Research Problems in SDN

Kireeti Kompella
Contrail Systems
IETF 85 – SDNRG

SDN: a Fundamental Step Forward?
(or just a new whip to beat vendors with?)

What really attracts me to SDN:

1. The idea that a network is more than the sum of its parts
   - I.e., take a network-wide view rather than a box-centric view
2. The idea that creating network services can be a science rather than a set of hacks-on-hacks-on-hacks
   - Especially hacks that vary by box, by vendor and by OS version
3. The idea that there should be a discipline and methodology to service correctness
   - Rather than testing (and more testing), declaring victory, only to fail in the real world because of some unanticipated interaction

Criteria for Success

SDN is a real step

1. IF SDN gives us an **abstraction** of the network
2. IF, through this abstraction, we have a means of **reasoning** about the network and network services
3. IF SDN offers a means of verifying **correct operation** of the network or of a service
4. IF SDN offers a means of predicting **service interaction**
5. Finally, IF SDN offers a means of setting (conceptual) **asserts** by which we can get early warning that something is wrong

First Question: Models of SDN

- There are several (quite different) models of SDN
  - In fact, I'll offer a new one
- There has been no attempt (to my knowledge) to classify or categorize SDN (I don't mean areas of deployment)
- This presentation proposes one, but perhaps that should be the first question that SDNRG tackles:

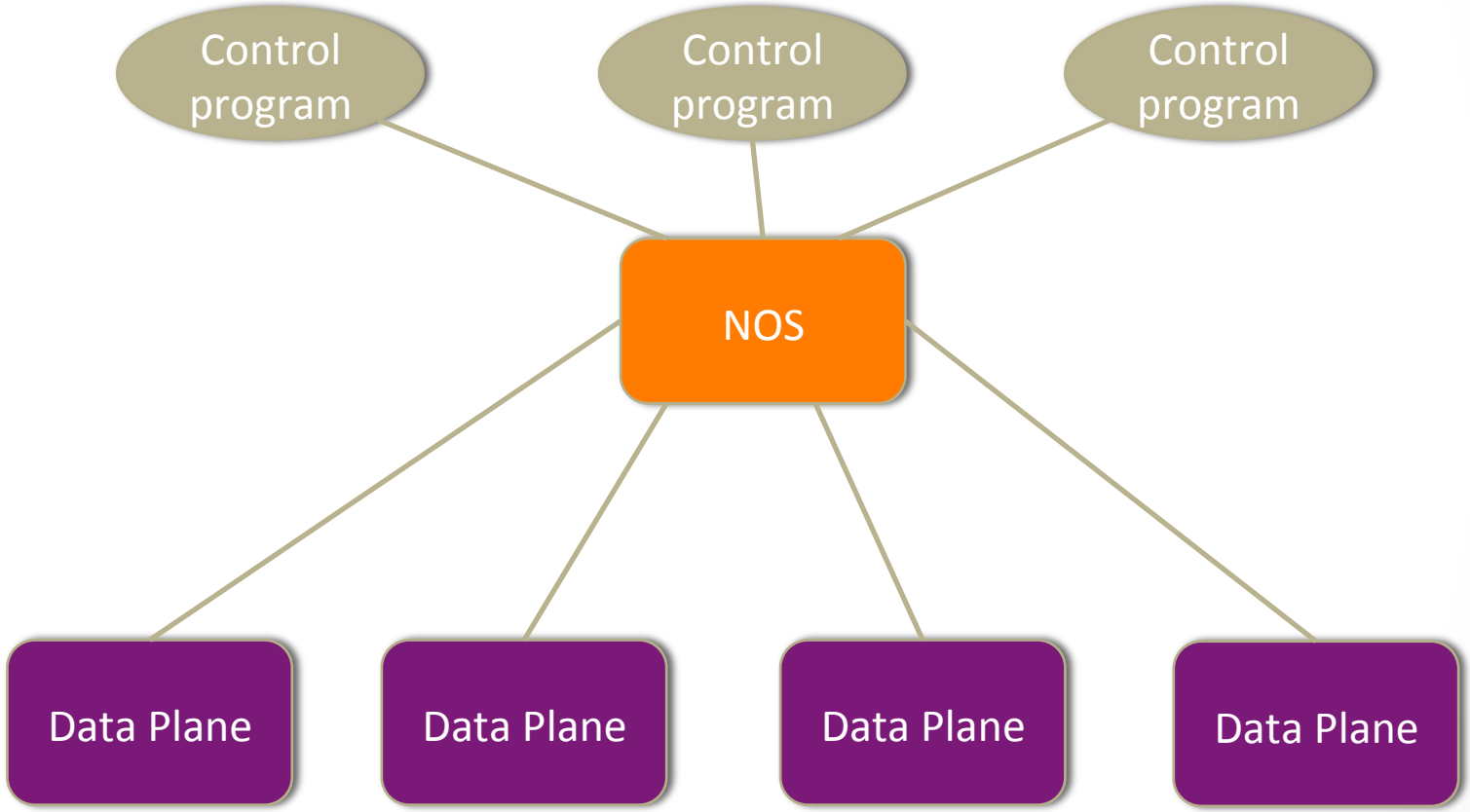
What are the distinct models of SDN?

Models of SDN

SDN can be considered in terms of three distinct models

1. A **Networking Operating System** that oversees the network data plane and hosts a number of "control programs" that define networking services
2. A **Broker** through which applications interact with and affect the network so that the apps are more effective, are more efficient and/or offer better user experience
3. A **Compiler** that translates a high-level language in which an operator defines what they want from the network and compiles it into low-level instructions for the data plane

1. SDN as a Network OS



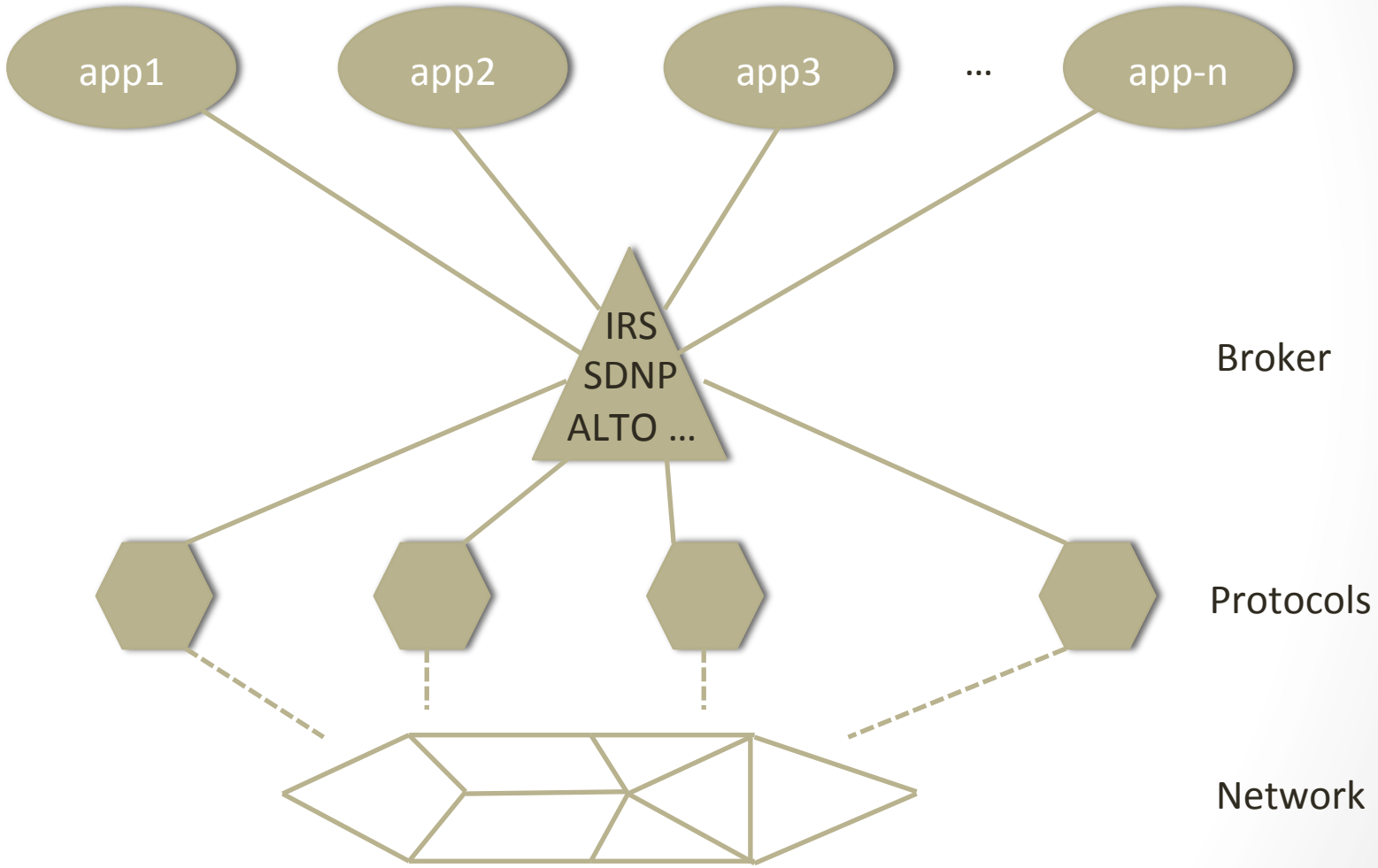
1. SDN as a Network OS

- The NOS offers a set of services (just as a “normal” OS offers scheduling, memory management, device abstraction, etc.)
- These services constitute a new “POSIX”, and are accessible through a set of APIs or libraries …
- … with which user can write an (imperative) control program to transform the current network state to a desired state
  - For example, the desired state might be a set of traffic engineered paths that reflect new bandwidth requirements
  - Or the desired state may be a new location for a Virtual Machine, with associated network state

1. SDN as a Network OS

- Fundamental abstraction
  - Network state = annotated graph of underlying data plane
- Reasoning about the network and services
  - Control programs are transformations of network state
- Correct operation of the network and services
  - Analyze network transformations as program correctness
- Anticipating service interaction
  - Need to develop a “calculus” of network transformations
- Creating *asserts* in the network
  - Need to create the equivalent of *loop invariants* at critical junctures of a transformation

2. SDN as a Broker



2. SDN as a Broker

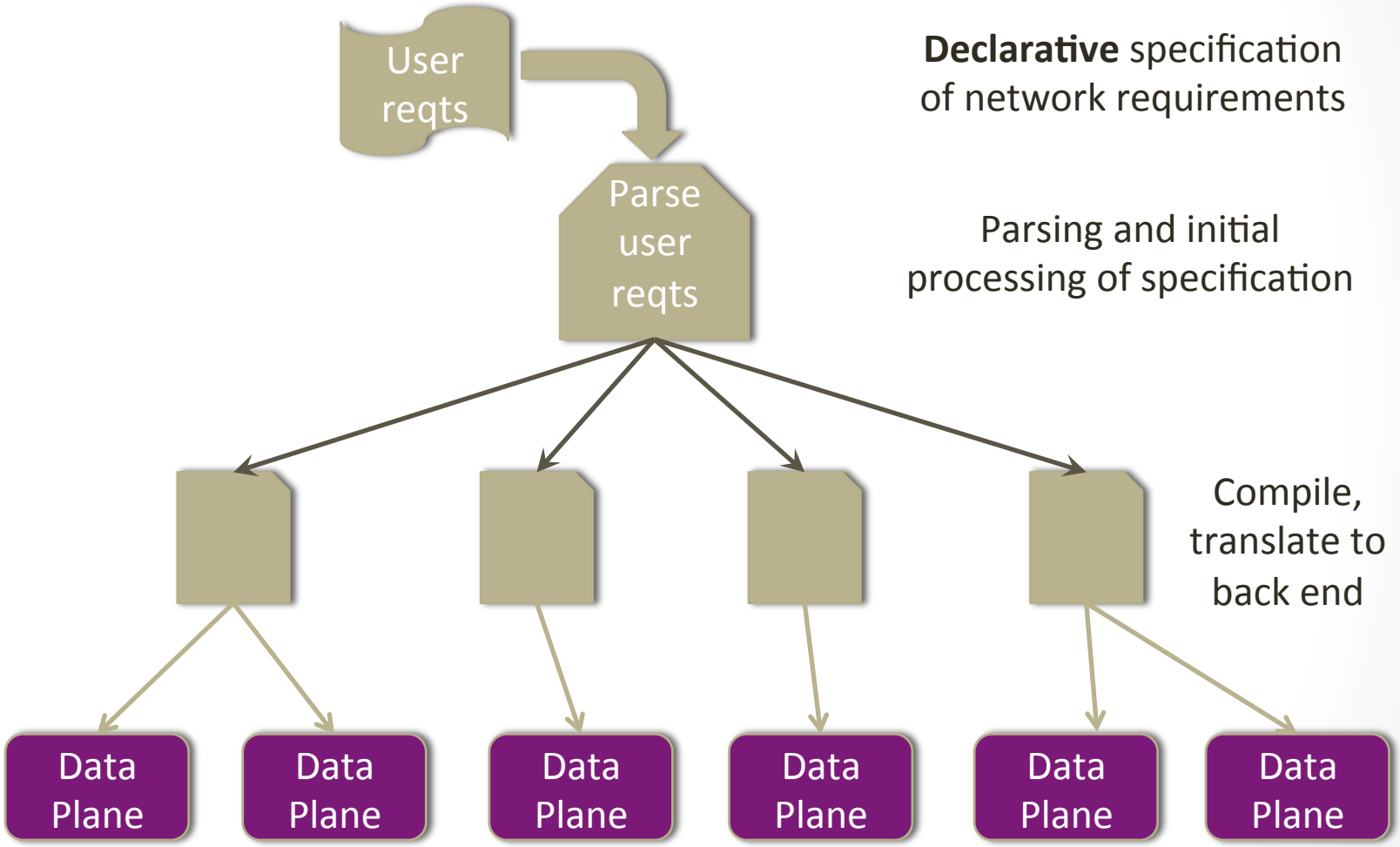
- This style of SDN offers the abstraction of, well, a broker
  - a translator (to go from protocols to APIs and vice versa),
  - a "condenser", which summarizes network properties, and
  - a security/policy gateway (which app is allowed to learn what and change what, and who gets priority)
- This style of SDN lets apps learn about the network …
- … then talk to the network to optimize app performance

2. SDN as a Broker

- This model is a very pragmatic approach to a very real problem in networking: how can applications and networks stop pretending that the other doesn't exist? :-)
  - E.g., Netflix works hard to overcome network congestion by clever programming and clever video encoding …
  - … but the picture can still pixelate or go into "Buffering" or SD
- This is a hard problem that's been tackled before …
  - … with limited success
  - Perhaps this time the outcome will be better
- However, it's hard (*) to see this as a fundamental step forward, notwithstanding the value of the outcome

(*) for me; but I'll keep an open mind

3. SDN as a Compiler



Declarative specification of network requirements

Parsing and initial processing of specification

Compile, translate to back end

3. SDN as a Compiler

- This style of SDN offers the abstraction of a high-level, declarative programming language
- The network administrator's job is to specify how she wants the network to look, who can talk to whom and how, etc.
- The SDN compiler then has to translate the high-level declarations, requirements and constraints to low-level instructions that each data plane element can implement
  - The "hacks that vary by box, by vendor and by OS version" is the compiler's problem, not the network administrator's!

3. SDN as a Compiler

- Fundamental abstraction: high-level specifications
- Reasoning about the network and services: a calculus of these specifications
  - The current state and/or the desired state of the network is irrelevant, as are the transformations one makes to get there
- Correctness of the network and/or services:
  - Requires matching specifications to requirements
  - Depends on how intuitive/natural the specification language is
- Anticipating service interaction
  - Requires understanding how specifications dovetail
- Asserts
  - Need to relate specifications to network state

Comments

- Should this all appear far too theoretical and idealistic, I'll confess to being a huge fan of Dijkstra (and Gries)
- The Art (and Science) of Programming may never have offered a convincing proof of a large, real-world program
  - But the idea of programming as a discipline has had a profound influence on those who want to write good programs
  - Could you put a conditional breakpoint in a loop without thinking (perhaps intuitively) about a loop invariant?
- A worthy goal of SDN is to create a discipline of networking

- But why am I apologizing for aiming high? ☺