Zen and the Art of Network Architecture

Larry Peterson
Zen and the Art of Motorcycle Maintenance
by
Robert Pirsig

- Rejected by 121 publishers (World Record)
- Classic v Romantic Perspectives
  - Rational vs Mystic
  - Analytical vs Intuitive
  - Science vs Art
Classic View
Romantic View
Quality

• Unifies Classic and Romantic Perspectives
• Whole is greater than the sum of the parts
• More about potential than measurable value
Buddhism’s First Noble Truth

Life is Suffering
Duality – Networking vs Distributed Systems
The Middle Way

• Involves Both Analysis and Intuition
• Balances Requirements*
  – Not about optimizing any one dimension
• Seeks Unifying Abstractions
  – Accommodates both this and that

Identifies 11 requirements (dimensions) and offers “rules” on resolving 7 inter-requirement tensions.
Path to Enlightenment
Path to Enlightenment

Maturity

Time

Controlled Lab Experiments

Deployment Studies

Pilot Demonstrations

Commercial Adoption

Change the Market

Market Reality

Customer Reality

Traffic & User Reality

Implementation Reality

Analysis

Idea

Analysis
Change the Market

• Operator CDNs...
  – Now incentives for CDN Interconnection (CDNI)
• Virtualized Commodity Servers at the Edge...
  – Enables Network Function Virtualization (NFV)
  – Dovetails with (but distinct from) SDN
Commodity Servers in the Net
NFV Proof-of-Concept
– with BT, Intel & HP –
Path to Enlightenment

• See Reality Clearly – Assumptions hide the truth
• Experience-Based – Users reveal hidden assumptions
• Operationalize – The New Bar!
  – Deploy & Operate > Implement > Thought Experiment
Entropy

- A Measure of Engineering’s Effect on Architecture
  - Natural part of the process
- Design Principles*
  - Acknowledge the dynamic nature of systems
- How Architecture Manifests
  - Represents the “fixed point” of an architecture

Manifestation of an Architecture

• Circa 1981 (ASCII renderings of protocol headers)
Manifestation of an Architecture

• Circa 2013 (Django Object Class Definition)

class Slice(PlCoreBase):
    tenant_id = models.CharField(max_length=200, help_text="Keystone tenant id")
    name = models.CharField(unique=True, help_text="The Name of the Slice", max_length=80)
    enabled = models.BooleanField(default=True, help_text="Status for this Slice")
    omf_friendly = models.BooleanField()
    description = models.TextField(blank=True, help_text="High level description of the slice and expected activities", max_length=1024)
    slice_url = models.URLField(blank=True, max_length=512)
    site = models.ForeignKey(Site, related_name='slices', help_text="The Site this Node belongs too")
    tags = generic.GenericRelation(Tag)
    serviceClass = models.ForeignKey(ServiceClass, related_name = "slices", null=True, default=ServiceClass.get_default)
    creator = models.ForeignKey(User, related_name='slices', blank=True, null=True)
Lessons

• Part Analysis, Part Intuition
  – Whole is greater than the sum of its parts
• Unifying Abstractions
  – Duality is an opportunity
• Balance Requirements
  – Not about optimizing a single dimension
• Experience (Reality) Driven
  – Deploy It, Operationalize It, Use It
• Dynamicity (Evolution) is the Norm
  – Define Principles and Invariants
This slide intentionally left blank
Putting Lessons to Action

• Software Defined Networking (SDN)
  – Separating the Control and Data Planes

• Network Function Virtualization (NFV)
  – Data plane functions running in VMs on commodity servers

• Scalable Cloud Applications and Services (Apps)
  – Applications running on top of the network

Or... Finding the middle way for Open Networking Lab (ON.Lab) and the PlanetLab Consortium (PLC)
Distinctions without a Difference

• Three implementation points for “network functions”
  – SDN, NFV, Apps
• Blurring the SDN/Application Line
  – Is a proxy that cuts-through uninteresting flows a Controller?
  – Is a scalable Controller that uses a NoSQL DB an App?
  – Is a CDN that manages a caching hierarchy a Controller?
• Blurring the NFV/Application Line
  – Is a proxy an example of NFV or is it an application?
• Blurring the NFV/SDN Line
  – Is a firewall in the data plane or the control plane?
Topology

Virtual Topology (Big Switch)

Network Virtualization Layer
– Topology Isolation
– Address Space Isolation
– Semantic Isolation

Physical Topology
Topology Optimizations

Cut-Through

In-Line
Scaling Functions

Interesting question: How to partition functions into DC and edge “subroutines”?
Refactoring the Space

• Model all “network functions” as scalable services
  – Application vs Controller vs NFV distinction is arbitrary
• Use SDN to bootstrap a virtualization layer that...
  – Isolates virtual networks from each other
  – Maps virtual topology to physical topology
    • Maintains this mapping in the presence of failures, etc.
    • Tunnels vs OpenFlow is an implementation choice
    • Supports a cut-through optimization (service hint)
• NFV reduces to an implementation choice
  – Put function “in line” at the edge when appropriate
XaaS – Everything-as-a-Service

• Service as a Unifying Abstraction
  – Unifies across resources (Compute, Network, Storage)
  – Unifies across the network (DC, WAN, Access)
  – Unifies across service levels (IaaS, PaaS, SaaS)

• XOS – XaaS Operating System
  – Defines service as a first class object
  – Supports managing services, not servers
  – Supports seamless service extensions to XOS
  – Integrates service orchestration with resource provisioning
  – Supports both service isolation and service composition
Service Abstraction

• Provides a well-defined function
• Exports a programmatic (REST) interface
• Available network-wide (location independent)
• Scalable, elastic, and resilient
  – Scales with the number of users (self-balancing)
  – Seamlessly grows/shrinks based on demand
  – Built out of unreliable components (self-healing)
• Runs in a set of VMs connected by one or more VNs
• Build new services by composing with existing services
  – Some are building blocks (*NoSQL DB*), some are user-facing (*Facebook*), and some are both (*DropBox*)
Examples of Service Composition

• CoBlitz: Operator CDN (Now Akamai Aura)
  – HyperCache (HPC)
  – Request Router (RR)
  – Intercept Service (IS)

• Syndicate: Scalable Storage Service
  – Durability of Cloud Storage (S3, DropBox, Google Drive, Box)
  – Scalability of a CDN (HPC, RR)
  – Coherence of a Local FS (NoSQL DB – Google App Eng)

• Third: Scalable Monitoring & Analytics Service
  – Distributed data collection, analysis, and archiving
  – Leverages Storm, Cassandra, RabbitMQ and ZooKeeper
Service Isolation/Composition

e.g., “Content Acquisition” Network

Big Switch (Virtual Net)

Scalable Service “F”
XOS

Cloud Management System (CMS)

OSaaS (Syndicate)
CaaS (Nova)
NaaS (Quantum)
IMaaS (Keystone)
AaaS (Third)

XOS
(REST API + Data Model + Controller)

Node

OvS
Libvirt

Node

OvS
Libvirt
XOS Data Model

• Service runs in one or more Slices
  – Extend data model with service-specific objects
  – Define “shim” so programs can access service from VMs

• Slice is a resource container
  – Set of VMs + Set of VNs
  – Constraint-based VM placement
  – VMs added and deleted over time
  – VNs provide service isolation and composition

• Each VN is...
  – A big switch that fully connects all VMs in Slice
  – Private or Public (routable)
  – Closed or Open (available for multiple slices to join)
Operationalizing OpenStack

Policies, Configurations and Workflows that Codify Operational Practices* and Usage Models

OpenStack Components and Mechanisms (Nova, Quantum, Keystone, Glance...)

**OpenCloud Pilot – Hardware**

Cloud Data Centers

Regional PoPs

Operator Backbone

Edge Sites

End-Users

**ViCCI**
(5 SDN-Capable Data Centers)

**Internet2**
(SDN-Capable Backbone + ViNI)

**PlanetLab**
(500+ Sites, many with campus SDN)
OpenCloud Pilot – Software

Dashboard
REST API
Data Model
Quantum Keystone Nova

OpenCloud CMS

Effectively Defines XOS
– Codifies Operational Experience
– Explicit Support for XaaS

OpenVirteX

Node

OvS Libvirt

Node

OvS Libvirt
Status

• Near-term Development
  – Initial prototype of OpenCloud (XOS) running in the lab
  – Will deploy on operational system this fall
  – Deployment will include exemplar services
  – Integrating generalized Network Virtualization is next

• Longer term research questions
  – What are the right abstractions to support XaaS?
  – How do XaaS and Software Routers “meet in the middle”?
  – How is functionality best split between DC and the edge?
  – What is the performance impact of service composition?
Conclusions

I am indebted to many people, including...

- Tom Anderson
- Scott Baker
- Andy Bavier
- Sapan Bhatia
- Mic Bowman
- Brent Chun
- David Culler
- Bruce Davie
- Jim Dolce
- Serge Fdida
- Marc Fiuczynski
- John Hartman
- Mike Hluchyj
- Santosh Krishnan
- David Lowenthal
- Tony Mack
- Rick McGeer
- Nick McKeown
- Steve Muir
- Aki Nakao
- Jude Nelson
- Vivek Pai
- KyoungSoo Park
- Thierry Parmentelat
- Guru Parulkar
- Marcin Pilarski
- Patrick Richardson
- Timothy Roscoe
- Scott Shenker
- Stephen Soltesz
- David Tennenhouse
- Siobhan Tully
- Michal Wawrzoniak