

XyDromatics Encounter Engine — HL7 v2 / FHIR Conformance Statement

Document number: DOC-2026-408 **Title:** XyDromatics Encounter Engine — HL7 v2 / FHIR Conformance Statement **Version:** v5 **Category:** DICOM Conformance Statement **Product:** XyDromatics Encounter Engine **Release version:** 1.0.1.109 **Classification:** Non-device software (FD&C Act §520(o)(1)(D)) **Owner:** Synthology Healthcare Solutions Group (SHSG) — Regulatory Affairs / Product Engineering **Effective date:** (on approval) **Review cycle:** 1 year **Next review due:** TBD **Audience:** HIS/EMR integration teams, ADT-feed administrators, RIS interface engineers, and EMR order-management staff wiring the Encounter Engine into a clinical messaging environment.

HL7 v2.x message semantics in this statement follow HL7 v2.3 / v2.5 field layouts as implemented in product source. FHIR semantics follow HL7 FHIR R4. This document describes ONLY message types, segments, resources, and interactions that are implemented in the shipping source tree. Anything not implemented is explicitly marked “**not supported**” or “**planned**”.

1. Introduction

1.1 Revision History

Version	Date	Change
Draft	2026-06-26	Initial draft. Grounded against Encounter Engine source v1.0.0.87.

1.2 Product role and regulatory posture

The XyDromatics Encounter Engine is **software only — a NON-DEVICE under FD&C Act §520(o)(1)(D)**. Its regulatory class is “*Non-device software (FD&C Act §520(o)(1)(D))*”. It is not a medical device, and is neither a current nor a pending device. This conformance statement makes no diagnostic or interpretive claim; the Encounter Engine moves and reconciles patient-identity and order metadata, it does not interpret clinical content.

Functionally, the Encounter Engine is a **patient-identity reconciliation and order/result messaging adjunct** that sits beside a DICOM ingest path. Its HL7 v2 and FHIR surfaces serve three jobs:

1. **Inbound ADT feed** — consume an HL7 v2.x ADT stream over MLLP to populate a short-retention identity cache used to *propose* the correct patient identity for an incoming study (human-confirmed; never auto-committed by the ADT path).
2. **Outbound order/result forwarding** — after an operator verifies a study, emit the verified identity downstream as either an HL7 v2 ORM^O01 (over MLLP) or a FHIR R4 ServiceRequest (over HTTPS), as license-gated forward destinations.

3. **EMR order round-trip (Phase 1, default-OFF)** — mint a Placer Order Number, send an HL7 v2 ORM^O01 order request to an EMR, and consume the EMR's order-response message to obtain the EMR-assigned accession (Filler).

1.3 Conformance references

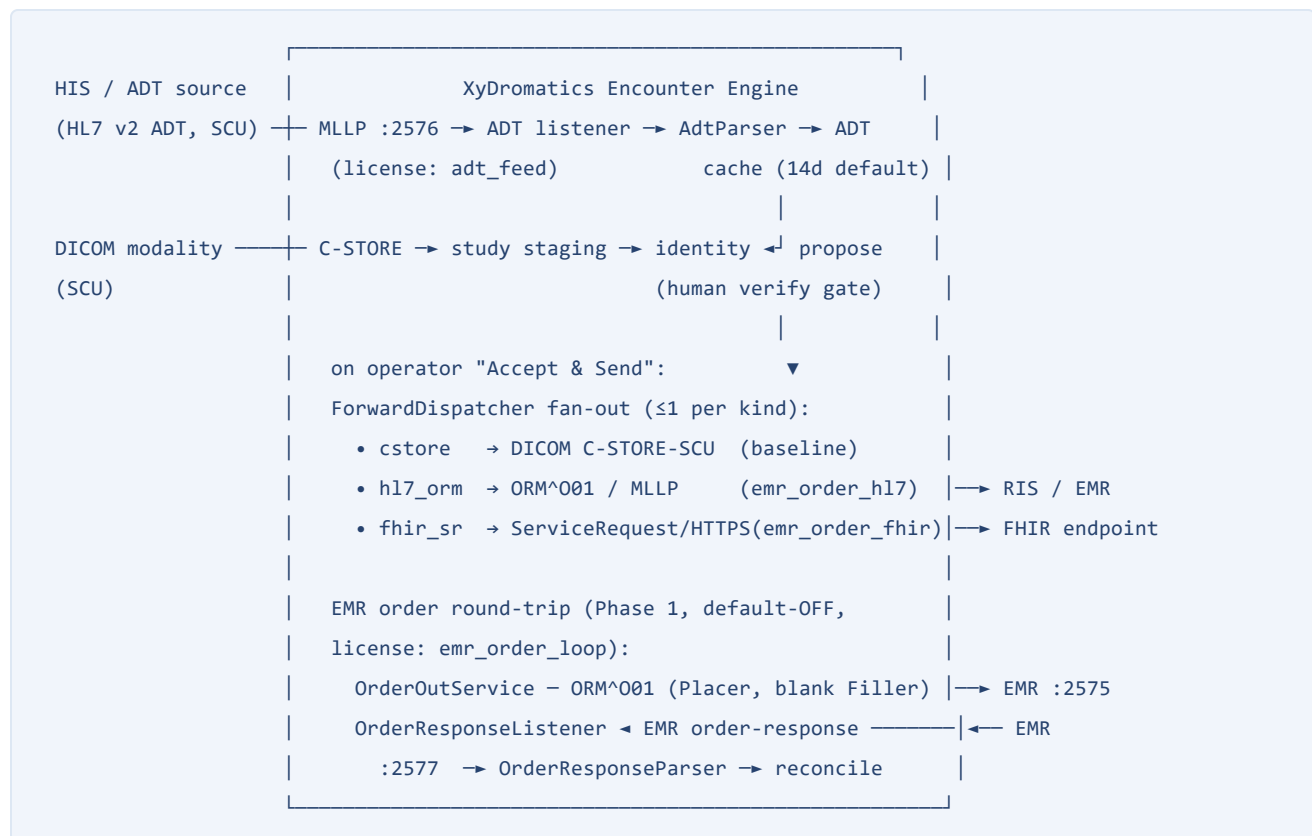
- HL7 v2.x messaging (ADT, ORM, order-response message families).
- HL7 Minimal Lower Layer Protocol (MLLP): `<VT> 0x0B + body + <FS> 0x1C + <CR> 0x0D`.
- HL7 FHIR R4 — `ServiceRequest` resource, RESTful `create` interaction.

1.4 Implementation notes that bound the conformance claim

- The HL7 v2 parsers are **purpose-scoped inline parsers**, not a general HL7 engine. The ADT parser reads only MSH + PID + PV1 (+ OBR-18 best-effort). The order-response parser reads only MSH + ORC + OBR. Fields and segments outside those are neither validated nor preserved. (`H17/AdtParser.cs` , `H17/OrderResponseParser.cs` .)
 - The FHIR `ServiceRequest` payload is **hand-built from a JSON template** (no FHIR SDK dependency). Only the fields enumerated in §4 are emitted. (`Services/Forwarders/FhirServiceRequestForwardSender.cs` .)
 - All HL7 v2 message text is exchanged as **UTF-8**. (`H17/MllpFraming.cs` , `H17MllpSender` .)
-

2. Implementation model

2.1 Application data flow



(The C-STORE/DICOM ingest + forward path is covered by the Encounter Engine DICOM conformance material; this document covers the HL7 v2 + FHIR surfaces only.)

2.2 Functional definitions

- **ADT MLLP listener** — inbound HL7 v2.x ADT receiver; license-gated by `adt_feed`; binds `AdtListenerPort` (default 2576/tcp) when enabled. (`Services/AdtListenerService.cs` .)
- **ADT identity cache** — short-retention store (default 14 days, `AdtCacheRetentionDays`) of cacheable ADT events, keyed for patient-identity lookup at study verification time. (`Stores/AdtCacheStore.cs` .)
- **HL7 ORM forwarder** — outbound HL7 v2.3 ORM^O01 emitter over MLLP for a verified study. (`Services/Forwarders/Hl7OrmForwardSender.cs` .)
- **FHIR ServiceRequest forwarder** — outbound FHIR R4 ServiceRequest emitter over HTTPS for a verified study. (`Services/Forwarders/FhirServiceRequestForwardSender.cs` .)
- **EMR order round-trip** — outbound ORM^O01 order request + inbound EMR order-response listener for accession assignment. Phase 1, default-OFF, license `emr_order_loop` . (`Services/OrderOutService.cs` , `Services/OrderResponseListener.cs` , `HL7/OrderResponseParser.cs` , `Services/OrderReconcileService.cs` .)

3. Supported HL7 v2 messages

3.1 Inbound — ADT (received by the ADT MLLP listener)

The ADT listener accepts any HL7 v2.x message on its MLLP port and **ACKs every parseable payload** per HL7 best practice. It only *caches* identity rows for the ADT trigger events enumerated below. Non-ADT message types, and ADT events not in the cacheable set, are **ACKed but not cached** — the ADT cache is an identity-resolution surface, not a general HL7 receipt log. (`Hl7/AdtParser.cs` `CacheableEvents` ; `Services/AdtListenerService.cs` `ProcessMessage` .)

Cached ADT trigger events (`AdtParser.CacheableEvents`):

Message	Trigger event	Meaning	Cached?
ADT^A01	A01	Admit / visit notification	✓
ADT^A04	A04	Register a patient	✓
ADT^A05	A05	Pre-admit a patient	✓
ADT^A08	A08	Update patient information	✓
ADT^A28	A28	Add person information	✓
ADT^A31	A31	Update person information	✓
ADT^A40	A40	Merge patient — patient identifier list	✓

Other ADT events (e.g. A02, A03, A09, A11, A13, ...) — ACKed `AA` (“Accepted — event not cached”), **not cached**. Not an error condition; simply outside the identity-cache scope.

Non-ADT message types (ORM, ORU, SIU, ...) arriving at the ADT port — ACKed `AA` (“Accepted — not cached”). The listener does not parse or route them.

Segments parsed from a cached ADT message (`AdtParser.TryParse`):

Segment	Fields consumed	Mapped to
MSH	MSH-9 (message type), MSH-10 (control id), MSH-3 (sending app), MSH-4 (sending facility)	message_type, control_id, sending_app, sending_facility
PID	PID-3 (patient id — first component before <code>^</code>), PID-5 (name), PID-7 (DOB), PID-8 (sex)	patient_id, patient_name, patient_dob, patient_sex
PV1	PV1-2 (patient class), PV1-3 (location), PV1-19 (visit number)	patient_class, location, visit_number
OBR	OBR-18 (best-effort; usually empty in ADT)	accession

A cacheable ADT message with **no PID-3** is ACKed `AA` but **not cached** (no identity key to match on). Segments other than MSH/PID/PV1/OBR are ignored. There is no repeat-field, component-substructure, or Z-segment processing beyond the fields listed.

ACK semantics emitted by the ADT listener (`AdtParser.BuildAck`):

Condition	MSA-1 ACK code
Cached successfully / accepted-not-cached / no-PID / event-not-cached	AA
Empty payload, no segments, or malformed/missing MSH	AR (Application Reject)
Cache insert exception	AE (Application Error)

The ACK is itself a minimal HL7 v2.5 MSH + MSA message (MSH|^~\&|ENC_ENGINE|XyDromatics|...|ACK|...|P|2.5 + MSA|<code>|<control id>).

3.2 Outbound — ORM^O01 (forward to RIS/EMR after verification)

When a study is verified and an enabled `hl7_orm` forward destination exists, the engine emits **one HL7 v2.3 ORM^O01 per verified study** (not per instance) over MLLP. (`Services/Forwarders/Hl7OrmForwardSender.cs` `BuildOrmO01` .)

Segment	Produced fields	Notes
MSH	MSH-1/2 encoding chars; MSH-3 sending app; MSH-4 sending facility; MSH-5 receiving app; MSH-6 receiving facility; MSH-7 datetime; MSH-9 = <code>ORM^O01</code> ; MSH-10 control id (StudyInstanceUID + timestamp, ≤199 chars); MSH-11 = <code>P</code> ; MSH-12 = <code>2.3</code>	Sending/receiving app/facility from destination config; defaults <code>ENCOUNTER_ENGINE</code> / <code>XYDROMATICS</code> / <code>RIS</code> / <code>HOSPITAL</code> .
PID	PID-1 set id; PID-3 patient id; PID-5 name; PID-7 DOB (YYYYMMDD); PID-8 sex	Built from the verified (final) identity, not raw device values.
PV1	PV1-1 set id; PV1-2 class = <code>0</code> (outpatient)	Fixed minimal PV1 (<code>PV1 1 0</code>).
ORC	ORC-1 = <code>NW</code> (new order); ORC-2 placer = accession; ORC-3 filler = accession	Placer and Filler are both the engine's own minted accession (single-source origin).
OBR	OBR-1 set id; OBR-2 placer = accession; OBR-3 filler = accession; OBR-4 universal service id = modality (or <code>DICOM_STUDY</code>); OBR-7 observation datetime	Modality-derived service id when known.

ORM^O01 emission is deliberately narrow (an “order placed + study complete” event), and is **not** an ORU^R01 result message — the Encounter Engine does not transmit observation results over HL7 v2. Sites needing richer structured content use the FHIR ServiceRequest forwarder (§4) or read the DICOM directly. HL7 field escaping (`|` , `^` , `~` , `\` , `&` → their escape sequences) is applied to all interpolated values.

3.3 Outbound — ORM^O01 order request (EMR order round-trip, Phase 1)

When the EMR order round-trip is enabled (`EmrOrderLoop.Enabled` AND license `emr_order_loop` ; **default OFF**), the engine mints a Placer Order Number and sends an HL7 v2.3 ORM^O01 **order request** to the EMR over

MLLP. This shape differs from §3.2 in one respect: **ORC-3/OBR-3 (Filler) are BLANK** — the EMR owns accessioning and returns the Filler in its order-response. (`Services/OrderOutService.cs` `BuildOrderRequest` .)

Segment	Produced fields	Notes
MSH	as §3.2, MSH-9 = <code>ORM^001</code> , MSH-12 = <code>2.3</code>	Receiving app default <code>EMR</code> .
PID	PID-1/3/5/7/8	Verified identity resolved at the verify step.
PV1	<code>PV1 1 0</code>	Minimal.
ORC	ORC-1 = <code>NW</code> ; ORC-2 = Placer Order Number; ORC-3 = blank	Placer minted from <code>EmrOrderLoop.PlacerTemplate</code> (default <code>EEP-{YYYYMMDD}-{SEQ:06d}</code>).
OBR	OBR-1; OBR-2 = Placer; OBR-3 = blank ; OBR-4 universal service id; OBR-7 obs datetime	Filler intentionally empty.

3.4 Inbound — EMR order-response (EMR order round-trip, Phase 1)

The order-response listener (MLLP, `EmrOrderLoop.InboundListenerPort` , default 2577/tcp — **distinct from the ADT port**) consumes the EMR's return message that echoes the Placer and carries the EMR-assigned Filler (accession). The parser is **layout-driven, not message-type-driven**: it reads the Placer/Filler from ORC/OBR regardless of the exact MSH-9, so a single parser serves the common return message families.

(`H17/OrderResponseParser.cs` ; configured by `EmrOrderLoop.ReturnMessageType` .)

Return message family	ReturnMessageType value	Segments parsed	Supported?
ORM^O01 with ORC-1 = NA/RE	<code>orm_na</code>	MSH, ORC, OBR	✅ Phase 1
ORG^O20	<code>org_o20</code>	MSH, ORC, OBR	✅ Phase 1
ORR^O02 (general order response; common on some EMRs)	<code>orr_o02</code>	MSH, (MSA), ORC, OBR	✅ parsed by the same ORC/OBR layout reader
SIU^S12 (scheduler)	<code>siu_s12</code>	SCH segment	❌ not supported / planned — the number is carried in SCH, which this parser does not read. Config value is accepted but the SCH layout is not yet implemented.

Fields parsed from the order-response (`OrderResponseParser.TryParse`):

Field	Source	Use
MSH-9	message type	logged/echoed
MSH-10	control id	echoed in the ACK
OBR-2 → fallback ORC-2	Placer Order Number (EI; id component)	look up the held study
OBR-3 → fallback ORC-3	Filler Order Number = EMR-assigned accession	stamp + reconcile

A response with **no Placer in OBR-2/ORC-2** is rejected (`AR`). Reconciliation (stamp held .dcm files with verified identity + the EMR accession, then flip the held study to verified, reusing the same forward dispatch) is idempotent on re-delivery and recovers late accessions that arrive after the timeout sweeper has parked a study as `order_exception` . (`Services/OrderReconcileService.cs` .)

ACK semantics emitted by the order-response listener: `AA` on successful reconcile; `AE` on unknown-placer / no-accession / stamp failure; `AR` on unparseable payload. (`Services/OrderResponseListener.cs` `ProcessMessage` .)

3.5 HL7 v2 messages explicitly NOT supported

Message	Status
ORU^R01 (observation result)	Not supported. The Encounter Engine emits ORM^O01, not ORU^R01. (The sibling SR Engine carries the ORU^R01 path; see DOC-2026-077.)
SIU^S12 / scheduling (SCH segment)	Not supported / planned — accepted as a <code>ReturnMessageType</code> config value but the SCH layout is not parsed (§3.4).
MDM, DFT, BAR, RDE, MFN, QBP/RSP, and all other v2 families	Not supported. Outside the engine's scope.
General HL7 routing / receipt logging	Not supported. Non-ADT messages at the ADT port are ACKed but not stored; the Router product owns the wider HL7 receipt log.

4. Supported FHIR resources and interactions

The engine produces FHIR over HTTPS through one forwarder kind (`fhir_sr`), license-gated by `emr_order_fhir` , when an enabled FHIR forward destination exists for a verified study. (`Services/Forwarders/FhirServiceRequestForwardSender.cs` .)

4.1 FHIR version and interaction

- **FHIR version:** R4.
- **Interaction:** RESTful `create` — HTTP `POST {base_url}/ServiceRequest` .
- **Content type:** `application/fhir+json` (request and `Accept`).
- **Success criterion:** HTTP 2xx. 4xx/5xx → failure, with the first 512 bytes of the response body captured as a snippet.
- **Transport security:** `base_url` MUST be `https://` (cleartext `http://` is permitted only for `localhost` /loopback smoke testing).
- **Authentication (optional):** bearer token (`Authorization: Bearer ...`) or HTTP Basic (`basic_username` / `basic_password`); both are configured per-destination.

4.2 Resource produced — `ServiceRequest` (R4)

One `ServiceRequest` resource is POSTed per verified study. The payload is a hand-built, intentionally flat template carrying only the fields a receiving EMR needs to link a completed study to an order:

FHIR element	Value	Source
<code>resourceType</code>	"ServiceRequest"	constant
<code>identifier[0].system / .value</code>	accession (system default <code>urn:xydromatics:accession</code> , overridable via <code>identifier_system</code>)	verified accession
<code>status</code>	"completed"	constant — study past its verification gate
<code>intent</code>	"order"	constant
<code>code.coding[0]</code>	DICOM modality coding (<code>system</code> <code>http://dicom.nema.org/resources/ontology/DCM</code> , <code>code</code> = modality or <code>OT</code>) + <code>code.text</code>	study modality
<code>subject.identifier.system / .value</code>	patient id (system default <code>urn:xydromatics:patient-id</code> , overridable via <code>patient_system</code>)	verified patient id
<code>subject.identifier.display</code>	patient name (when present)	verified name
<code>occurrenceDateTime</code>	ISO-8601 UTC verification time	verify timestamp
<code>requester.display</code>	"XyDromatics Encounter Engine (verifier: <user>)"	authenticated verifier
<code>note[0].text</code>	"StudyInstanceUID: <uid>"	study UID cross-reference

4.3 FHIR interactions and resources explicitly NOT supported

Capability	Status
FHIR <code>read</code> / <code>search</code> / <code>update</code> / <code>delete</code> / <code>transaction</code> / <code>batch</code>	Not supported. The engine is a one-way <code>create</code> producer; it does not host a FHIR server or query endpoint.
<code>Patient</code> , <code>ImagingStudy</code> , <code>DiagnosticReport</code> , <code>Encounter</code> , <code>Bundle</code> , or any resource other than <code>ServiceRequest</code>	Not supported / not produced.
FHIR R5 / STU3 / DSTU2	Not supported. Output targets R4.
FHIR conformance/ <code>CapabilityStatement</code> advertising	Not applicable — the engine is a client, not a server.
Inbound FHIR (the engine consuming FHIR)	Not supported. Inbound clinical context arrives via HL7 v2 ADT (§3.1), not FHIR.

5. Encoding and transport

5.1 HL7 v2 over MLLP

- **Framing (MLLP):** <VT> (0x0B) + message body + <FS> (0x1C) + <CR> (0x0D). A peer that omits the sentinels or frames LF-only is treated as a sender bug, not accommodated. (`Hl7MllpFraming.cs` ; `Hl7MllpSender` in `Synthology.Shared.Dicom` .)
- **Character encoding:** UTF-8 on both send and receive.
- **Segment terminator:** <CR> (0x0D). Inbound parsers normalize `\r\n / \n` to `\r` defensively.
- **Inbound frame size cap:** 8 MB per MLLP frame; a frame exceeding the cap with no <FS> terminator causes the connection to be dropped (DoS guard). Read timeout 30 s, write timeout 10 s. (`Services/AdtListenerService.cs` , `Services/OrderResponseListener.cs` .)
- **Outbound send:** single MLLP connect → write → read ACK within the timeout window; MSA-1 of the returned ACK is interpreted as `AA / CA` → success, any other code (`AE` , `AR` , `CE` , `CR` , ...) → NAK/failure. (`Hl7MllpSender.SendAsync` .)

5.2 Default ports

Channel	Direction	Default port	Config key	License feature
ADT feed	inbound (EE listens)	2576/tcp	<code>AdtListenerPort</code>	<code>adt_feed</code>
EMR order request (ORM^O01)	outbound (EE → EMR)	2575/tcp	<code>EmrOrderLoop.EmrPort</code>	<code>emr_order_loop</code>
EMR order-response	inbound (EE listens)	2577/tcp	<code>EmrOrderLoop.InboundListenerPort</code>	<code>emr_order_loop</code>
HL7 ORM forward	outbound (EE → RIS/EMR)	per-destination	<code>destination config_json.port</code>	<code>emr_order_hl7</code>
FHIR ServiceRequest forward	outbound HTTPS (EE → FHIR endpoint)	per-destination <code>base_url</code>	<code>destination config_json.base_url</code>	<code>emr_order_fhir</code>

All ports are operator-configurable. The ADT and EMR order-response listeners bind `0.0.0.0` on their respective ports. Listeners do not bind unless both the relevant config flag is set AND the gating license feature is present (the engine force-resets the enable flag to false at startup when unlicensed).

5.3 FHIR over HTTPS

See §4.1 — `POST {base_url}/ServiceRequest` , `application/fhir+json` , HTTPS enforced (localhost-only cleartext exception), optional bearer/basic auth, configurable timeout (default 30 s, floor 5 s).

6. Conformance notes and limitations

1. **Purpose-scoped parsers, not a general HL7 engine.** Only the segments and fields enumerated in §3 are parsed/produced. Repeating fields, component sub-structure beyond what is listed, and Z-segments are

not processed. A message that round-trips through the engine is **not** preserved field-for-field.

2. **ADT path is propose-only and human-gated.** The ADT cache feeds an operator-confirmed identity proposal; it never auto-commits an identity to a study. The Encounter Engine does not act on ADT clinical content beyond the identity fields in §3.1.
3. **ORM^O01 is an order/event signal, not a result.** No ORU^R01, no observation (OBX) transmission.
4. **EMR order round-trip is Phase 1, default-OFF.** It binds nothing and changes no behavior until both `EmrOrderLoop.Enabled` and the `emr_order_loop` license feature are set. `siu_s12` return handling (SCH segment) is **planned, not implemented** (§3.4).
5. **FHIR is one-way create of ServiceRequest only.** No FHIR server, no query surface, no other resources, R4 only (§4.3).
6. **ACK code mapping is fixed** (§3.1, §3.4). The engine ACKs liberally (accept + don't-cache) on parseable-but-out-of-scope inbound messages, and reserves `AR` / `AE` for unparseable or processing-error conditions.
7. **License gating is enforced server-side.** Disabling a license feature removes the corresponding inbound listener (at next start) and blocks the corresponding forward kind at dispatch time, independent of UI state.
8. **No diagnostic or interpretive function.** Consistent with the non-device posture (§1.2), nothing in the HL7/FHIR surface analyzes, interprets, or makes a clinical determination on patient data.

7. Versions

Item	Value
Product	XyDromatics Encounter Engine v1.0.x (source grounded at v1.0.0.87)
HL7 v2 message versions emitted	MSH-12 = 2.3 (ORM^O01 outbound, order request); 2.5 (ACK messages)
HL7 v2 message versions accepted (inbound)	v2.x ADT and order-response families per the segment layouts in §3; MSH-12 not version-gated
FHIR version (outbound)	R4
MLLP	<VT> 0x0B / <FS> 0x1C / <CR> 0x0D, UTF-8

8. Source grounding

This conformance statement was authored against the following Encounter Engine source files:

- `H17/AdtParser.cs` — ADT parse + cacheable events + ACK builder.
- `H17/MllpFraming.cs` — MLLP framing constants/helper.
- `H17/OrderResponseParser.cs` — EMR order-response parser (ORC/OBR layout).
- `Services/AdtListenerService.cs` — inbound ADT MLLP listener + ACK semantics.
- `Services/OrderResponseListener.cs` — inbound EMR order-response MLLP listener.

- `Services/OrderOutService.cs` — outbound ORM^O01 order request (Placer/blank Filler).
- `Services/OrderReconcileService.cs` — Placer/Filler reconcile + stamp + flip.
- `Services/Forwarders/Hl70rmForwardSender.cs` — outbound ORM^O01 forwarder.
- `Services/Forwarders/FhirServiceRequestForwardSender.cs` — FHIR R4 ServiceRequest forwarder.
- `Services/Forwarders/IForwardSender.cs` — forward context/result contract.
- `Services/Forwarders/ForwardDispatcher.cs` — forward fan-out + license gating.
- `Stores/ForwardStore.cs` — `ForwardKind` constants (cstore / hl7_orm / fhir_sr).
- `Configuration/EmrOrderLoopConfig.cs` — EMR order-loop config + return message types.
- `Configuration/EncounterEngineConfig.cs` — ADT listener ports / retention.
- `Synthology.Shared.Dicom/Hl7/Hl7MllpSender.cs` — shared MLLP send + ACK parse.

End of DOC-2026-408 (Draft).

Regulatory classification. Non-device software under the FD&C Act §520(o)(1)(D) software exclusion. This statement is published by Synthology Healthcare Solutions Group, LLC as a controlled document; the authoritative version is the approved record in the Synthology quality-management system.