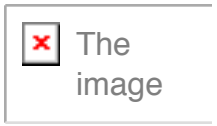


ANIMA and Intent

NMRG Workshop on Intent Based Networking

T. Eckert, Huawei (tte@cs.fau.de)

v1.5

Problem statement

- A few years ago, NRMG threw the notion of Intent over to ANIMA
  - And hoped we would be able to figure out how to standardize it
- We (ANIMA) where not able to put this on the WG charter because we where not sure what exactly we could do
- We (ANIMA) are now starting to re-charter to take on new work
  - Unfortunately, we can still not take on explicit work for Intent because we think we have no clear enough framework/proposals to make relevant progress for the ANIMA WG.
  - Want to write into re-charter that we would like to take on any work for Intent once we have a clear enough picture about what ANIMA could do

Summary

- No pressure on NMRG,… but:
  - There is a candidate customer of “Intent” output from NRMG (ANIMA) – and it would be great if NMRP Intent work could try to do Intent work that wold be sufficient for ANIMA to pick it up

Overview: From NMRG to ANIMA

- NMRG defined RFC7575/RFC7576 for **Autonomic Networks:**
- **Goal**: evolve networks to be built with self-X (configuring, healing, managing, optimizing, protecting)
- **Key building block: ASA** – Autonomic Service Agents. Distributed software modules embodying a distributed function/service on a node.
  - Managed by Intent (Q: what is Intent ?)
  - Leveraging a shared Autonomic Network Infra
- This was the seed to charter ANIMA
  - Bottoms up, starting with ANI

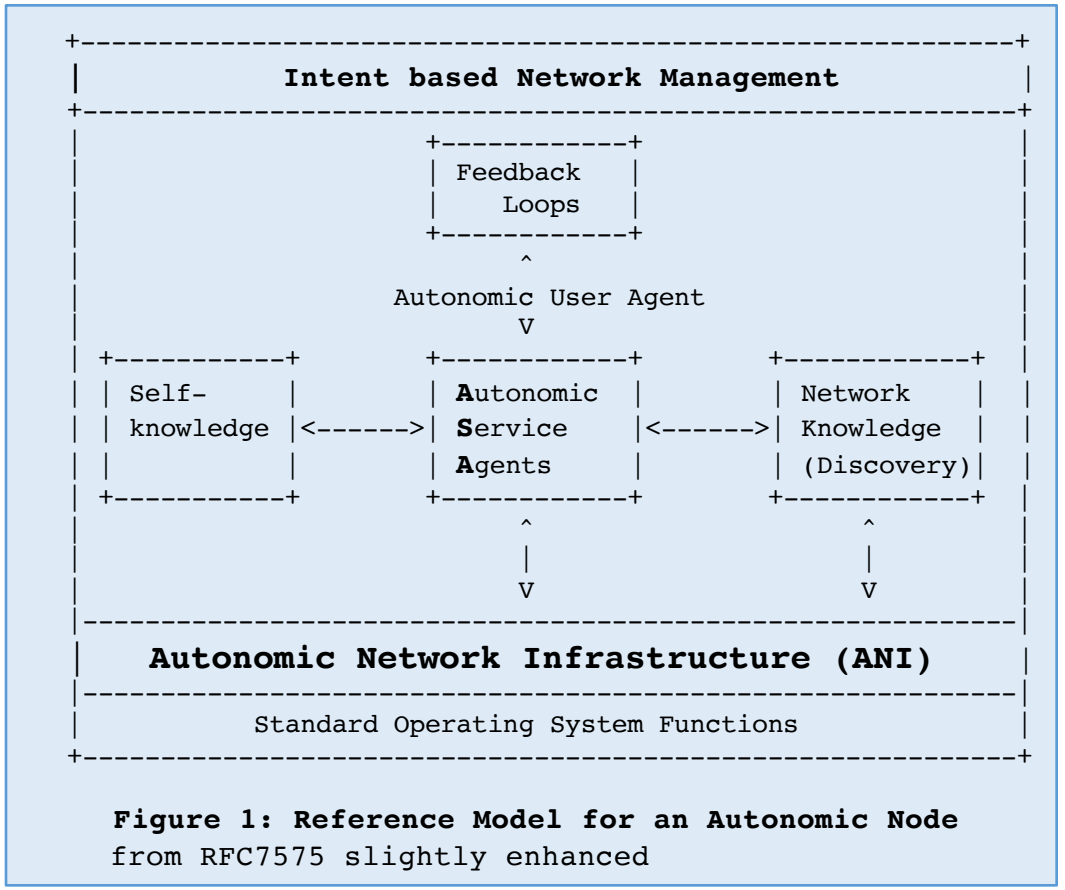


Figure 1: Reference Model for an Autonomic Node
from RFC7575 slightly enhanced

Overview: ANIMA now

- Charter of ANIMA until now:
- **Build ANI**
  - Details next slide
- **Define two example validation documents**

  To show applicability of ANI

  RFC8368 - use/benefits of ANI for classical centralized network management ("stable connectivity)

  draft-ietf-anima-prefix-management – automated prefix assignment for access interface via ANI (ACP/GRASP). First simple ASA. Prototype code:
  - https://github.com/becarpenter/graspy/blob/master/pfxm3.py
  - documented at
  - https://github.com/becarpenter/graspy/blob/master/pfxm3.pdf

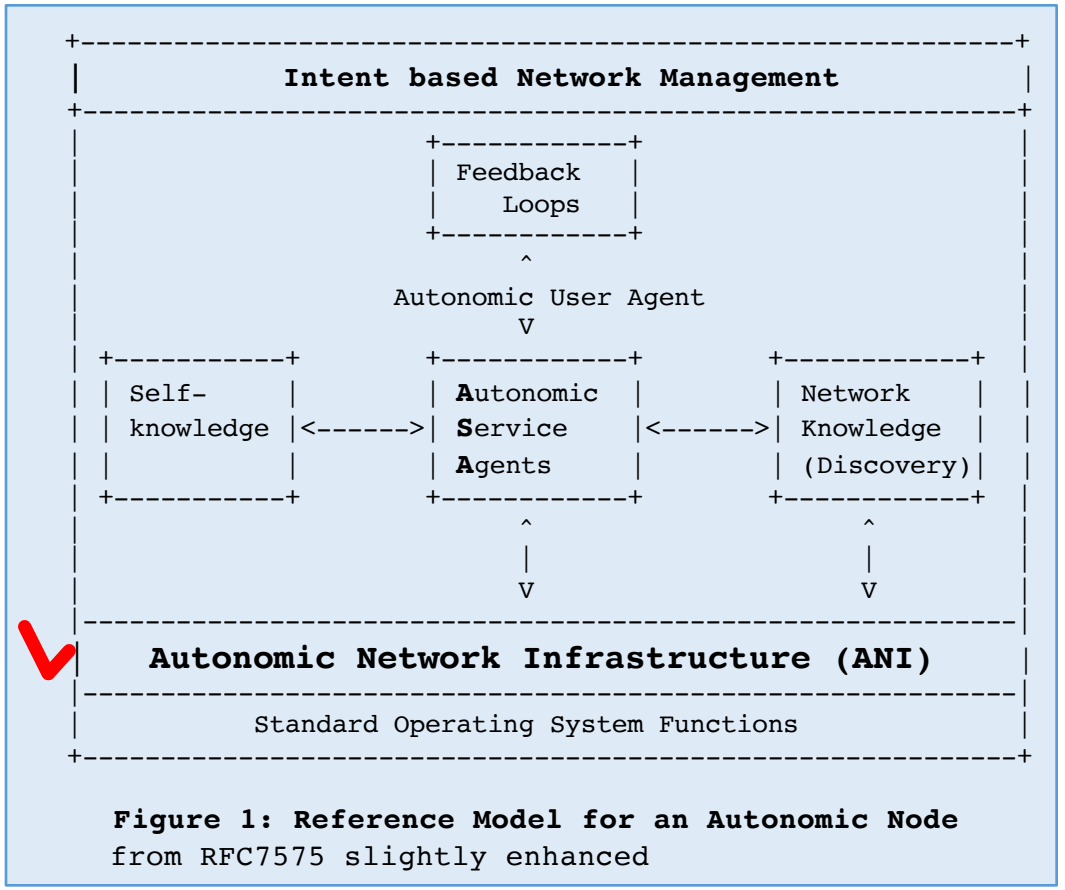
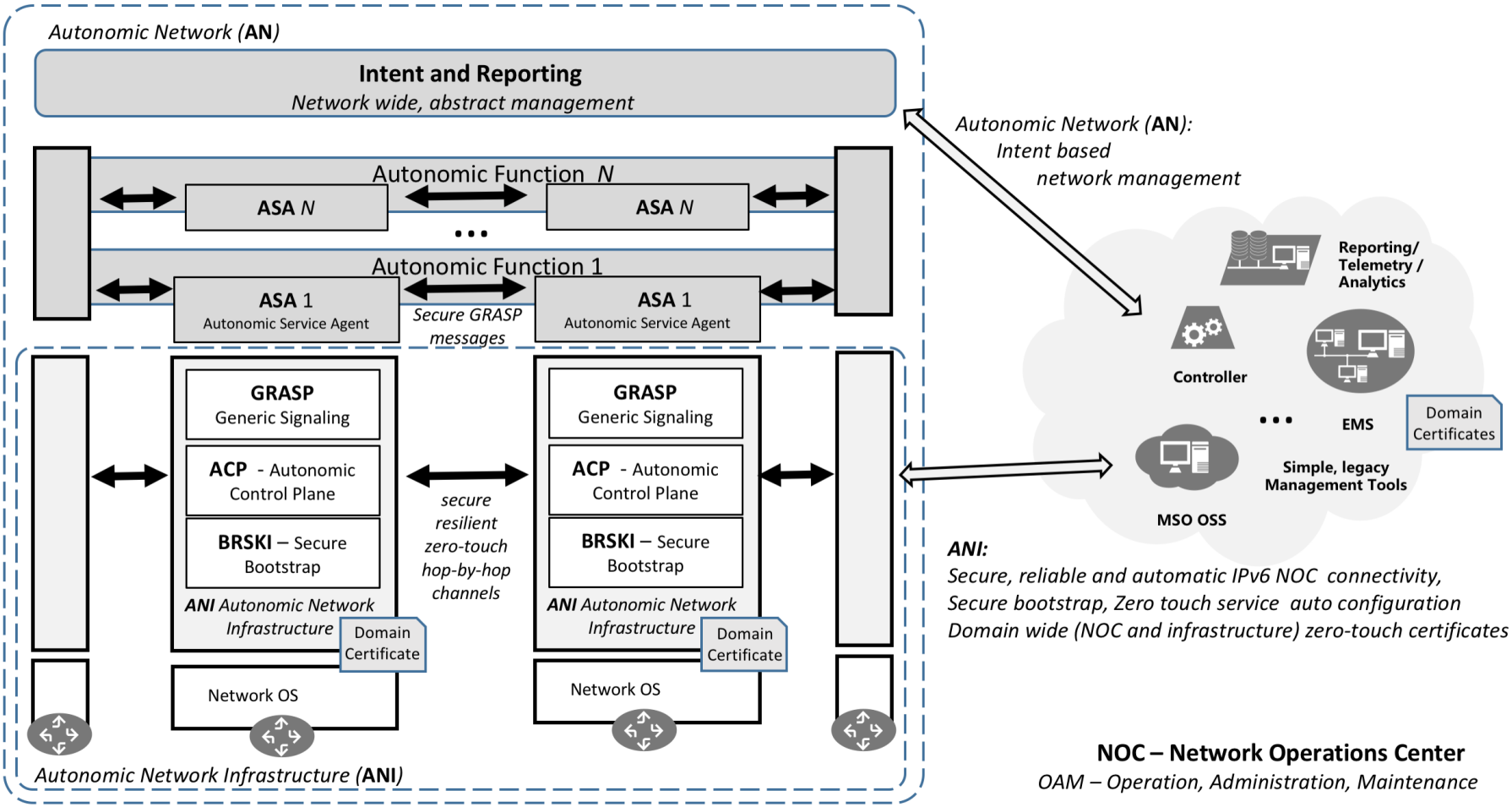


Figure 1: Reference Model for an Autonomic Node
from RFC7575 slightly enhanced

Autonomic Network according to ANIMA



Intent – data vs. system interpretation

- Data interpretation:
  - ANIMA (from NMRG) understands Intent as a set of data input into the network (could be expressed via Yang model or other domain specific language, declarative preferred)
- System/Processing interpretation
  - Other industry players use Intent to describe properties of an overall system, but do not apply the name to any specific set of data
    - These Intent based systems are always? Strongly centralized
    - And there may be good arguments to practically use centralized elements:
    - Many complex/NP-complete algorithms very hard to decentralize
    - And even if possible, is the benefit larger than the cost ?

Intent in ANIMA

- Non-agreement on what data is Intent in ANIMA
  A. “EVERYTHING” you send into the network
  B. NO!. For everything we already have a better word, we use that better word, and we use “Intent” only for stuff we do not understand:
     1. Service, Service-Instance Definitions (eg: L3VPN YANG service model RFC8299)
     2. Subscriber / Resource Policy Definitions
     3. …
     4. Intent – everything that is left

  B) Is frustrating: “Intent = God of the Gap”.

  B) Is even more frustrating if there is no agreed term for “EVERYTHING” (no Taxonomy): aka: how do you call the class of input to the network that includes all of 1.,2., 3.,4. ?

Simple ask

- ANIMA needs one term for EVERYTHING put into the network
  - This term could easily replace “Intent”. “Intent” could be a subset of it.

- ANIMA would should (IMHO) want to distinguish “EVERYTHING into few large buckets:
  A. EVERYTHING applying to more than a single node (network, role-wide)
  B. Everything more fine-grained
     - Operators will will still need to do more fine-grained interactions with the network, e.g.: for troubleshooting or operational workflows involving humans
       - Take interface smoothly out of network services, bring ip up into a test cycle once HW is fixed, then bring up fully operational

  With these two words for A, B we could replace “Intent” in ANIMA reference model and eliminate confusion about Intent (as Data Input into network)

draft-du-anima-an-intent

- Takes (undefined) intent (aka: A from previous slide)
- Floods it across network (e.g.: GRASP protocol)
- Nodes interpret it (e.g.: based on role)
- The act on that interpretation

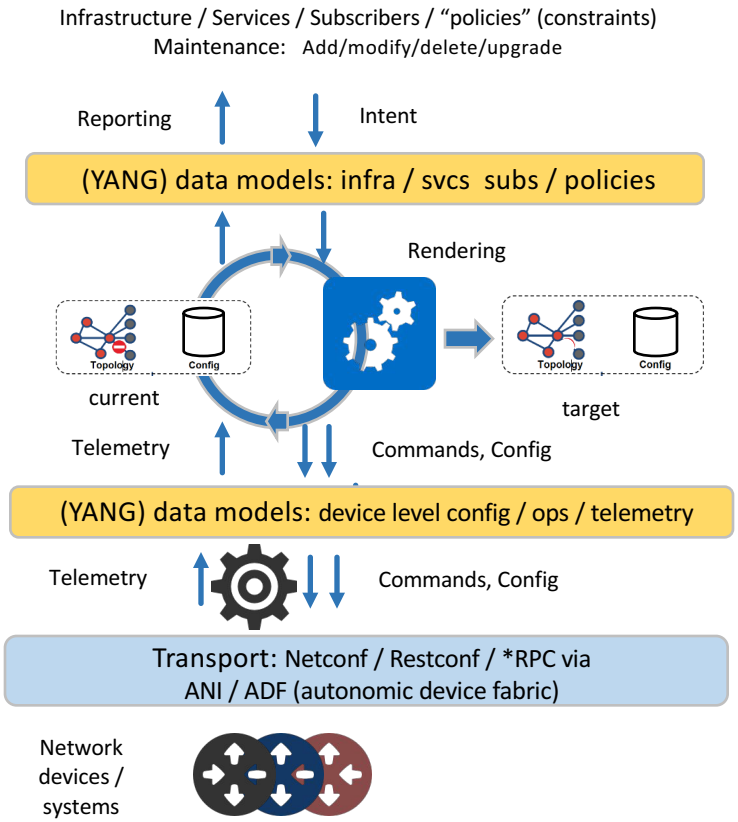
- Once we have an A that we can map to actual data that we know how to flood (e.g.: YANG model representation), we could go back to this

- Main issue IMHO:
  - Need to find use-cases where flooding of this information is quantitatively better than sending this information from an SDN controller individually to every node
  - Because this is really primarily about flooding vs. repeated unicast.
  - We have some non-ANIMA technology where we make exactly this claim, but I have not thought harder about how to make the argument for “A/Intent”

Distributed vs. Centralized Intent processing

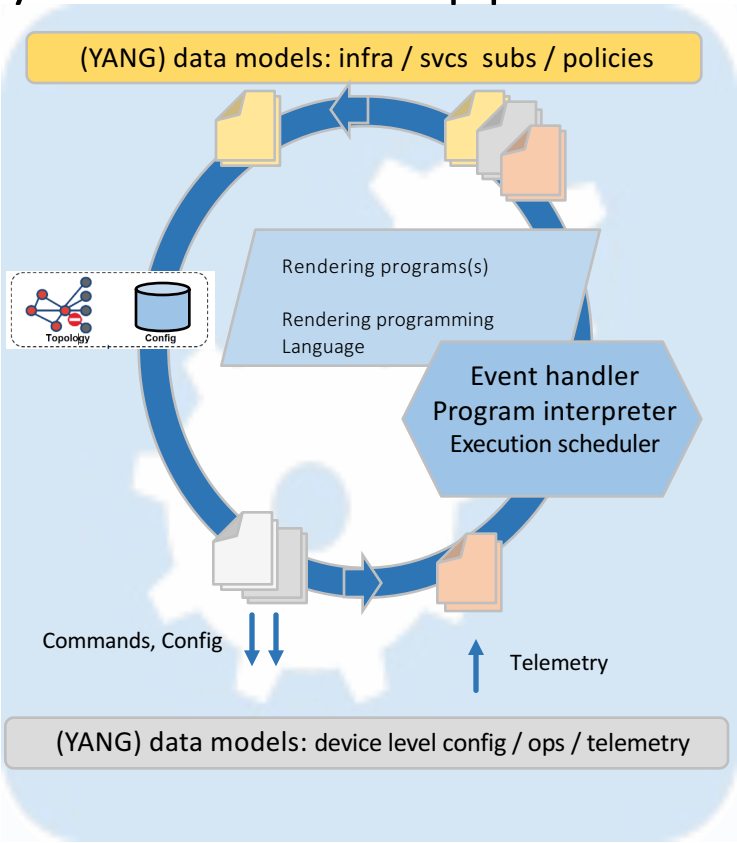
- draft-du-anima-an-intent is what I would called “distributed intent processing system”
- Would be great if NMRG would come up with a framework that explains that “Intent processing” can be centralized and/or distributed/decentralized
  - Hybrid in the general case. Based on specific requirements
  - IMHO very complementary. Should IMHO not try to fight for Intent processing to ONLY be one or the other

“Centralized” intent based operations: framework



- Communicate with operator / subscriber / ..
  via data-model defined interfaces
  - Reporting == from network to user, Intent == from user to network
  - Likely with some GUI tools on top – we ignore that piece
- Workflows =-= rendering intent into running network state
  - Continuously done, adopting to network change
  - Typically multi-step cycle of pushing config changes, validating them
    - Possible multi-step rollback / save state
    - Possible reporting of necessary operator action
  - Rendering can be multi-level / hierarchical (only one level shown on picture)
  - Rendering can involve intelligence (network brain)
    - Eg: traffic balancing / engineering
- Network device management/control
  - Ideally Device vendor independent (YANG) models
  - Reliable, secure, indestructible transport infrastructure for connectivity

“Centralized” intent based operations: Key innovation opportunities ?



- Automatic linkage of southbound data model to northbound data model (input)
  - Rendering result declaration
  - Dependency declarations
  - Operational state reporting
- Rendering Programming language
  - Optimized for simple, error-free programming of rendering
  - Parsing / expression of data models
  - Parsing / definition of graphs and attribution of graph
  - Automatic linkage
  - Simplified reporting
  - Declarative ?
    - Might allow for better static analysis, deferred, event-driven execution, backtracking, ..
    - Quite common in domain specific languages (Tensorflow, …)
- Rendering execution system
  - Automatic ? Backtracking
  - …

Thank You