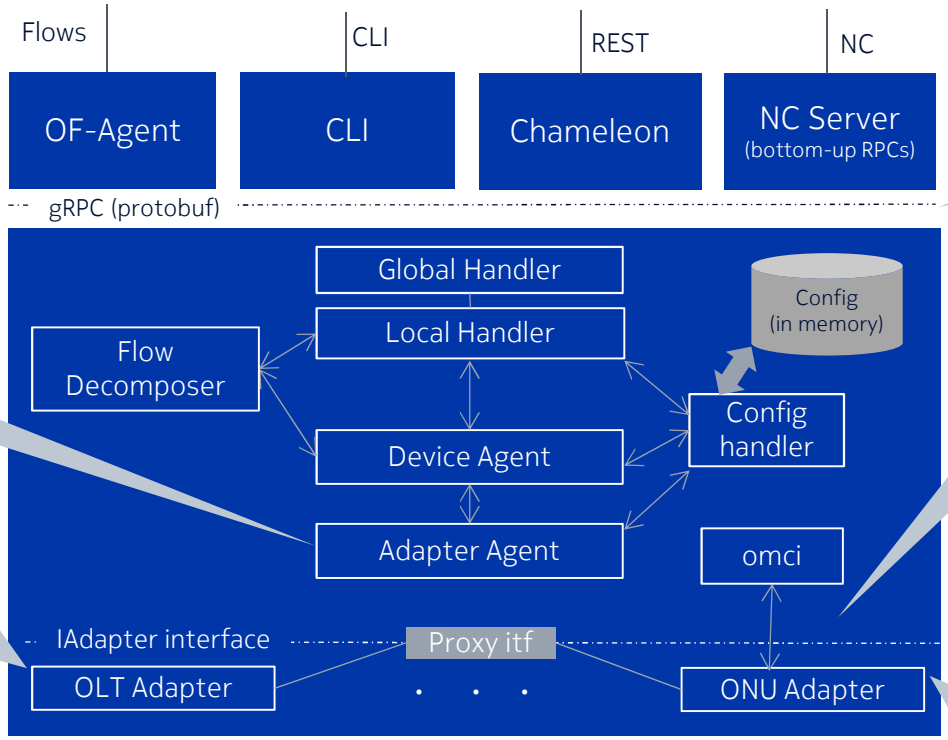
Proposal for xPON & ONU Provisioning in VOLTHA

TST Meeting

June 1 2017

Current Architecture
Hard-coded PON and ONU provisioning



- No support for ONU pre-provisioning from NBI
- No support for PON provisioning from NBI
- OLT device can be pre-configured
- OLT device-id is auto-generated (improvement required in future for NBI configurable ID/name)

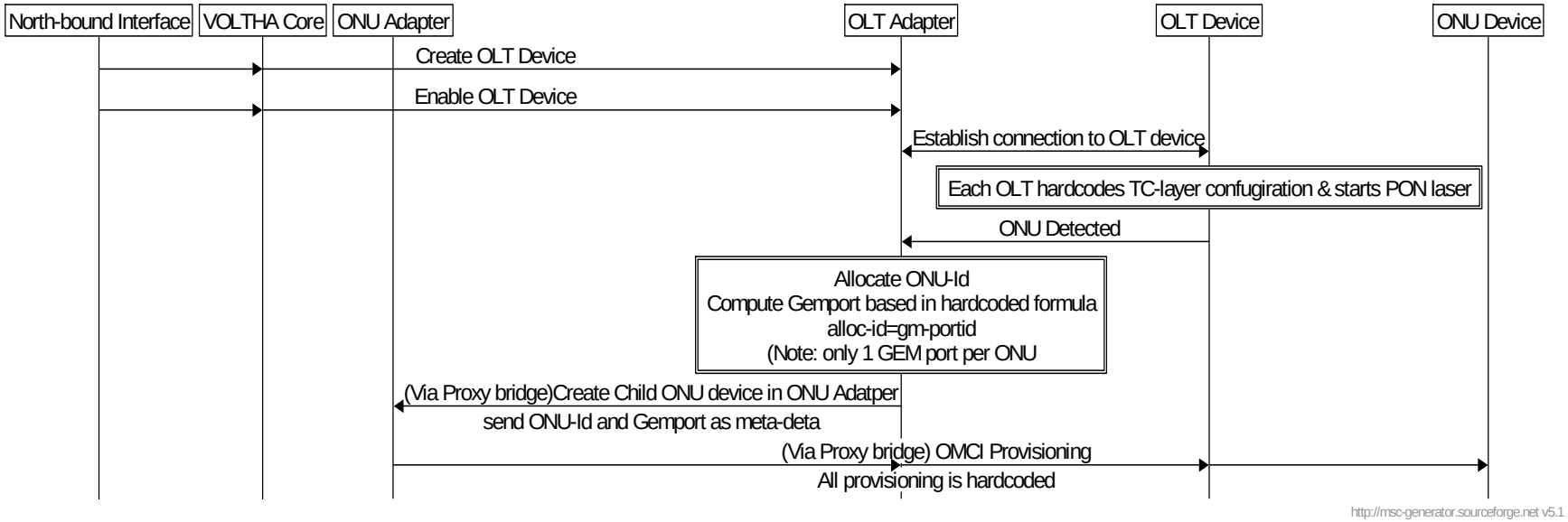
- Use to relay OMCI messages between the ONU and OLT adapters

- Supports configuration for flows
- No configuration support for PON related attributes
- Handles ONU-id assignment (based on a hard-coded formula
- UNI Port numbers are hard-coded
- Hardcoded GEM ports/alloc-ids

Direct interface into core

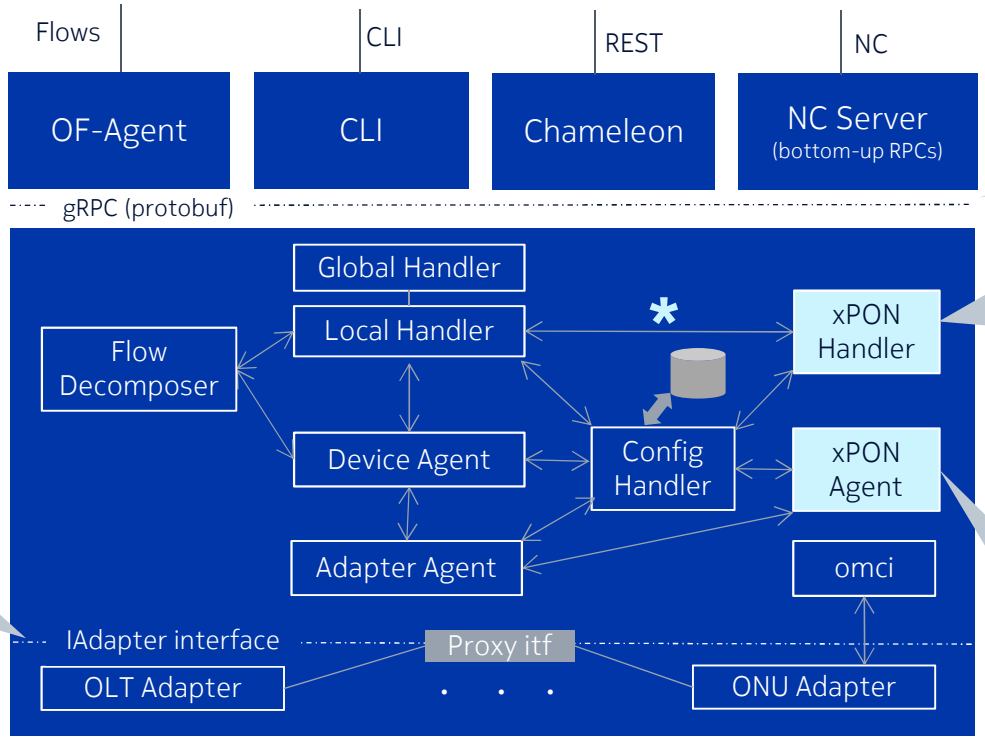
- ONU Device automatically created when ONU is detected.
- All the ONU attributes are hard-coded)
- No pre-configuration
- UNI port numbers are hard-coded
- Some ONU adapters (e.g BCM ONU) use hard-coded VLAN ids (same as in-port)

ONU detection & provisioning
Current Architecture



How to support PON and ONU provisioning
Introduction of xPON Handler & xPON Agent

* This interface could change when HA introduced (tbd)



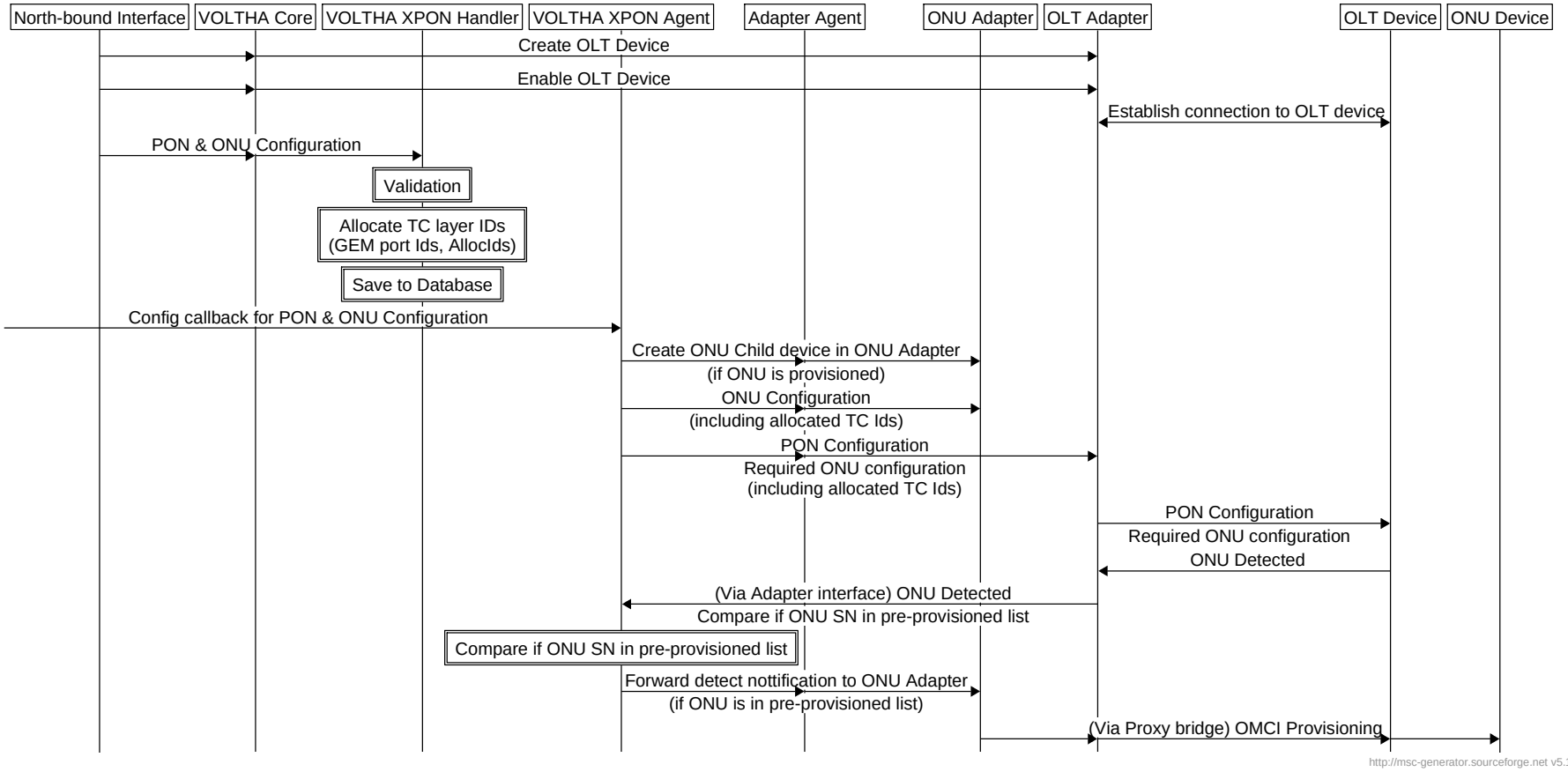
- Support PON and ONU provisioning via NBI
- Derive protos from WT-385 xPON models

- xPON Handler responsible for handling PON and ONU provisioning
- Responsible for validation and saving provisioning to database
- TC-Id allocation (i.e. Gem-port and Alloc-IDs)

- xPON Agent handles all the post-processing of PON and ONU provisioning following the database save
- Responsible for sending the configuration towards the adapters

- IAdapter interface extended for PON and ONU provisioning in a generic fashion

ONU Provisioning & Detection
Proposal



IAdapter Interfaces

Generic HW Abstraction for PON and ONU Provisioning

- Support for generic and common interfaces for PON and ONU Provisioning
  - For e.g. createInterface (...) as opposed to createPonInterface, createOnuInterface etc..
- Protos generated from WT-385 model are also used as interface protos towards the Adapters as much as possible
- The interface has 2 parameters
  - Device-id
  - Proto (message structure with relevant configuration information)

PON & ONU Provisioning
What is in progress?

- xPON Handler & xPON Agent Framework
- Subset of WT-385 objects
    - Channel-Group
    - Channel-partition
    - Channel-pair
    - Channel-termination
    - ONT-ANI & V-ONT-ANI (which includes SN & ONU-ID provisioning)
    - V-ENET
- Protos generated (to the best possible extent) from WT-385 model using yang2proto script
- Generic IAdapter interface for all of the above objects
    - CreateInterface(), UpdateInterface(), RemoveInterface()

PON & ONU Provisioning
Remaining required for VOLTHA 1.0!

- These remaining WT-385 objects would be required for VOLTHA 1.0
  - Traffic-Descriptor
  - GEM-Port
  - Alloc-Id

Note: Several other ONU provisioning parameters (e.g planned SW version, are a part of the entity/equipment domain and are not in scope for this presentation)

Open Items

- Interaction with HA for load-balancing of xPON configuration is still TBD (to be further discussed with HA team)
- Interaction with config restore is still TBD, i.e. what determines the order of the configuration restored (to be discussed further with team responsible for config/restore)
- Currently core only supports one UNI port per ONU. This restriction needs to removed
- OLT device needs an operator identifiable name or id
- ONU device is automatically created in xPON Agent currently. It could be independently created in future (similar to OLT Device)
- OMCI <-> ONU Adapter interface is not in the scope of this presentation. How is this interface supported when adapters are separated from core (still TBD)?
- Several other ONU provisioning parameters (e.g planned SW version, are a part of the entity/equipment domain and are not in scope for this presentation). These are generic entity-related provisioning and not specific to xPON