← Overview

OmniNGAP Operations

OmniNGAP is the **NGAP protocol codec** for the Omnitouch 5G Core. NGAP (NG Application Protocol, 3GPP TS 38.413) is the application-layer signalling protocol carried over the **N2 interface** between a 5G NR base station (gNB) and the Access and Mobility Management Function (AMF). OmniNGAP compiles the complete NGAP ASN.1 syntax from TS 38.413 and provides encode and decode of NGAP Protocol Data Units (PDUs) using **Aligned Packet Encoding Rules (APER)**, the wire format mandated by the specification.

OmniNGAP is consumed as a build dependency:

- **OmniAMF** links OmniNGAP to encode and decode every message it exchanges with a gNB over N2 (NG Setup, Initial Context Setup, NAS transport, Paging, PDU Session Resource management, UE Context Release, Handover, and so on).
- **OmniSMF** produces the **N2 SM information containers** (for example the *PDU Session Resource Setup Request Transfer*) that the AMF embeds inside NGAP messages before they are encoded. The SMF never speaks N2 directly; it hands these opaque transfer containers to the AMF, which is the sole NGAP peer of the gNB.

How the codec is built

The NGAP syntax is expressed as a set of ASN.1 modules under `asn1/`. At build time these modules are compiled once, in dependency order, into the low-level codec that performs APER encode and decode. This compilation is a build concern only - there is nothing to configure or tune at run time. The modules that make up the NGAP syntax are:

ASN.1 module	Role
NGAP-CommonDataTypes	Base types shared across the syntax (Criticality, Presence, ProcedureCode, ProtocolIE-ID)
NGAP-Constants	Procedure code, IE identifier, and cardinality constant assignments
NGAP-Containers	Generic Protocol IE / Protocol Extension container templates
NGAP-IEs	All NGAP Information Element definitions, including the N2 SM transfer containers
NGAP-PDU-Contents	Per-procedure message bodies (InitiatingMessage / SuccessfulOutcome / UnsuccessfulOutcome payloads)
NGAP-PDU-Descriptions	Top-level NGAP-PDU type and the elementary-procedure dispatch table

APER (Aligned PER) is used deliberately: NGAP requires aligned encoding per TS 38.413, so the codec is generated in aligned-PER mode rather than unaligned PER or BER.

PDU shape: the three message variants

Every NGAP-PDU is a CHOICE of exactly one of three variants (TS 38.413 clause 9.3.5). A decode therefore always yields one of these, and the AMF pattern-matches on the variant to decide how to proceed:

Variant	When it appears	Meaning
`initiatingMessage`	Opens an elementary procedure	The first message of a procedure (a request, a report, or a one-shot indication)
`successfulOutcome`	Closes a procedure that succeeded	The positive response to an initiating message (for procedures that expect a response)
`unsuccessfulOutcome`	Closes a procedure that failed	The negative response, carrying a **Cause** (see the Cause reference below)

Each variant is a SEQUENCE of three fields: the **procedureCode** (which elementary procedure this is - see the code table below), the **criticality** of the message, and the **value** (the procedure-specific message body). Class-2 procedures (fire-and-forget, such as Paging, DownlinkNASTransport, ErrorIndication) have no successful/unsuccessful outcome - they consist of an `initiatingMessage` only. Class-1 procedures (such as NGSetup or InitialContextSetup) have one or both outcomes.

Criticality and Criticality Diagnostics

Every IE, and every message, carries a **criticality** drawn from `Criticality ::= ENUMERATED { reject, ignore, notify }` (TS 38.413 clause 9.3.1). Criticality tells a receiver what to do when it encounters an IE (or a whole message) whose ID or type it does not understand, or that it cannot handle in its current state:

Criticality	Receiver behaviour on a not-understood / not-handled item
`reject`	Reject the whole procedure - the item is essential, so the message cannot be processed
`ignore`	Silently skip the item and process the rest of the message
`notify`	Process the message without the item, but report the item back to the sender in **Criticality Diagnostics**

Criticality Diagnostics (ProtocolIE-ID 19) is the IE the receiver adds to its response or Error Indication to tell the sender exactly which IEs were rejected, ignored, or notified, and why. It is how the two ends reconcile version or capability mismatches without dropping the association.

OmniNGAP encodes and decodes the criticality fields and the Criticality Diagnostics IE like any other part of the PDU. It does **not** apply the reject/ignore/notify policy itself - deciding what to do with an unknown or mishandled IE, and building the resulting Criticality Diagnostics, is procedure logic that lives in the AMF.

3GPP Role and Specification References

Specification	Relevance
TS 38.413	NGAP protocol - message set, Information Elements, procedure codes, cause values, APER encoding
TS 38.412	N2 signalling transport - carriage of NGAP over SCTP
TS 23.502	5G system procedures that run over N2 (registration, PDU session establishment/modification/release, paging, handover)
TS 23.501	5G system architecture - position of the N2 reference point between gNB and AMF
ITU-T X.691	Packet Encoding Rules (PER), the encoding family NGAP uses in aligned form

The compiled ASN.1 corresponds to the **Rel-17 (h-series)** NGAP syntax of TS 38.413, so the full Rel-17 message set - including UE Context Resume/Suspend, RAN early status transfer, and MBS (multicast/broadcast) procedures - is present in the compiled codec.

Supported NGAP Procedures

OmniNGAP compiles the **entire NGAP elementary-procedure set** defined by TS 38.413. Any well-formed NGAP PDU for any of these procedures can be decoded, and any structurally valid term can be encoded - the codec does not restrict itself to a subset. What is exercised in practice is driven by the consuming NF (principally the AMF).

The table below lists the procedures most relevant to AMF/SMF operation, with their **actual** procedure codes as assigned in the compiled `NGAP-Constants` module. Direction is the direction of the *initiating* message.

Procedure	Code	Direction	Purpose
NGSetup	21	gNB → AMF	Establish the NG (N2) association; exchange gNB/AMF identities, PLMN and slice support
RANConfigurationUpdate	35	gNB → AMF	gNB updates its configuration on an established association
AMFConfigurationUpdate	0	AMF → gNB	AMF updates its configuration / TNL association info
NGReset	20	either	Reset part or all of the NG association
InitialUEMessage	15	gNB → AMF	First NAS message from a UE, opening a UE-associated signalling connection

Procedure	Code	Direction	Purpose
DownlinkNASTransport	4	AMF → gNB	Carry a NAS PDU toward the UE
UplinkNASTransport	46	gNB → AMF	Carry a NAS PDU from the UE
InitialContextSetup	14	AMF → gNB	Establish UE context, security, and (optionally) PDU session resources at the gNB
UEContextModification	40	AMF → gNB	Modify an established UE context
UEContextReleaseRequest	42	gNB → AMF	gNB requests release of the UE-associated connection
UEContextRelease	41	AMF → gNB	AMF commands release of the UE context (Command / Complete)
Paging	24	AMF → gNB	Page an idle-mode UE in a tracking area

Procedure	Code	Direction	Purpose
PDUSessionResourceSetup	29	AMF → gNB	Set up PDU session resources at the gNB (carries SMF N2 SM container)
PDUSessionResourceModify	26	AMF → gNB	Modify existing PDU session resources
PDUSessionResourceModifyIndication	27	gNB → AMF	gNB-initiated PDU session resource modification
PDUSessionResourceRelease	28	AMF → gNB	Release PDU session resources at the gNB
PDUSessionResourceNotify	30	gNB → AMF	Notify QoS flow / session status change
HandoverPreparation	12	Source gNB → AMF	NG-based handover: source gNB requests handover preparation

Procedure	Code	Direction	Purpose
HandoverResourceAllocation	13	AMF → Target gNB	NG-based handover: request resources at the target gNB
HandoverNotification	11	Target gNB → AMF	Target gNB confirms UE arrival
HandoverCancel	10	Source gNB → AMF	Cancel an in-progress handover
PathSwitchRequest	25	gNB → AMF	Xn-based handover: switch the N3 user-plane path to the target gNB
ErrorIndication	9	either	Report a protocol error not tied to a specific procedure
UEContextResume	58	gNB → AMF	Resume a suspended (RRC-INACTIVE) UE context

Procedure	Code	Direction	Purpose
UEContextSuspend	59	gNB → AMF	Suspend a UE context

> The other TS 38.413 procedures are all present in the compiled syntax and can be encoded/decoded, but are not part of the AMF's core connection-management and session-management flows. These include (this list is illustrative, not exhaustive) NRPPa transport, trace management, warning message broadcast (PWS), location reporting, RIM information transfer, secondary RAT usage reporting, the MBS broadcast/multicast procedures, overload start/stop, AMF status indication, NAS reroute and non-delivery indication, RRC inactive transition reporting, RAN and RAN-early status transfer, and private message. The compiled syntax defines the full elementary-procedure set of the release, so any procedure in TS 38.413 Rel-17 can be coded even when it is not listed here.

Key NGAP Flows

The following sequences show where OmniNGAP performs encode/decode. The codec is the translation layer at the AMF's N2 edge; the procedure logic lives in the AMF (and, for SM containers, the SMF).

NG Setup

Association bring-up, run once per gNB–AMF association before any UE traffic.

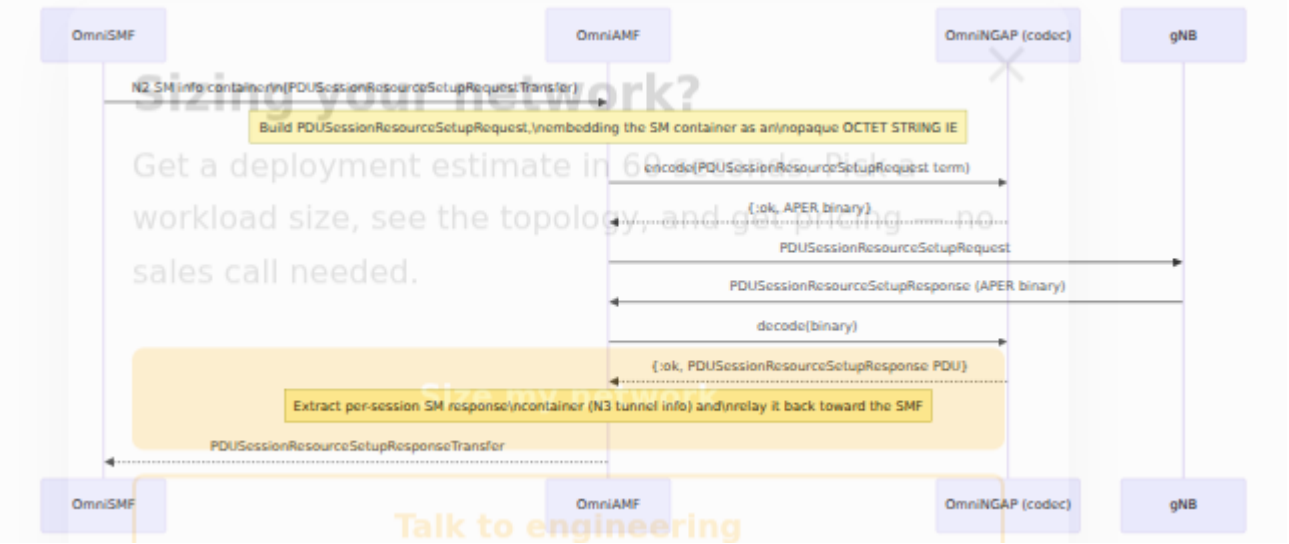
Parse error on line 13: ...signalling may begin ----------^ Expecting 'SOLID_OPEN_ARROW', 'DOTTED_OPEN_ARROW', 'SOLID_ARROW', 'BIDIRECTIONAL_SOLID_ARROW', 'DOTTED_ARROW', 'BIDIRECTIONAL_DOTTED_ARROW', 'SOLID_CROSS', 'DOTTED_CROSS', 'SOLID_POINT', 'DOTTED_POINT', got 'NEWLINE'

Try again

If the AMF cannot serve the gNB (for example an unsupported PLMN or slice), it instead encodes an **NGSetupFailure** carrying a **Cause** value (see Cause reference below).

PDU Session Resource Setup over N2

Runs during PDU session establishment. The SMF builds the N2 SM information container; the AMF wraps it in an NGAP message and OmniNGAP encodes the whole PDU.



The codec treats each SM information container as a nested transfer structure inside the PDU Session Resource message. The AMF does not interpret the SM container contents - it relays them between the gNB and the SMF unchanged.

Opaque payloads: NAS PDUs and SM containers

Two distinct payload families cross the N2 interface without NGAP looking inside them, and both are modelled in the ASN.1 as `OCTET STRING`:

- **NAS PDUs** - a NAS message (registration, authentication, PDU session establishment request, and so on) is carried inside NGAP as an opaque `NAS-PDU` OCTET STRING. It appears in the NAS-transport procedures (InitialUEMessage, DownlinkNASTransport, UplinkNASTransport) and can also ride inside InitialContextSetup and PDU-session messages. NGAP decodes it to the raw bytes; the AMF passes those bytes to its NAS layer (or relays them to the UE) - NGAP never parses NAS.

- **SM information containers** - the SMF-built PDU-session transfer structures described above, likewise carried as OCTET STRINGs between the gNB and the SMF.

The practical consequence is the same for both: OmniNGAP guarantees the *enclosing* PDU is well-formed and hands back (or transmits) the inner bytes verbatim. Whether those inner bytes are a valid NAS message or a valid SM container is the concern of the NAS layer and the SMF respectively, never of the NGAP codec. This is why a fault inside a NAS PDU or SM container never shows up as an NGAP encode/decode error.

NGAP Cause Values

Every NGAP failure or release message carries a **Cause**, a CHOICE selecting one of five cause groups defined in TS 38.413 (clause 9.3.1.2). OmniNGAP encodes and decodes all five groups and their enumerated values. The `Cause` CHOICE is:

Cause group	Meaning	Typical use
radioNetwork	Radio-network layer reason	Handover outcomes, radio link loss, user inactivity, resource unavailability, slice-not-supported
transport	Transport-network layer reason	Transport resource unavailable
nas	NAS layer reason	Normal release, authentication failure, deregistration
protocol	NGAP protocol error	Abstract/transfer syntax errors, message-not-compatible-with-receiver-state
misc	Miscellaneous reason	Overload, hardware failure, O&M intervention, unknown PLMN/SNPN

Representative enumerated values within each group (all per TS 38.413 clause 9.3.1.2):

Group	Value	Meaning
radioNetwork	successful-handover	Handover completed successfully
radioNetwork	handover-cancelled	Handover aborted by the source side
radioNetwork	user-inactivity	Connection released due to UE inactivity
radioNetwork	radio-connection-with-ue-lost	Radio link to the UE was lost
radioNetwork	radio-resources-not-available	gNB has no radio resources for the request
radioNetwork	unknown-PDU-session-ID	Referenced PDU session is unknown at the peer
radioNetwork	slice-not-supported	Requested S-NSSAI is not supported
radioNetwork	ims-voice-eps-fallback-or-rat-fallback-triggered	Fallback to EPS/other RAT for IMS voice
radioNetwork	release-due-to-ngran-generated-reason	NG-RAN-initiated release
radioNetwork	release-due-to-5gc-generated-reason	5GC-initiated release
transport	transport-resource-unavailable	No transport (N3/TNL) resources available
transport	unspecified	Unspecified transport-layer failure

Group	Value	Meaning
nas	normal-release	Normal NAS-triggered release
nas	authentication-failure	UE authentication failed
nas	deregister	UE deregistration
protocol	transfer-syntax-error	PDU could not be decoded (bad transfer syntax)
protocol	abstract-syntax-error-reject	Semantically invalid PDU, rejected
protocol	message-not-compatible-with-receiver-state	Message invalid for the peer's current state
protocol	semantic-error	Semantic error in an otherwise decodable PDU
misc	control-processing-overload	Peer is overloaded
misc	hardware-failure	Peer hardware failure
misc	om-intervention	Operations & maintenance intervention
misc	unknown-PLMN-or-SNPN	PLMN/SNPN not recognised by the peer
misc	unspecified	Unspecified miscellaneous failure

The radioNetwork group is the largest and includes the full Rel-17 set (handover-timer expiries, NPN/CAG access-denied, RedCap-not-supported, MBS-

related causes, and more). The values above are the ones most commonly seen in AMF connection- and session-management flows.

Integration

How the AMF uses OmniNGAP

- **Inbound:** SCTP payloads received on the N2 association are passed to the codec's decode entry point, which returns a structured NGAP PDU (one of the three message variants - `initiatingMessage`, `successfulOutcome`, or `unsuccessfulOutcome`) or a tagged error. Malformed or truncated PDUs return an error rather than crashing the AMF, and are logged so a bad peer message is never silent.
- **Outbound:** The AMF builds the PDU term for the procedure it is running, then calls the codec's encode entry point to obtain the aligned-PER encoding written to the SCTP association. The encoder returns the result as iodata (an iolist), which the caller flattens to a binary before transmission.
- The AMF owns all procedure semantics - IE population, optionality, and criticality handling. OmniNGAP only guarantees that a structurally valid term encodes to a spec-conformant binary and that a spec-conformant binary decodes to the corresponding term.

How the SMF uses OmniNGAP

The SMF does not call the codec directly for N2 transport. It constructs the **N2 SM information transfer containers** (for example *PDU Session Resource Setup Request Transfer*, *… Modify Request Transfer*, *… Release Command Transfer*) that the AMF embeds inside the corresponding PDU Session Resource NGAP message. Those containers are carried as opaque OCTET STRINGs at the NGAP layer and encoded/decoded as part of the enclosing PDU by the AMF's use of OmniNGAP.

← Overview

OmniNGAP Troubleshooting

Because OmniNGAP is a compile-time codec library with no runtime surface, there is nothing to configure, restart, or scrape. Its observable behaviour is confined to how it treats the binaries handed to it by the consuming NF (the AMF). Signalling-level observability of N2 comes from the AMF, not from this library.

Decode of malformed or truncated PDUs

Malformed or truncated PDUs return a tagged error rather than crashing the consuming NF. When a gNB sends a message the codec cannot decode:

- The decode entry point returns a tagged error instead of raising, so a bad peer message never takes down the AMF.
- The AMF logs the failure, so a bad peer message is never silent.
- The corresponding NGAP-layer signal to the peer is a **Cause** of the `protocol` group - for example `transfer-syntax-error` (the PDU could not be decoded) or `abstract-syntax-error-reject` (the PDU was decodable but semantically invalid). See the NGAP Cause Values reference for the full set.

Decode is anchored to the first PDU in the buffer: once a complete, well-formed PDU has been read, any trailing bytes after it are ignored rather than rejected. NGAP over SCTP delivers one PDU per message, so trailing data is not expected in normal operation; a decode that succeeds while a peer believes it sent additional content points to a framing fault in the peer or the SCTP layer, not in the codec.

Encode failures

OmniNGAP guarantees only that a *structurally valid* term encodes to a spec-conformant binary. IE population, optionality, and criticality are the responsibility of the AMF. An encode error therefore points at the term the AMF built (a missing mandatory IE, a value out of its constrained range, or a malformed nested SM container) rather than at the codec itself.

Opaque payload issues (NAS PDUs and SM containers)

NGAP carries two kinds of payload as opaque OCTET STRINGs without inspecting them: NAS PDUs and SMF-produced N2 SM information containers. OmniNGAP does not interpret either.

- **SM containers** - if a session-management transfer appears corrupt end-to-end but the enclosing NGAP PDU encodes and decodes cleanly, the fault lies in how the SMF built the container or how the AMF placed it, not in the NGAP codec.
- **NAS PDUs** - likewise, if a NAS message is malformed but the enclosing NGAP PDU (InitialUEMessage, DownlinkNASTransport, UplinkNASTransport, and so on) is well-formed, the fault is in the NAS layer, not in NGAP.

The general rule: a clean encode/decode of the enclosing PDU means the NGAP layer did its job - look inside the opaque payload's owner (the NAS layer or the SMF) for the corruption.