Assumptions for Guilin release

`This page captures the consolidated list of key assumptions.`

1. Common aspects

1. Association between NST and NSST shall be fixed at design time. In other words, an NST shall be composed of a set of NSSTs at design time.
2. As a result of (1), the AllocateNSSI API shall contain NSST info. This shall be re-visited beyond Guilin.
3. If the NSSMF is not capable of NSSI selection, ONAP acting as NSMF shall perform NSSI selection also and then pass the NSSI info in the AllocateNSSI API. This situation shall arise only when connecting to an external NSSMF, as all NSSMFs within ONAP shall support NSSI selection.
4. Sub-net capabilities are assumed to be pre-configured, i.e., no dynamic updates due to resource occupancy levels, etc.
5. When selecting an NSI or NSSI, resource occupancy levels shall not be taken into consideration when deciding to reuse an existing shareable NSI /NSSI (when the request allows sharing). The match with existing NSI/NSSI by OOF shall be made only on the attributes related to performance requirements (e.g., latency, throughput, etc.) and not based on attributes related to the slice/slice sub-net 'dimensions'.

2. Common NSSMF

1. The API from NSMF towards NSSMF shall be based on 3GPP APIs with the following exceptions:

    a. NSST id shall be passed in the AllocateNSSI API if the NSSMF is capable of NSSI selection. This is due to assumption (1) in "Common Aspects" - see above.
    b. NSSI id shall be passed in the AllocateNSSI API if the NSSMF is not capable of NSSI selection. This is to enable interoperability with external NSSMFs that aren't capable of NSSI selection (so ONAP acting as NSMF shall also do NSSI selection).
2. NSMF shall determine whether (external) NSSMF is capable of performing NSSI selection through a query to the NSSMF adaptor.

3. Core Slicing

1. No impact is foreseen for Closed Loop or Intelligent Slicing for updating the configuration of the core NSSI.
2. One Core NSST shall always be associated with one NSD.

4. RAN Slicing

1. RAN NFs are assumed to be instantiated apriori. So in case of RAN NSSI creation, the step involving RAN NF instantiation shall be skipped.
2. RAN NSST will be a nested service composing RAN-NF NSST, TN-FH and TN-MH NSSTs. This shall be revisited beyond Guilin

5. Transport Slicing

1. FH and MH NSST and Slice Profile attributes shall be simply the same as BH attributes for G-release.
2. Transport NSSMF shall always be invoked via the NSSMF adaptor (whether it is called by NSMF or RAN NSSMF).
3. The endpoints for connecting RAN and Core (i.e., BH) shall be configured via UUI and provided to TN NSSMF. The endpoints for FH and MH are assumed to be not needed (i.e., pre-configured) because RAN NFs are already instantiated in the preparation phase.

6. RAN & Transport Slicing interactions

- **Option 1** = RAN NSSI shall comprise of RAN NF NSSI, FH NSSI and MH NSSI. RAN NSSMF shall be responsible for RAN NSSI and RAN NF NSSI orchestration, and shall trigger Transport NSSMF for FH and MH NSSI orchestration. NSI shall comprise of RAN NSSI, TN (BH) NSSI and Core NSSI. Internals of RAN NSSI are "don't care" for NSMF (to keep the implementation simple, and common for both Option 1 and Option 2).
- **Option 2** = In G-release, external RAN NSSMF will be invoked with RAN Slice profile. Internals of RAN NSSI are "don't care" for NSMF (to keep the implementation simple, and common for both Option 1 and Option 2). This means, FH and MH are abstracted out for Option 2 in G-release.

(Option 2 beyond G-release: RAN NSSI shall denote just RAN NF NSSI. NSMF shall be responsible for triggering TN NSSMF for FH, MH and BH NSSI orchestration. NSI shall comprise of RAN (NF) NSSI, FH NSSI, MH NSSI, BH NSSI and Core NSSI.)

1. Both options for RAN and Transport Slicing interaction shall be supported in Guilin release. RAN NSSMF within ONAP shall support Option 1, while external RAN NSSMF shall support Option 2. NSMF within ONAP shall support both options.
2. In a single ONAP deployment, only one of the 2 options shall be supported, and shall be known apriori. As a consequence, the NSTs also shall reflect only the option that is supported in the deployment. For Guilin, this doesn't really make a difference (based on above assumptions), however, beyond G-release, for example, if Option 1 is supported, NST shall comprise of RAN, Core and TN (BH) NSSTs, while when Option 2 is supported, NST shall comprise of RAN, Core, TN (FH), TN (MH) and TN (BH) NSSTs.
3. SO (NSMF) in G-release shall not be really concerned with which Option is supported. Beyond G-release, SO (NSMF) shall know which Option is supported by examining the NST (Option 1 => NST contains 3 NSSTs, Option 2 => NST contains 5 NSSTs). OOF shall operate based on the NST contents in a model-driven way.

7. KPI Monitoring

1. No trigger for KPI monitoring shall be sent from SO or other micro-services in DCAE. The KPIs to be computed by PM Mapper MS shall be configured via Policy, and the NSI/NSSI for which the KPIs shall be computed shall be determined by accessing AAI before the computation.
2. It is assumed that the NFs already "know" which PM data should be reported to ONAP.

8. Closed Loop

1. Any resource re-allocation for slice resource optimization shall be computed based on rules by SDN-R for the RAN (without involving OOF).
2. The PM data will be posted via VES (and not the DFC-->PM-mapperDMaaP route). This is because of slice-specific info such as S-NSSAI which have to be interpreted by PM-mapper, which will be done beyond Guilin.

9. Intelligent Slicing

1. The training shall happen outside ONAP, and the ML models shall be onboarded as an MS in DCAE. This MS shall not be part of ONAP repos, and shall be used only for PoCs/demos.
2. The PM data will be posted via VES (and not the DFC-->PM-mapperDMaaP route). This is because of slice-specific info such as S-NSSAI which have to be interpreted by PM-mapper, which will be done beyond Guilin.