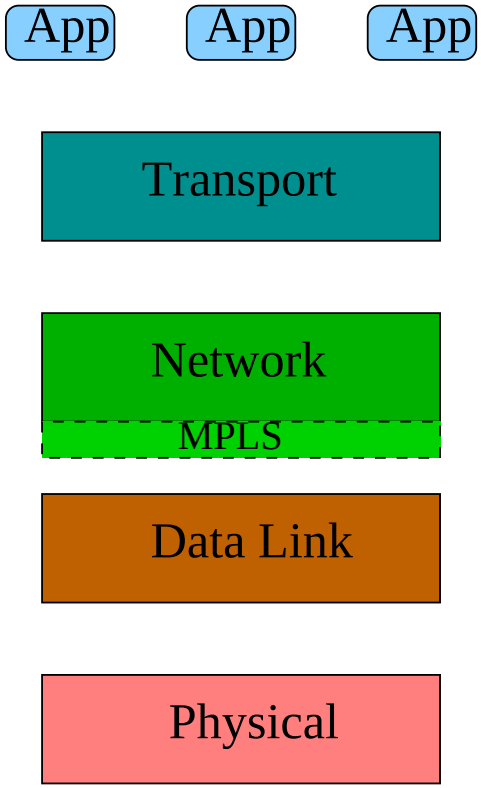
Fixed layer architecture is outdated



Context (1)

The Internet is broken!

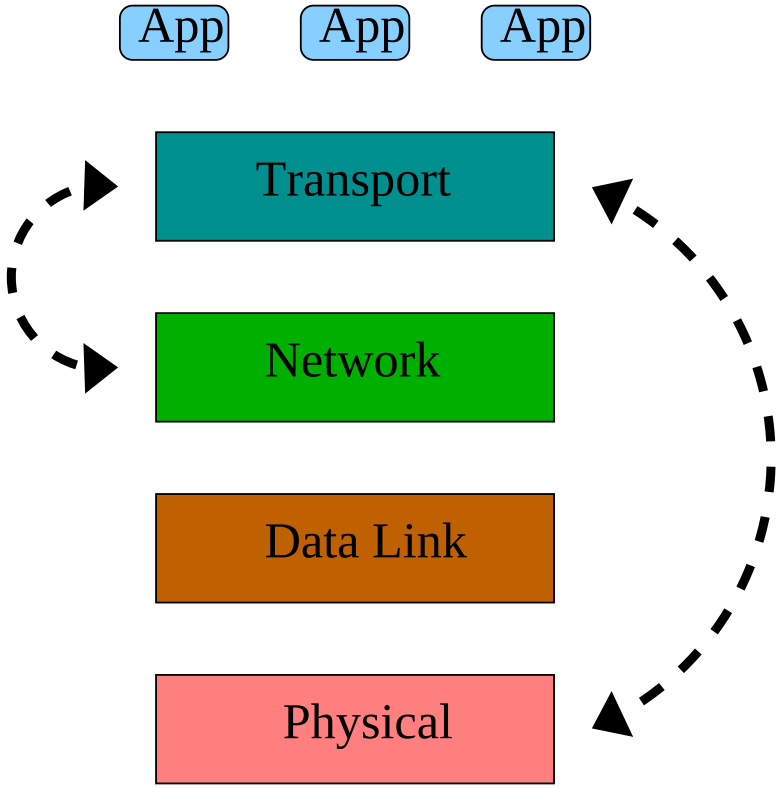
Fixed layer architecture is outdated



Context (1)

The Internet is broken!

Cross-layer interactions difficult: TCP over wireless



Context (1)

The Internet is broken!

Clear need for clean-state initiatives $\longrightarrow$ NSF FIND, EU FIRE, $\cdots$

1. research in new network architectures
2. large-scale experimental facilities $\longrightarrow$ GENI

Context (2)

The Internet is doing just fine, thank you!

Context (2)

The Internet is doing just fine, thank you!

- Biological metaphor: mutation and natural selection
- Evolutionary designs: more robust, less expensive
- Mid-layer protocols must be conserved – not ossified
  $\longrightarrow$ innovation at lower/upper layers of architecture

Context (2)

The Internet is doing just fine, thank you!

- Biological metaphor: mutation and natural selection
- Evolutionary designs: more robust, less expensive
- Mid-layer protocols must be conserved – not ossified
  $\longrightarrow$ innovation at lower/upper layers of architecture

$\longrightarrow$ Evolution beats revolution

Our View

- Internet architecture successful in accommodating change

Our View

- Internet architecture successful in accommodating change
- But: current practice of patches/tweaks cannot continue forever

Our View

- Internet architecture successful in accommodating change
- But: current practice of patches/tweaks cannot continue forever
- New architecture must be designed for adaptability/evolvability

Our View

- Internet architecture successful in accommodating change
- But: current practice of patches/tweaks cannot continue forever
- New architecture must be designed for adaptability/evolvability
- SILO objective:

Our View

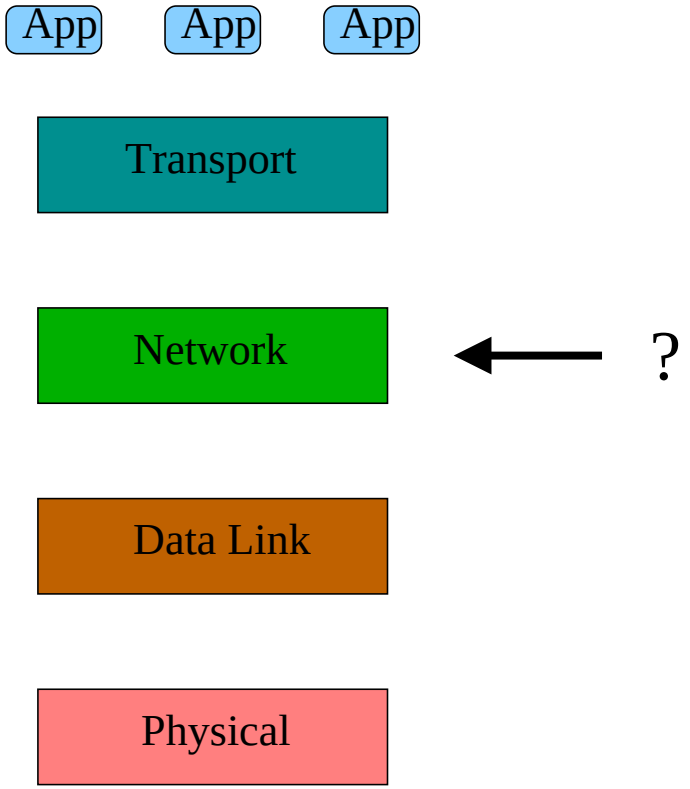
- Internet architecture successful in accommodating change
- But: current practice of patches/tweaks cannot continue forever
- New architecture must be designed for adaptability/evolvability
- SILO objective:

The goal is not to design the “next” system, or the “best next” system, but rather a system that can sustain continuing change

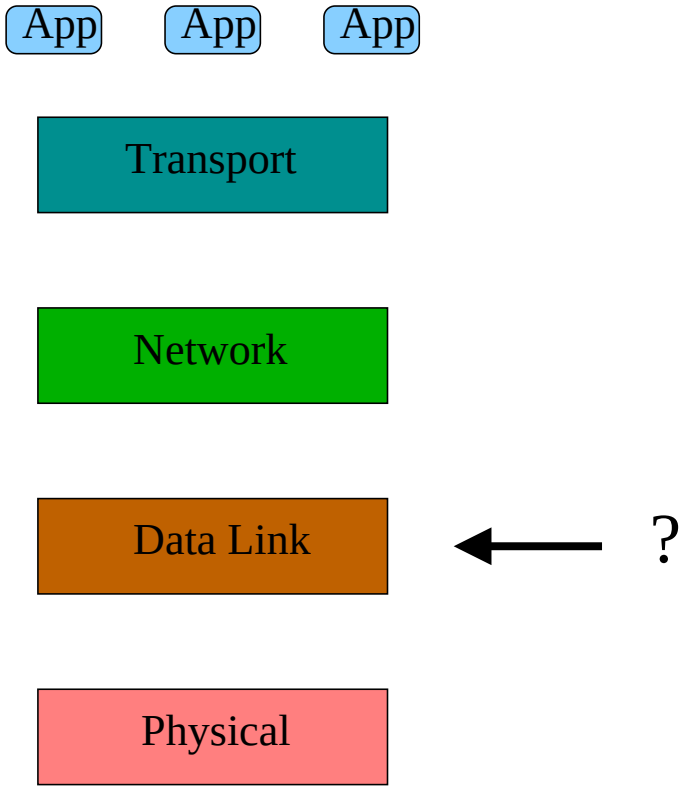
OBS And The Layer Stack

Where does OBS fit in the stack?

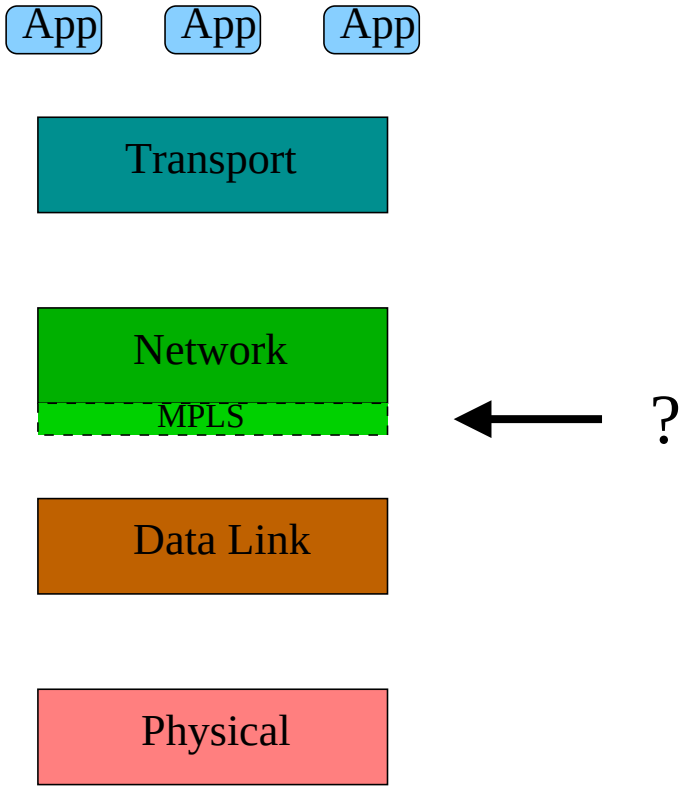
OBS And The Layer Stack



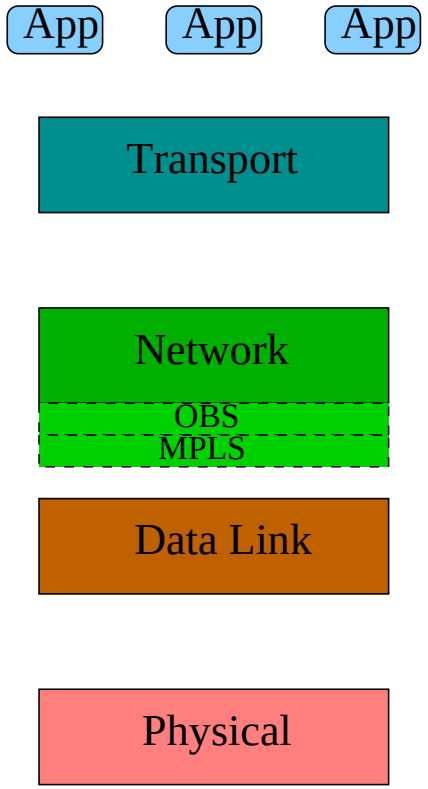
OBS And The Layer Stack



OBS And The Layer Stack



OBS And The Layer Stack



Cross-Layer Interactions: TCP Over OBS

Does “TCP Over OBS” make sense?

Cross-Layer Interactions: TCP Over OBS

Does “TCP Over OBS” make sense?

Yes!

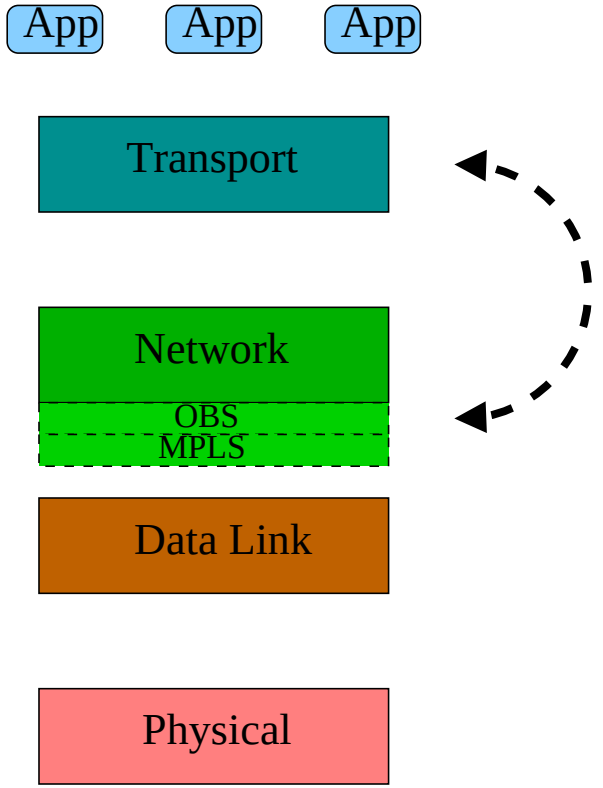
- TCP carries ≈95% of Internet traffic
- good understanding of TCP performance is crucial

Cross-Layer Interactions: TCP Over OBS

Does “TCP Over OBS” make sense?

No!

- which TCP flavor?
- which OBS flavor?
- transport and OBS layers must be optimized for each other
- not as straightforward as “TCP over wireless”



Software Defined Optics

- Optical substrate can no longer be viewed as black box

Software Defined Optics

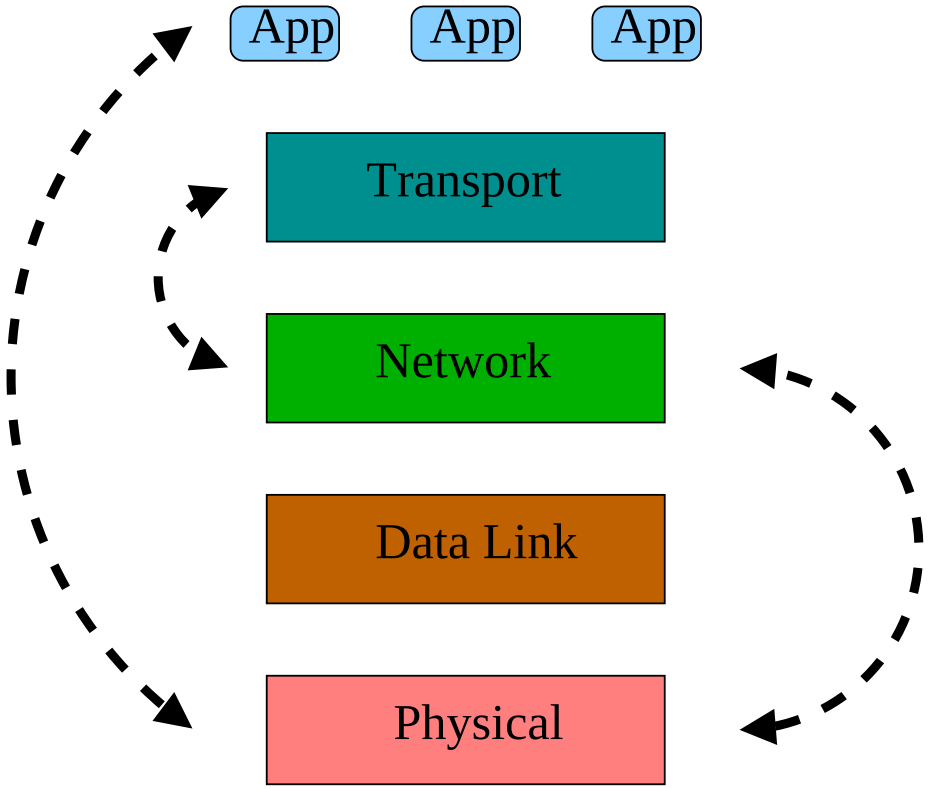
- Optical substrate can no longer be viewed as black box
- Collection of intelligent and programmable resources:

Software Defined Optics

- Optical substrate can no longer be viewed as black box
- Collection of intelligent and programmable resources:
  - optical monitoring, sensing mechanisms
  - amplifiers, impairment compensation devices
  - tunable optical splitters
  - configurable add-drop
  - programmable mux-demux (e.g., adjust band size)
  - adjustable slot size
  - . . .

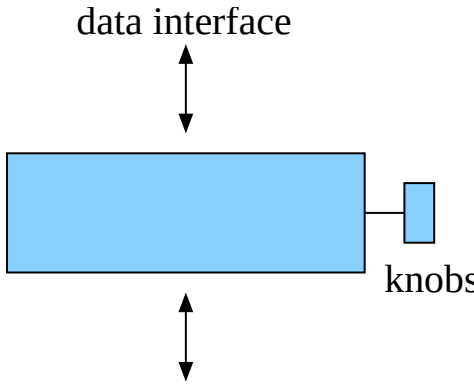
Cross-Layer Interactions

- Impairment-aware routing
- Traffic grooming
- Network resiliency
- . . .

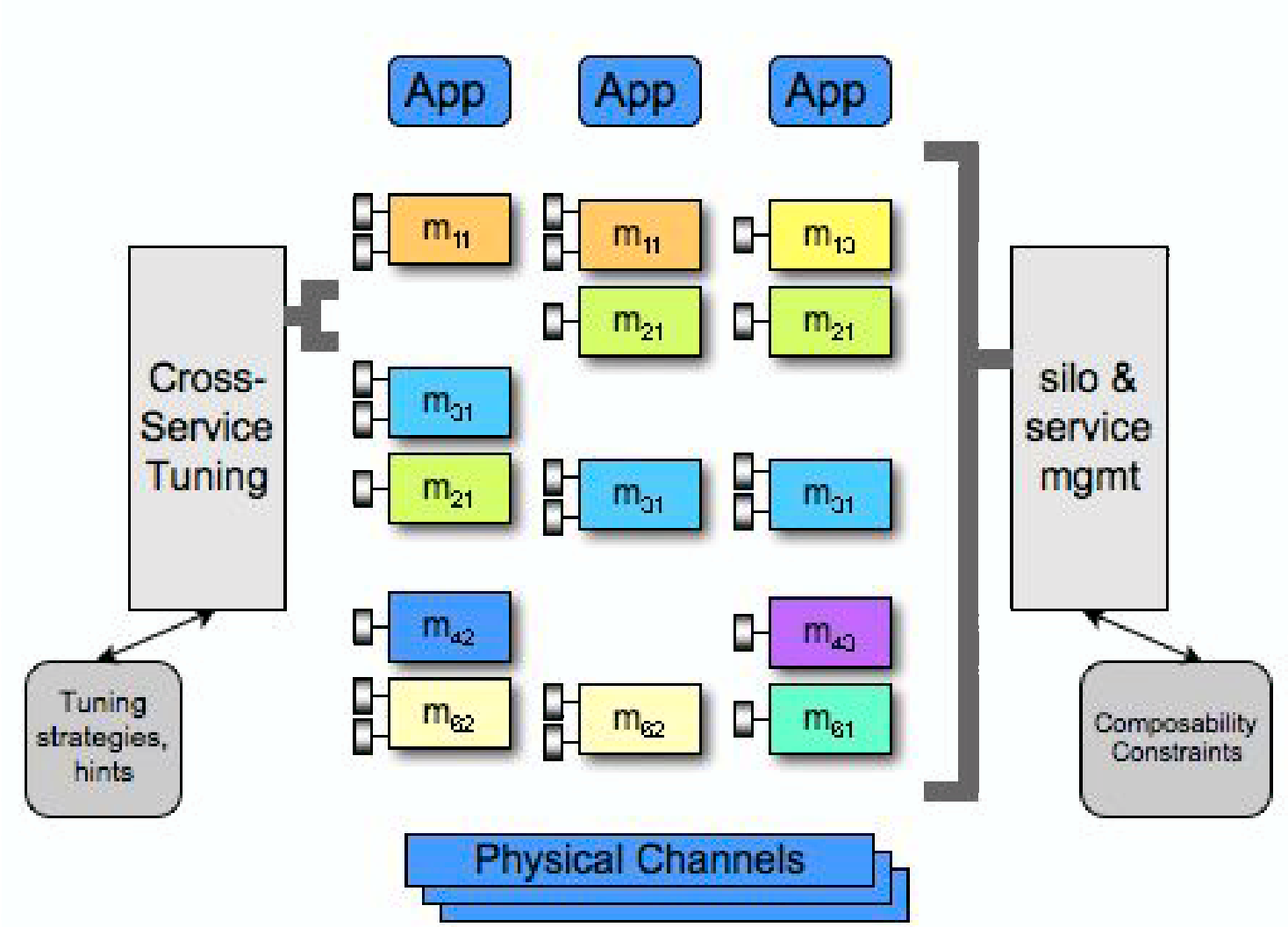


SILO Architecture Highlights

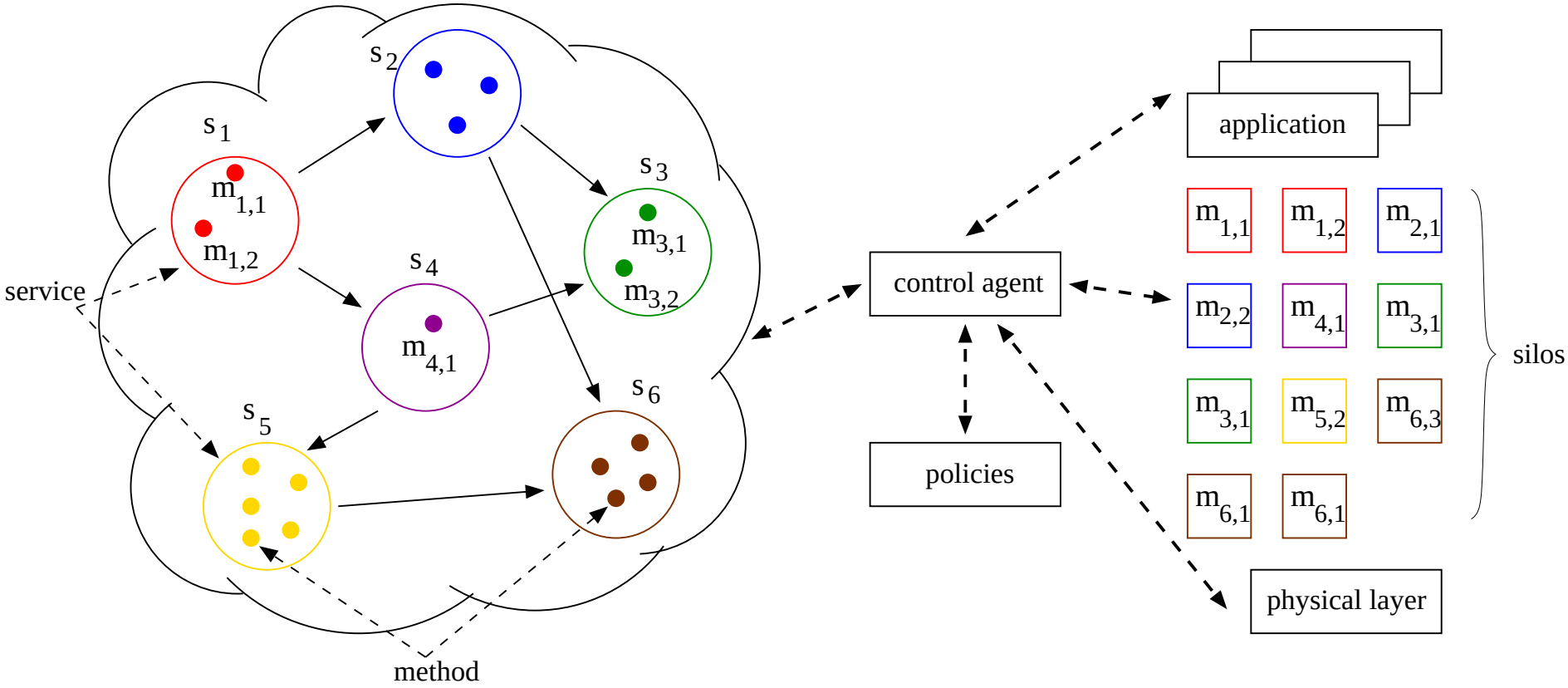
- Generalizes traditional layer stack:
  - services: building blocks of fine-grain functionality
  - silo: per-flow vertical composition of services
  - decoupling of layers and services
- Enables inter-layer interactions:
  - knobs: explicit control interfaces
- Facilitates introduction of new services:
  - ontology: describes services and their relationships
  - composition algorithm to construct silos
  - standard ontology languages and reasoning engines may be used



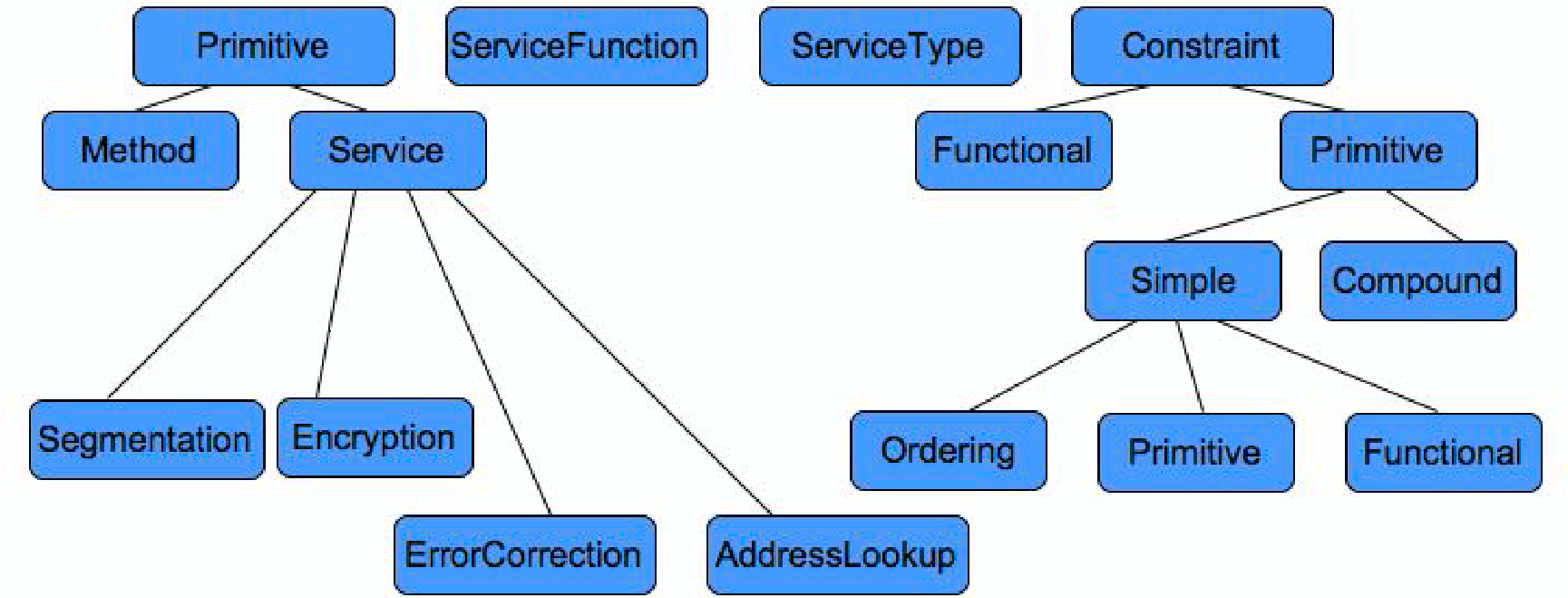
SILOs


App
App
App
Cross-Service Tuning
silo & service mgmt
Tuning strategies, hints
Composability Constraints
Physical Channels

Ontology


s 1
s 2
s 3
s 4
s 5
s 6
m 1,1
m 1,2
m 3,1
m 3,2
m 4,1
service
method
control agent
policies
application
physical layer
m 1,1
m 1,2
m 2,1
m 2,2
m 4,1
m 3,1
m 3,1
m 5,2
m 6,3
m 6,1
m 6,1
silos

Ontology – Networking Knowledge

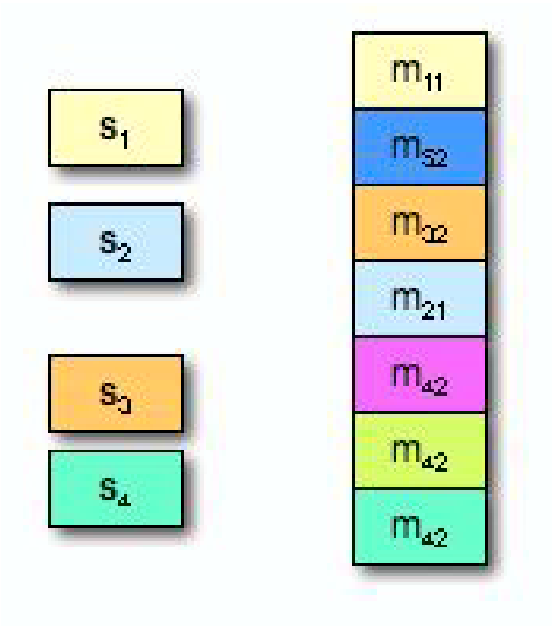


Service Composition

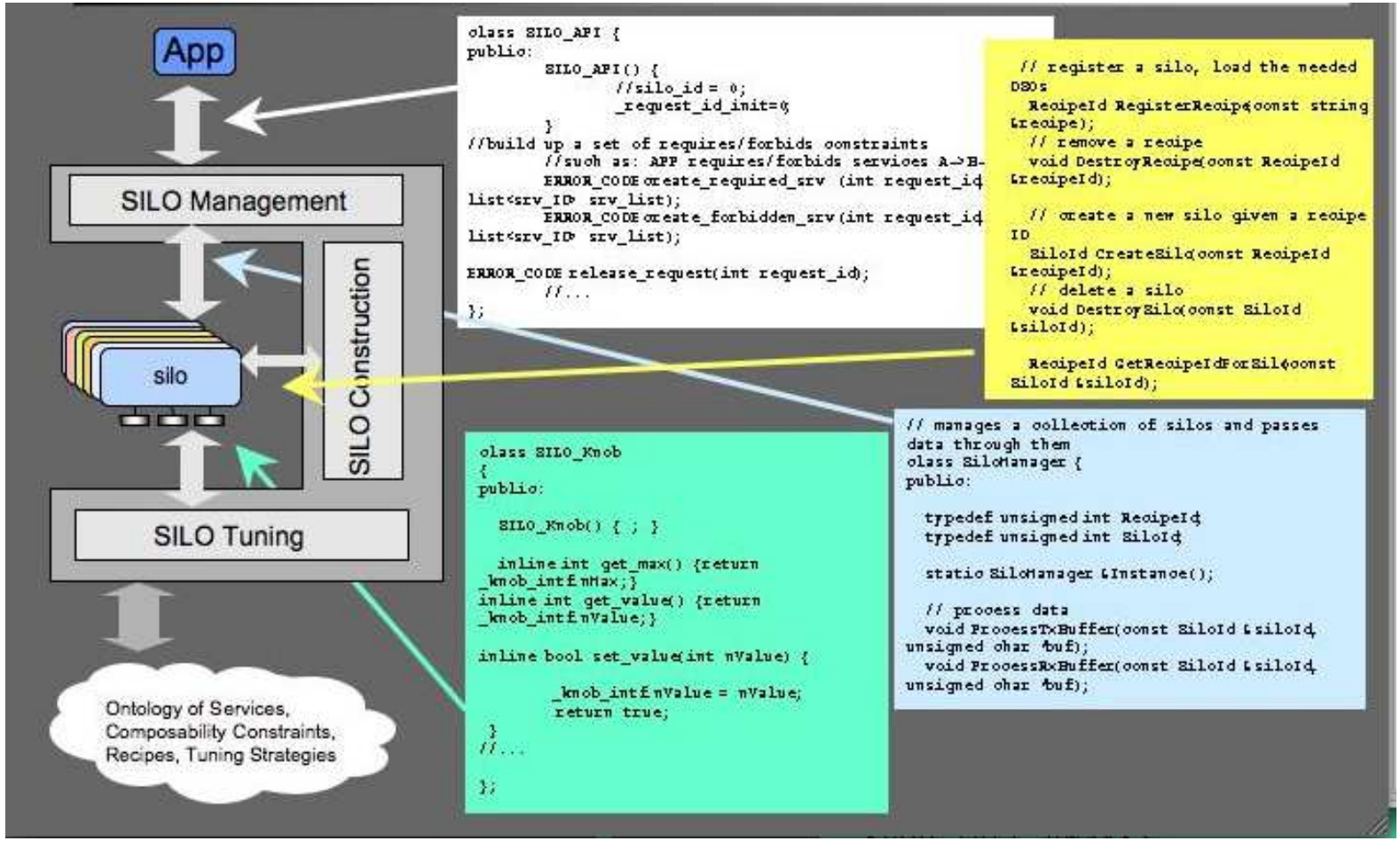
- Constraints on composing services A and B:
  - A requires B
  - A forbids B
  - A must be above (below) B
  - A must be immediately above (below) B
  - Negations, AND, OR
- Minimal set:
  - Requires, Above, ImmAbove, NotImmAbove
- All pairwise condition sets realizable
  - Forbids = (A above B) AND (B above A)
  - Above = NOT Below

Composition Problem

- Given: a set of essential services ⟵ application
- Obtain a valid ordering of these and additional services
  - or, identify conflicts with constraints
- Simple composition algorithm



SILO Software Prototype



http://net-silos.net/

Summary

- Vision – enable flexibility, evolution: “design for change”
  - fine-grain, reusable services, explicit control interface
    - enables experimentation, flexibility, community of innovation
  - per-flow service composition (silos)
    - ease of evolution, policies
- Framework – provide architectural support to vision:
  - constrained composition
  - commoditize cross-layer interaction / optimization
- Ongoing efforts:
  - extend the prototype
  - new research directions: software defined optics, virtualzation
  - influence GENI development efforts