

OpenADR 2 0 B VEN Pull Test Plan

HTTP message exchange testing

Version 1.0 | 21 September 2026

This plan defines a VTN simulator and 98 scenario groups for evaluating an OpenADR 2.0 B VEN operating in HTTP pull mode. It covers registration, polling, events, reporting, opt operations, XML validation, transport failures, and recovery. Each scenario identifies the stimulus, observable result, requirement basis, and implementation priority.

The primary target is a full B-profile VEN. Report Only VENs use a tailored subset. The tests establish whether messages are correct and whether the VEN maintains the protocol state needed to exchange them reliably. Electrical load response, metering accuracy, and utility program performance are outside scope.

All connections use plain HTTP. The specification normally requires TLS; this is an intentional transport substitution for this test campaign. TLS, certificates, authentication through certificates, and XML signatures are excluded. A passing result therefore establishes coverage of the tested message behavior, not complete OpenADR certification.

1 Scope and required capabilities

The VTN harness is an HTTP server; the VEN is always the HTTP client. VTN-initiated operations reach the VEN in an HTTP response to oadrPoll. The VEN sends any required application response as a new HTTP POST to the corresponding service endpoint. The harness never needs to initiate an HTTP connection to the VEN.

Use the following scope when configuring the suite. Optional feature tests become required for this campaign when the feature is declared or offered in metadata. The product capability declaration must distinguish a feature that is not offered in a deployment from a missing minimum B-profile capability.

Area	Full B VEN scope	Tailoring
Registration and polling	EiRegisterParty and oadrPoll	Required for both full and Report Only VENs
Events	SIMPLE with level; ELECTRICITY_PRICE with price; LOAD_DISPATCH with setpoint	Minimum capabilities in CR 510; omit event tests for Report Only VENs
Reporting	Metadata; TELEMETRY_STATUS including oadrOnline and oadrManualOverride for a resourceID; TELEMETRY_USAGE capability for testing; storage for at least 100 recent data points	CR 510; sample usage data is permitted when no meter is attached
Additional reporting	HISTORY_USAGE, custom reports, consumption of VTN data reports	Conditional on capability; HISTORY_USAGE is required for Report Only VENs
Opt operations	Event-specific opt qualification through EiOpt	Required for a full VEN under CR 510; opt schedules are conditional
Other event features	Advertised signals, resource targeting, baselines, optional event fields	Exercise supported combinations and rejection of unsupported combinations
Excluded	TLS, certificates, XML signatures, XMPP, HTTP push, A-profile fallback, physical control verification	No verdict for these capabilities

CR means a numbered conformance rule in section 11.2 of the attached profile specification. The source baseline and interpretation decisions appear in section 8. The OpenADR PICS remains the appropriate companion for a

formal certification applicability decision; this engineering plan is executable using a product capability declaration.

2 VTN harness capabilities

The harness should combine a compliant VTN model with explicitly enabled fault injection. Normal fixtures must pass schema and semantic checks before transmission. Negative fixtures must identify the single intended defect so a harness mistake cannot be mistaken for a VEN failure.

H01 HTTP service endpoints

Expose `http://host:port/prefix/OpenADR2/Simple/2.0b/` followed by `EiRegisterParty`, `OadrPoll`, `EiEvent`, `EiReport`, or `EiOpt`. Support an optional prefix and port. Dispatch by the operation inside the XML, validate the service destination, and capture method, URL, headers, raw body, connection, and timestamps. Permit multiple simultaneous VEN requests so asynchronous report traffic and polling can progress independently.

H02 XML and semantic validation

Compile the top-level `oadr_20b.xsd` and its local imports. Validate all ordinary inbound and outbound messages. Add a separate rule engine for constraints that XSD does not enforce: required `schemaVersion`, allowed transport value, ID relationships, time syntax, event interval sums, required acknowledgments, report descriptors, and service sequencing. Use a secure parser with external entity resolution and network schema retrieval disabled. Do not add `xsi:schemaLocation` to transmitted payloads.

H03 Registration state

Maintain per-VEN `venID`, `vtnID`, `registrationID`, `profile`, `transport`, `pull flag`, `poll frequency`, `metadata`, and `report subscriptions`. Provide `query`, `initial registration`, `re-registration`, `cancellation` in both logical directions, `explicit rejection`, and `simulated loss of VTN registration state`. Permit `preconfigured IDs` and `out-of-band registration` as separate fixtures.

H04 Poll queue and concurrency

Provide a per-VEN queue of the eight permitted response types in section 3. Return one operation per poll. Support an `empty-queue response`, `queue replenishment while draining`, `delayed acknowledgments`, `duplicate delivery`, and both VTN policies permitted by Annex C: `continue serving polls with logical responses pending`, or `temporarily withhold a poll response`. Use a bounded wait when withholding a response. Track logical operations separately from TCP connections and HTTP transactions.

H05 Event model

Maintain a complete event set, versions, target definitions, response requirements, signal intervals, lifecycle times, `cancellation state`, and each VEN `opt response`. Normal `oadrDistributeEvent` messages carry the full relevant event snapshot, not only changed events. Generate all mandatory signals, multiple signals, optional baselines, randomization windows, late deliveries, and mixed valid and invalid events. Preserve request IDs independently of event IDs and versions.

H06 Reporting model

Act as both report requester and report provider. Exchange `metadata`, request subsets of `rIDs`, schedule one-shot and periodic reports, support report requests embedded in `oadrRegisteredReport`, and `cancellation` embedded in `oadrUpdatedReport`. Maintain `reportSpecifierID`, `reportRequestID`, `rID`, `pending reports`, `data quality`, `interval`

chronology, and metadata replacement effects. Supply deterministic telemetry and historical data for comparison. Offer an empty VTN report catalog as a normal fixture.

Report-request fixtures use `specifierPayload` readingType `x-notApplicable` and exclude the optional elements forbidden by CR 321, including `itemBase` and `report-interval` notification, tolerance, ramp-up, and recovery. Apply these checks to VEN-generated report requests as well.

H07 Opt model

Accept event-specific and scheduled opt operations. Keep resource-specific event opt state separate from the broad opt state conveyed by `oadrCreatedEvent`. Model program-specific schedule precedence explicitly, return matching IDs, and allow forced application errors without committing the requested state.

H08 Fault injection

Independently control HTTP status, application response code, payload type, schema defects, semantic defects, response timing, truncated bodies, disconnects before and after request acceptance, and lost acknowledgments. Support permanent and transient HTTP errors, retry sequences, and a recovery point. Record whether a request was committed before its response was lost.

H09 Time control

Use synchronized UTC clocks for integration runs, monotonic timestamps for elapsed-time assertions, and a deterministic seed for generated schedules. Provide a clock adapter only if the VEN supports test-time control. Otherwise use real-time fixtures. The harness must not assume that advancing its own clock advances the VEN clock. Record clock skew and timing tolerance separately from protocol deadlines.

H10 VEN observation and stimulation

Provide an adapter for configuration, reset, local opt choices, online/manual override state, and test telemetry. Read protocol state through a supported diagnostic API, logs, or supervised UI observation. This adapter is test instrumentation, not another OpenADR transport. If only wire traffic is available, report internal-state checks as unobservable; a successful acknowledgment alone does not prove correct scheduling, deletion, or physical execution.

H11 Evidence and verdicts

For each run, retain case and vector ID, DUT build/configuration, source hashes, fixture seed, input data, raw HTTP/XML, normalized fields, schema results, rule assertions, timestamps, retries, state transitions, and final verdict. Produce an operator-readable report and machine-readable results. Identify the first violated assertion and retain subsequent traffic for diagnosis.

H12 Isolation and harness self checks

Reset registration, event, report, and opt state between independent scenarios. Retain state only for an explicitly defined sequence. Validate fixtures and expected responses with an independent schema validator. Use local round trips to check queue routing, full event snapshots, exact error injection, and correlation before testing a VEN. A failed harness precondition makes a case invalid, not a DUT failure.

3 Message exchange map

All rows below describe payloads inside the normal `oadrPayload` / `oadrSignedObject` envelope. HTTP 200 and an application responseCode of 200 are separate checks. Every POST originates at the VEN.

VEN initiated exchanges

VEN POST operation	Service	Normal HTTP response payload
oadrQueryRegistration or oadrCreatePartyRegistration	EiRegisterParty	oadrCreatedPartyRegistration
oadrCancelPartyRegistration	EiRegisterParty	oadrCanceledPartyRegistration
oadrRequestEvent	EiEvent	oadrDistributeEvent
oadrCreatedEvent	EiEvent	oadrResponse
oadrRegisterReport	EiReport	oadrRegisteredReport; may contain oadrReportRequest
oadrCreateReport	EiReport	oadrCreatedReport
oadrUpdateReport	EiReport	oadrUpdatedReport; may contain oadrCancelReport
oadrCancelReport	EiReport	oadrCanceledReport
oadrCreateOpt or oadrCancelOpt	EiOpt	oadrCreatedOpt or oadrCanceledOpt respectively

Responses to a VEN poll

Payload in HTTP response to OadrPoll	Required follow-up VEN POST	Service and harness completion
oadrResponse with success	None; queue is empty	Resume normal poll interval; no acknowledgment loop
oadrDistributeEvent	oadrCreatedEvent for events marked always	EiEvent; harness returns oadrResponse
oadrCreateReport	oadrCreatedReport	EiReport; harness returns oadrResponse; requested data follows separately
oadrRegisterReport	oadrRegisteredReport	EiReport; normally complete with oadrResponse when no piggyback request is sent
oadrCancelReport	oadrCanceledReport	EiReport; harness returns oadrResponse; final data follows if requested
oadrUpdateReport	oadrUpdatedReport	EiReport; harness returns oadrResponse; conditional report-consumer path
oadrCancelPartyRegistration	oadrCanceledPartyRegistration	EiRegisterParty; harness returns oadrResponse
oadrRequestReregistration	oadrResponse, then oadrCreatePartyRegistration	EiRegisterParty; first POST gets empty HTTP 200, then oadrCreatedPartyRegistration

The re-registration acknowledgment ends with empty HTTP 200 as shown in Figure 22 on page 49. Do not generate an endless exchange of oadrResponse messages. A successful empty poll response is an XML oadrResponse, not an empty HTTP body. For report piggyback requests in either logical direction, follow the

report exchange in section 8.3.2.1 and validate the resulting `oadrCreatedReport`; do not assume every report acknowledgment ends the logical transaction.

Example event sequence

1. VEN posts `oadrPoll` to `OadrPoll` with its `venID`.
2. Harness responds HTTP 200 with `oadrDistributeEvent` containing event E1, `modificationNumber 0`, `requestID Q1`, and `responseRequired` always.
3. VEN posts `oadrCreatedEvent` to `EiEvent`. Its `eventResponse` identifies E1/0 and Q1, returns the selected `optType` and `responseCode`, and carries the VEN ID in the schema-defined location.
4. Harness responds HTTP 200 with `oadrResponse` and application success.
5. VEN polls again while draining. Harness returns a successful `oadrResponse` when the queue is empty. Normal polling then resumes.

Within `oadrCreatedEvent`, the enclosing `eiResponse requestID` is empty when `eventResponses` are present; the relevant `requestID` appears in each `eventResponse`. `oadrPoll` has no `requestID`. Do not invent one or correlate events solely by the distribution `requestID`, which may be reused. See CR 40-43 and 500.

4 Fixtures and verdict rules

Common setup

Before the first run, record the VEN build, full or Report Only role, enabled signals and units, resource IDs and classes, market contexts, report catalog, optional report-consumer role, opt scheduling capability, compression encodings, retry limits, and restart policy. Configure unsigned messaging and a plain HTTP VTN URL. Confirm that this mode is reachable in the VEN test configuration before executing cases.

Unless a scenario says otherwise, start from a registered VEN with metadata exchanged in both directions, initial event synchronization completed, no queued operations, no events, no scheduled data reports, and no opt schedules. Complete each application exchange before checking its resulting state. Preserve unrelated active objects in cases testing targeted updates or cancellation.

Use `VEN_A`, `VTN_A`, and `REG_A` as illustrative IDs, replacing `VEN_A` with a preconfigured or assigned value when needed. Provide resources `RES_1` and `RES_2` if supported, with different device classes, and market context `https://example.test/dr/program`. Multi-resource vectors are conditional on the device model; do not fail a single-resource VEN merely because a second physical resource is unavailable.

Use event E1 with `modificationNumber 0`, future start `T0`, duration `PT60S`, four `PT15S` SIMPLE intervals with `UIDs 0-3` and values `0, 1, 2, 3`. Include required active-period properties and `x-eiNotification`. Set `T0` sufficiently ahead of the measured poll and processing delay. Use separate numeric and resource-status telemetry fixtures. Select granularity and `reportBackDuration` from the VEN metadata; `PT10S/PT30S` are convenient examples only if supported.

Timing and observability

Let `P` be the requested poll interval, `A` the agreed maximum application response time for this campaign, and `E` the measured clock/network tolerance. Start with `P = PT5S` when supported, `A = 5` seconds for a local bench, and `E = 250` ms, then adjust from an initial latency measurement. `A` and `E` are test settings, not OpenADR limits. HTTP request timeout must be configurable and at least 5 seconds. Measure an application response deadline from complete receipt of the triggering HTTP body.

Observe ordinary polling over at least 30 idle cycles, retries over at least five consecutive failures, and event randomization over at least 30 independently reset events. Values must satisfy the configured bounds; identical

samples or absent jitter require investigation. Statistical uniformity is not mandated by the profile. Do not use an exact interval-count oracle where the specification permits report timing variation.

Applicability and requirement strength

P0 means essential before useful interoperability testing; P1 means required for the normal regression campaign; P2 means extended robustness or endurance. Priority does not alter whether a requirement is mandatory.

M identifies a MUST requirement. S identifies a SHOULD recommendation; report a departure separately and assess its justification. C identifies a conditional feature with mandatory rules when exercised. R identifies an engineering robustness expectation for which the specification does not define every recovery detail. Cases can contain more than one class; keep their assertions separate.

Use Pass, Fail, Not applicable, Blocked, or Inconclusive. Fail a mandatory assertion only when its applicability and oracle are established. Use Inconclusive for a specification ambiguity or an unobservable state assertion; Blocked for missing setup or instrumentation. A case with a SHOULD departure is not automatically a mandatory-conformance failure.

Error handling oracle

Condition	Expected interpretation
HTTP 200 plus application 4xx	Transport succeeded; application operation failed. Do not commit success solely from HTTP status.
HTTP 404, 406, 501	Apply the distinct retry prohibitions in section 9.1.5.
HTTP 503	Quiesce and retry; HTTP 500 permits retry after quiescence.
Schema-valid logical event error	Use application error handling, normally per-event optOut with 4xx while preserving valid events.
Invalid XML or schema in an HTTP response	VEN is the HTTP client and cannot return HTTP 406 as the response to that response. Check safe rejection, diagnostics, and recovery; use an application error only where a valid exchange supports it.

Exact codes are required where the conformance rule requires them, notably 460 in CR 109, 469 in CR 112, and 454 in CR 340. CR 48 otherwise permits a suitable 4xx for logical errors unless another rule specifies the value. Codes such as 450 for stale event versions and 452 for report IDs are often attached to SHOULD rules; preserve that distinction in verdicts.

5 Test scenarios

Each group below expands into one run per listed variant. For every run, apply the common setup, inject the stated stimulus, validate outbound XML and service routing, evaluate the expected result, and save the evidence. Mutate one condition at a time unless the group explicitly tests mixed input. A normal transaction after a negative test demonstrates recovery.

HTTP transport

T01 POST methods and endpoint routing

P0 | M | Sections 9.1.1-9.1.3; CR 37, 51, 511

Stimulus: Run registration, polling, event acknowledgment, reporting, and opt exchanges with and without a URI prefix and nondefault port. Allow only VEN-originated connections.

Expected: Every request uses POST and the correct versioned service path. All exchanges complete without a VEN HTTP listener or VTN-initiated connection. The operation is selected by XML, not an invented operation suffix in the URL.

T02 Required and optional HTTP headers

P0 | M/C | Section 9.1.9; CR 53

Stimulus: Capture requests for every service. Return valid application/xml responses, both without charset and with charset=utf-8. Vary User-Agent presence where configurable.

Expected: Host, Content-Length, and Content-Type are correct. Accept may be absent; when sent it is application/xml. Optional User-Agent does not change behavior. Encoding declarations and actual UTF-8 agree.

T03 Byte lengths and message boundaries

P1 | M | Section 9.1.9.4-9.1.9.5

Stimulus: Include multibyte UTF-8 text in permitted descriptive fields and consecutive messages of different sizes. Inspect framing in both directions.

Expected: Content-Length counts transmitted bytes, including compressed bytes when applicable. VEN requests do not use chunked transfer encoding. Each response is parsed exactly once without consuming bytes from another response.

T04 Connection close and reuse

P1 | R | Section 9.1.4; HTTP interoperability

Stimulus: Alternate compliant persistent responses and Connection: close responses. Close an idle connection before the next request.

Expected: The VEN reconnects or reuses connections correctly and completes the exchanges. Do not require connection reuse as a profile feature. Registration, events, and report state are not keyed to the lifetime of one TCP socket.

T05 Advertised response compression

P1 | C | Section 9.1.9.2-9.1.9.3

Stimulus: If the VEN sends Accept-Encoding, return each advertised encoding in a separate run with the corresponding Content-Encoding and correct length. Also run uncompressed exchanges without an encoding advertisement.

Expected: The VEN decodes the encodings it advertises and produces normal logical responses. Do not require gzip or deflate support from a VEN that does not advertise it. Unsupported encodings are a separate robustness vector.

T06 Configurable timeout and delayed valid replies

P0 | M/R | Section 9.1.6

Stimulus: Measure the configured timeout. Delay a valid response until just below it, then beyond it, and then restore immediate replies. Repeat for polling and one state-changing request.

Expected: The request timeout is configurable and at least 5 seconds. A reply inside the limit completes normally. A timeout triggers controlled recovery; it does not mark a logical operation successful. A late response does not cause duplicate state commitment.

T07 TCP failures around request acceptance

P1 | S/R | Sections 9.1.4 and 9.1.7

Stimulus: Refuse a connection, close mid-request, and close after the harness commits a complete request but before it returns the response.

Expected: The VEN records the failure and uses quiescence. Retry is recommended for TCP failure. When a request is retried, duplicate delivery does not produce conflicting registrations, report schedules, or opt state. Preserve the acceptance point in the evidence.

T08 Exponential retry backoff and jitter

P0 | M | Section 9.1.7

Stimulus: Cause five or more consecutive retryable failures, continue until the configured cap is reached, and then recover. Repeat with the random source independently seeded if supported.

Expected: Retry intervals use truncated binary exponential backoff with random jitter, rather than a tight loop or constant-rate flood. Check the configured initial interval and cap; the specification's approximately 1-second starting value is illustrative. Success restores the product's documented normal behavior.

T09 Transient HTTP server errors

P0 | M/R | Sections 9.1.5 and 9.1.7

Stimulus: Return HTTP 503 repeatedly, then 200; separately return 500. Use both empty and non-XML error bodies.

Expected: A 503 causes quiescence and retry. A 500 may be retried after quiescence. The VEN does not require an OpenADR body in a non-200 response and does not confuse a server error with an application success or an empty event snapshot.

T10 Permanent HTTP error handling

P1 | M | Section 9.1.5

Stimulus: Return 404, 406, and 501 in separate runs to a specific VEN operation. Repair configuration or payload before attempting recovery.

Expected: A 404 does not cause the same request to be resent. A 406 is not resent unchanged. A 501 is not resent without correcting the method. Record the failed service/operation and avoid declaring unrelated state-changing requests successful.

T11 HTTP success with application failure

P0 | M/R | Sections 8.7 and 9.1.5; service-specific rules

Stimulus: Return HTTP 200 carrying an appropriate schema-valid application error to registration, report creation, opt creation, and event acknowledgment in separate runs.

Expected: The VEN inspects responseCode and reports the rejected operation. It does not install an unaccepted registration or subscription, or present a rejected opt change as confirmed. Further retry behavior follows the declared recovery policy, not HTTP success alone.

T12 Invalid HTTP response framing

P2 | R | Fault tolerance extension

Stimulus: Separately send a truncated body, inconsistent length, empty body where XML is expected, wrong media type, invalid UTF-8, unsupported compression, and a chunked response. Finish each run with a valid exchange.

Expected: No crash, hang beyond the configured timeout, or success based on partial data. Rejection and recovery are observable. Receiving a nonconforming chunked response does not create a requirement that the VEN support chunking; distinguish tolerant acceptance from the required prohibition on sending it.

XML and common payload rules

X01 Envelope and operation structure

P0 | M | oadr_20b.xsd oadrPayload and oadrSignedObject; CR 509

Stimulus: Validate every observed VEN operation against the bundled schema and inspect its outer envelope.

Expected: Unsigned messages contain oadrPayload and one oadrSignedObject carrying exactly one operation, without ds:Signature. The operation has ei:schemaVersion="2.0b", including inherited attributes such as oadrCreateOpt. The XML namespaces and element ordering match the schema.

X02 Rules beyond schema validity

P0 | M | CR 45-46, 508-509; operation-specific rules

Stimulus: Audit positive traffic for schemaLocation, profile version, conditional IDs, and fields that are optional in XSD but required by a rule. Mutate an incoming message to omit one rule-required field while preserving XSD validity.

Expected: VEN output meets both schema and semantic rules. Omitted/empty optional values are treated as equivalent only when type constraints and specific rules permit it. Preserve initial-registration and query exceptions; do not insert venID where no schema element exists.

X03 Namespace and lexical equivalence

P1 | M/R | XML namespace semantics; bundled XSD

Stimulus: Send semantically identical valid messages with different namespace prefixes, valid default namespaces, indentation, attribute order, and XML declaration presence. Use valid Boolean lexical forms in Boolean fields.

Expected: Parsing uses namespace URIs and values, not prefix spelling or whitespace. Logical results are identical. Do not apply Boolean parsing rules to testEvent, which is a string and has its own rule.

X04 UTC time and duration syntax

P1 | M/R | CR 1 and 114

Stimulus: Test UTC timestamps ending in Z, with and without fractional seconds. Separately inject a zero-offset timezone form, a non-UTC offset, and a fractional duration that XSD may permit.

Expected: VEN output uses the specified UTC-Z representation and integer duration quantities. Valid fractional timestamp seconds are accepted or truncated appropriately. Nonconforming incoming forms are diagnosed without corrupting schedules. Millisecond configuration resolution does not authorize fractional durations on the wire.

X05 Identifier equality and string handling

P1 | M/C/R | CR 21, 43, 304, 400, 512; XSD string types

Stimulus: Change only ID case; reuse a legal requestID across transactions; use descriptive rIDs and IDs at the product's documented length limits. Test market-context normalization separately in E17.

Expected: venID and vtnID are case-sensitive. RequestID reuse is not treated as a duplicate event by itself. String IDs are not silently truncated or forced to numeric types. Bounds beyond declared limits produce controlled handling, not an invented protocol-wide maximum.

X06 Malformed XML and parser isolation

P2 | R | Fault tolerance extension

Stimulus: Return unclosed XML, malformed entities, and an external-entity reference to a controlled test endpoint, one at a time.

Expected: The VEN does not commit partial message state, crash, or retrieve the external resource. A subsequent valid transaction succeeds. Record diagnostics rather than requiring an impossible HTTP error response from the VEN client to an already received HTTP response.

X07 Schema-invalid payload mutations

P1 | R | Bundled XSD; section 9.1.5 context

Stimulus: Remove a mandatory element, duplicate a singleton, reorder sequence elements, use a wrong namespace or enum, and put two operations under oadrSignedObject. Change only one item per run.

Expected: The VEN avoids successful application of invalid data and recovers. Validate the mutation independently to prove it is schema-invalid. An exact on-wire error payload is required only where the profile defines a usable logical response for the fault.

X08 Unexpected but schema-valid operation

P1 | M/R | CR 33 and 48; section 8.6

Stimulus: Return a schema-valid payload of the wrong type to a VEN request, and separately return an operation not allowed as a poll response. Include an unexpected ID in an otherwise correct response.

Expected: The VEN detects the mismatch and does not silently commit it as success. Missing expected information and unexpected payload types use the CR 33 application-error rule where an application response is possible. Keep diagnostic-only paths separate when no return exchange is specified.

Registration

R01 Optional capability query

P1 | C/M | Section 8.4.1; CR 401 and 404

Stimulus: If query is supported, start unregistered and return oadrCreatedPartyRegistration advertising 2.0b/simpleHttp, with no venID or registrationID. Repeat with several advertised transports.

Expected: A query alone does not establish registration. The VEN selects the configured HTTP pull mode and can proceed to registration. Absence of IDs and poll frequency in an unregistered query response is not an error. Query support is not a prerequisite to direct registration.

R02 Initial in-band registration

P0 | M | Section 8.4; CR 401-405; registration XSD

Stimulus: Boot an unregistered, configured VEN. Return successful `oadrCreatedPartyRegistration` with assigned IDs, 2.0b/simpleHttp support, and requested poll frequency.

Expected: `oadrCreatePartyRegistration` selects 2.0b, exact XSD token `simpleHttp`, and `oadrHttpPullModel=true`. It omits `registrationID`, includes the required report-only and XML-signature capability flags, and need not supply a transport address. The VEN adopts returned IDs for later exchanges.

R03 Startup synchronization

P0 | M/S | CR 301, 309, 405 and 500

Stimulus: Complete a new registration with existing pending events in the VTN and an empty VTN metadata catalog. Observe the entire bootstrap sequence.

Expected: Metadata is exchanged in both directions and `oadrRequestEvent` obtains relevant events before ordinary service activity. Allow either metadata or initial event synchronization first, consistent with CR 309. Allow polls needed to retrieve VTN metadata. Verify event responses and transition to `oadrPoll` for ongoing retrieval.

R04 Preconfigured and out-of-band identity

P1 | C | CR 402 and 405

Stimulus: Run initial registration with a preconfigured `venID`. Separately configure an explicitly supported out-of-band registration, retaining the correct VTN identity and pull settings.

Expected: The preconfigured `venID` may appear in initial registration but `registrationID` does not. Out-of-band operation works according to its setup contract; it does not excuse a full B VEN from implementing `EiRegisterParty`. Mark unavailable configuration paths Not applicable.

R05 Registration rejection and incomplete replies

P1 | M/R | CR 33, 400, 403 and 509

Stimulus: Return application rejection; then a purported success missing `registrationID`, `venID`, or poll frequency in separate runs; then an unsupported profile/transport response.

Expected: Rejection does not establish registration. A schema-valid but incomplete success is identified as a semantic defect. The VEN does not begin normal exchanges with fabricated IDs or silently switch to an excluded transport. Where the specification leaves recovery open, evaluate the documented policy separately.

R06 VTN requested re-registration

P0 | M/R | Section 8.4.1; Figure 22; CR 402 and 405

Stimulus: With active events and reports, return `oadrRequestReregistration` to a poll. Acknowledge the VEN `oadrResponse` with empty HTTP 200. Return updated poll frequency to the subsequent registration request.

Expected: The VEN acknowledges and sends `oadrCreatePartyRegistration` with its existing `venID` and `registrationID`. The registration ID persists and the new configuration is applied. Do not require an initial-registration event refresh or metadata exchange solely because re-registration occurred; see the CR 301/405 discrepancy in section 8.

R07 VEN initiated re-registration

P1 | C/M | Section 8.4.1; CR 402

Stimulus: Change a supported registration setting locally and trigger re-registration while the harness retains REG_A. Accept the update without changing registrationID.

Expected: The VEN includes existing IDs, correlates the reply, and updates configuration without treating the exchange as a new registration. An intentionally new registration without existing IDs is tested as a separate reset of the relationship, not an equivalent re-registration.

R08 VTN cancels registration

P0 | M | CR 401 and 407; section 8.6

Stimulus: With metadata, subscriptions, and events present, deliver oadrCancelPartyRegistration by poll. Acknowledge the resulting oadrCanceledPartyRegistration.

Expected: The VEN replies through EiRegisterParty and erases its stored information about that VTN, including registration and report information. No subsequent routine exchanges use the canceled registration. A new registration attempt is permitted at a deployment-specific interval and must bootstrap as a new relationship.

R09 VEN cancels registration

P1 | C/M | Section 8.4.1; CR 401 and 407

Stimulus: Trigger local deregistration where exposed. Return success, and in a separate run lose the response after the harness has accepted cancellation.

Expected: The request contains the correct registrationID and venID and is sent to EiRegisterParty. The VEN handles the acknowledgment and any retry according to its documented recovery policy without continuing to report confirmed registered status after successful cancellation.

R10 Registration identity errors

P1 | S/R | CR 400 and 512

Stimulus: Send re-registration and cancellation controls for an unexpected venID or registrationID; separately return a reply identifying the wrong VTN. Change only one ID per run.

Expected: The VEN validates IDs and does not erase or replace the correct relationship based on another party's message. A 452 response is the CR 400 recommendation where applicable. Distinguish this identity check from certificate authentication, which is excluded.

Polling and dispatch

P01 Idle queue preserves protocol state

P0 | M | CR 500 and 502

Stimulus: Install an event and a periodic report, then return successful oadrResponse to multiple idle polls.

Expected: The VEN treats the response as no queued operation, sends no acknowledgment to it, and retains its event/report state. No events are canceled merely because no operation was returned. Contrast with the empty event snapshot in E08.

P02 Every legal poll response type

P0 | M/C | Section 8.6; CR 500; section 3 exchange map

Stimulus: Deliver each of the eight listed response types in a suitable state. Use R06/R08 for registration controls and D02/D03/D14/D21 for report operations.

Expected: The VEN routes each logical response to the correct service, supplies the correct payload, and handles its HTTP completion. Conditional report-consumer behavior is assessed only with a matching declared capability and valid subscription. Unsupported operations do not stall unrelated polling indefinitely.

P03 Drain a mixed service queue

P0 | S/M | CR 500

Stimulus: Queue an event snapshot, report request, and registration control; append another valid operation while the queue is being drained. End with successful oadrResponse.

Expected: The VEN should continue polling without waiting a full ordinary interval between queued operations, processing one payload per response. It returns to ordinary polling after the empty-queue response. The faster drain is permitted even when shorter than the requested poll interval.

P04 Sequential acknowledgment schedule

P0 | M | Annex C scenario 1

Stimulus: Return oadrCreateReport and then oadrDistributeEvent, providing prompt acknowledgments to the VEN's logical responses.

Expected: A VEN that acknowledges each operation before polling again completes both exchanges. The harness does not impose concurrency or require another poll before accepting a logical response.

P05 VTN waits for a pending logical response

P1 | C/R | Annex C scenario 2; CR 500

Stimulus: If the VEN polls with an earlier logical response pending, withhold the new poll response while accepting the earlier response on another request. Resume within the harness's bounded wait policy.

Expected: The outstanding poll does not prevent the VEN from sending its pending response, and normal operation resumes. A strictly sequential VEN need not create this race; record the vector Not applicable rather than failing lack of concurrency.

P06 Multiple logical operations in flight

P1 | C/M | Annex C scenario 3

Stimulus: If the VEN permits overlapping operations, return an event on a new poll before its prior report response has completed. Vary the order of HTTP completions.

Expected: Event and report responses retain their own IDs and destinations and neither is lost or confused with the other. Do not impose the A-profile-only CR 64 serial-processing restriction on a B-profile VEN.

P07 Poll frequency and jitter

P0 | M/S | Section 9.1.8; CR 403 and 500

Stimulus: Request different supported polling intervals at registration and re-registration. Measure idle cycles with configured jitter; separately configure a millisecond-resolution local interval.

Expected: The VEN supports millisecond configuration and configurable jitter. Ordinary polling should respect the VTN's requested interval, allowing longer intervals for processing constraints. Queue draining is excluded from this comparison. Report polling slower than 20% of average event duration as a CR 500 recommendation issue, not a new hard timeout.

P08 Poll errors and queue progress

P1 | M/R | CR 500 and application parts of CR 517

Stimulus: Return `oadrResponse/452` for an invalid VEN ID, then `oadrResponse/463` for lost registration. Separately queue several controls and trigger a recoverable failure on one follow-up.

Expected: Error responses are not interpreted as ordinary empty-queue success. The VEN exposes the failure and uses its documented identity/registration recovery. Independent work resumes after recovery without an unbounded poll loop. Recovery by re-registration is a test policy unless specifically required by another rule.

Event handling

E01 New event and initial opt decision

P0 | M | Section 8.1; CR 12, 35, 42 and 62

Stimulus: Deliver the baseline event with `responseRequired=always`. Run once with local `optIn` policy and once with local `optOut` policy.

Expected: The VEN sends `oadrCreatedEvent` with the correct `eventID`, `modificationNumber`, `requestID`, `venID`, success code, and selected `optType`. `eventResponses` is present; the enclosing `eiResponse` `requestID` is empty. A refusal to participate in a valid event is not itself a protocol failure.

E02 Events requiring no response

P0 | M | CR 12, 62 and 206

Stimulus: Deliver new, repeated, modified, and canceled events with `responseRequired=never`. Attempt a local event-specific `opt` qualification after the most recent such delivery.

Expected: Events are processed but no `oadrCreatedEvent` `eventResponse` is sent for them. The VEN also does not send event-specific `oadrCreateOpt` for an event whose latest response requirement is never. No logical response is invented merely to drain the queue.

E03 Mixed response requirements

P0 | M | CR 12 and 62

Stimulus: Send several events in one distribution, mixing always and never, and change one event's response requirement on a later valid delivery.

Expected: Responses include exactly the events requiring a response on that delivery. Combined and separate `oadrCreatedEvent` messages are both allowed. The VEN does not use a single response-required flag for the entire distribution or retain a stale requirement after it changes.

E04 Duplicate events and reused request IDs

P0 | M/R | CR 12, 42-43 and 62; section 9.1.4

Stimulus: Resend an unchanged event with always using both the same and a different distribution `requestID`. Then use a previously seen `requestID` for a different event or version.

Expected: Every always delivery receives a response, even if unchanged. The correct received `requestID` is echoed per event. Repeated delivery does not create a second schedule, and `requestID` reuse does not cause a different event to be discarded as a duplicate.

E05 Version advancement and stale versions

P0 | S/M | CR 4-5 and 57-58

Stimulus: Deliver E1/0, then a valid E1/1 update; simulate missed deliveries and send E1/3; finally send E1/1 again, including a stale cancellation variant.

Expected: Higher versions replace older versions without requiring receipt of every intermediate version. A lower version does not roll back accepted state; CR 58 recommends error 450. Every delivery marked always still receives appropriate application handling. Harness-generated consecutive changes increment versions correctly.

E06 Explicit cancellation

P0 | M/S | CR 10, 36, 52 and 59

Stimulus: Cancel pending and active events with an incremented modificationNumber and always. Repeat the cancellation once; after acknowledgment, remove it from future VTN snapshots.

Expected: The VEN acknowledges the cancellation; optIn means it confirms cancellation, while optOut means it cannot cancel. Inspect state to distinguish these outcomes. Once acknowledged, the VEN does not spontaneously include that canceled event in later created-event messages unless the VTN sends it again.

E07 Cancellation of an unknown event

P0 | M/S | CR 21 and 60

Stimulus: Send an unknown eventID with status cancelled, first with always and then with never in an independent run.

Expected: The VEN does not return an invalid-event-ID error solely because the event was unknown. With always it still acknowledges the event; with never it does not respond. The unknown canceled event should not create an active schedule.

E08 Implied cancellation and empty snapshot

P0 | M | CR 50 and 61; CR 502

Stimulus: Install E1 and E2. Send a complete snapshot containing only E2, then a schema-valid distribution with zero events. Compare with P01's empty-queue response.

Expected: Missing events are removed from the known set. An active event that cannot immediately stop may be retained under CR 61's operational exception; record that reason. A zero-event distribution clears applicable known events. The harness must not accidentally omit unchanged events in normal cases.

E09 Lifecycle without status notifications

P0 | M/C | CR 31-32, 66 and 114-115

Stimulus: Deliver a future event once, then return only empty-queue responses through its start, interval transitions, and end. Also deliver an already active but not expired event after delayed communication.

Expected: The VEN maintains the correct time-based schedule and handles the current interval according to its declared event processing. A B VEN may calculate status instead of trusting the transmitted status, except that cancelled must be honored. Do not demand a new VTN message for each transition.

E10 Event interval semantic validation

P1 | M/R | CR 2, 8, 9, 33, 100, 102-105, 107 and 116

Stimulus: Independently introduce incorrect interval sums, missing/nonsequential UID, duplicate signalID, interval dtstart, extra signalPayload, SIMPLE outside 0-3, SIMPLE type other than level, or omitted x-eiNotification. Keep the outer XML schema-valid when possible.

Expected: Invalid logical content is detected without accepting a corrupt schedule; use suitable application errors for schema-valid faults. The harness distinguishes schema failures from semantic failures. Do not require an exact 450/454/459 code unless the governing rule does.

E11 Randomized start and duration

P1 | M/S | CR 30-31

Stimulus: Deliver an event with positive startafter repeatedly under independent reset conditions. Update its future signal values without changing startafter, then separately change the randomization window.

Expected: The selected start lies between dtstart and dtstart plus startafter; completion is measured from the randomized start. The same offset should persist across modifications unless startafter changes. Use internal schedule observations or controlled protocol telemetry, not assumptions from receipt time.

E12 Randomized termination and completed status

P1 | M/S | CR 65-66

Stimulus: Explicitly and implicitly cancel active randomized events. Separately deliver completed status while the VEN's randomized execution period has not yet ended.

Expected: Cancellation termination is randomized; the recommended termination window is zero to startafter. Receipt of completed alone must not prematurely terminate the randomized execution. Validate the harness's inclusion of the event through dtstart plus duration plus startafter.

E13 Minimum B signals and multiple signals

P0 | M | CR 104, 106-107, 116 and 510; Table 1

Stimulus: Test SIMPLE/level, ELECTRICITY_PRICE/price with currency-per-kWh units, and LOAD_DISPATCH/setpoint with declared power units separately and in multi-signal events. Use unique signalIDs and multiple intervals.

Expected: The full VEN accepts the mandatory supported combinations and preserves each signal's values, units, scale, and identity. It does not treat an unfamiliar signalID as an unsupported signalName. Do not infer a universal physical relationship between SIMPLE and other simultaneous signals.

E14 Unsupported signal combinations

P0 | M | CR 48 and 109

Stimulus: Send a recognized but unsupported signalType/unit combination. Then include one unsupported signal alongside a supported signal in the same event. Use always.

Expected: The event is rejected with application 460 and optOut when any requested signal combination cannot be supported. A supported signal does not make the whole event acceptable. Test an unrecognized name separately with an appropriate logical error, without conflating unknown names and recognized unsupported combinations.

E15 Event level targets

P1 | M/C | CR 21-22 and 48; section 8.2.1

Stimulus: Test empty eiTarget, matching and nonmatching venID/resourceID, each additional declared target type, and multiple target criteria with one match. Include an ID case mismatch.

Expected: Empty target identifies the recipient. Multiple criteria are ORed for determining a match. A completely nonmatching target produces logical error and optOut. Respect the implementation-dependent response behavior noted in CR 22 and declared resource mapping; do not require support for every optional target type.

E16 Device class targeting and unresolved resources

P1 | M/C | CR 111-112

Stimulus: Combine event-level targets with signal-level endDeviceAsset classes. Use several classes with one match, then classes excluding the event-targeted resources. Separately create a signal that cannot be assigned unambiguously to a resource.

Expected: Classes are ORed within the signal and ANDed with event-level targeting. The unresolved-resource condition in CR 112 returns 469. Positive fixtures use only endDeviceAsset for signal-level eiTarget and omit that target element when no class is present.

E17 Market context matching and normalization

P1 | M/C | CR 48 and 512

Stimulus: Send the configured marketContext, an equivalent URI with scheme/hostname case changes, and a different context. Repeat if the VEN supports a declared accept-any-context configuration.

Expected: URI normalization recognizes equivalent contexts without indiscriminately lowercasing case-sensitive path components. A mismatching context causes a logical error and optOut unless the accept-any configuration is enabled. Record that configuration as part of the verdict.

E18 Mixed valid and invalid events

P0 | M/S | CR 19 and 48

Stimulus: Put one valid event, one target mismatch, and one unsupported signal event in a single schema-valid distribution, all marked always.

Expected: The valid event is processed rather than rejecting the entire distribution. Check appropriate per-event success/opt status and logical errors with optOut for invalid events. CR 48 recommends overall success with per-event results when the distribution itself has no error. Do not apply this oracle to an unparseable whole XML document.

E19 Updates to active and pending events

P1 | M/S/R | CR 5, 20, 33 and 57

Stimulus: Change future start, duration, or signal intervals while pending; change only future intervals while active. Then send a schema-valid update attempting to change already elapsed content.

Expected: Valid higher-version updates replace the relevant schedule. Past-changing updates are diagnosed as illogical and do not silently rewrite observed history. Distinguish VTN fixture violations from the VEN's error handling; an exact 451 code is not imposed without a rule requiring it.

E20 Adjacent events and overlapping programs

P1 | M/C | CR 3, 15 and 18

Stimulus: Send adjacent events and then overlapping events with different priorities, including priority zero and randomized effective end times. Supply a declared program policy for overlaps.

Expected: Adjacent boundaries use inclusive starts and noninclusive ends. Priority zero means no priority, not highest priority. Validate overlap behavior against the program policy, since the specification does not define a universal VEN control outcome. A normal harness snapshot follows the VTN ordering rules.

E21 Open ended events

P1 | M | CR 47

Stimulus: Deliver an event whose overall duration is zero, observe beyond its start, and then explicitly cancel it.

Expected: Zero overall duration means no defined end, not immediate completion. The event remains active until cancellation, subject to its opt state. The VEN neither schedules a zero-length completion nor needs periodic redistributions to preserve the event.

E22 Optional current values and descriptive fields

P1 | M/C | CR 14, 29, 46, 110 and 113

Stimulus: Omit currentValue from valid B events; then include correct active currentValue, modificationDateTime, modificationReason, notification, and other supported optional fields. For nonactive SIMPLE use zero; vary nonactive currentValue for other signals.

Expected: Omission of optional B currentValue is accepted. Included values are interpreted consistently with the signal and schedule. Do not apply SIMPLE's nonactive-zero restriction to other signal types. Validate protocol state without requiring physical ramp-up/recovery behavior, which CR 31 identifies as untestable there.

E23 Test event string semantics

P1 | M | CR 6; eventDescriptor XSD

Stimulus: Deliver otherwise identical events with testEvent absent, "false", "true", "0", and another nonempty string; include an empty-string vector under the agreed CR 46 interpretation.

Expected: Any present string other than exact "false" triggers test-event treatment under CR 6. Do not interpret "0" as a Boolean false. Observe the VEN's test-event classification without prescribing a physical actuation policy. Treat the empty-versus-omitted interaction as an interpretation item if necessary.

E24 Baselines and additional supported signals

P1 | C/M | CR 100-104 and 108; Table 1

Stimulus: For declared baseline support, send multiple baseline intervals, optional resourceIDs, and baseline duration independent of event duration. For every additional declared signal, exercise its permitted type/unit combinations and boundary values.

Expected: Baseline intervals sum to baseline duration and use correct UIDs and float payloads without interval dtstart. Multiple baseline resources represent an aggregate, not an inferred copy of event targeting. Additional signal values follow Table 1 or the declared custom-signal contract.

Reporting

D01 Advertise the VEN report catalog

P0 | M | CR 300-301, 305, 308-313, 331, 333 and 510

Stimulus: Capture oadrRegisterReport after new registration/reset. Compare it with the declared resource/report model.

Expected: It advertises all capabilities with unique reportSpecifierIDs and rIDs within each specifier. Metadata names have METADATA_ prefixes; reportDescription, duration, required units, and telemetry sampling rates are present. oadrMaxPeriod does not exceed the offered duration. No data intervals or dtstart occur in metadata. Each contained reportRequestID is "0". Required full-VEN status and usage test capabilities are not waived by the no-report fallback.

D02 Receive the VTN report catalog

P0 | M | CR 301, 305-306 and 309

Stimulus: Return oadrRegisterReport by poll, first with no oadrReport objects and then with a valid VTN metadata catalog in another run.

Expected: The VEN posts oadrRegisteredReport to EiReport and stores the received capabilities as applicable. An empty catalog is a legitimate VTN declaration of no reportable data, not malformed XML. Metadata reception and acknowledgment do not require that the VEN consume ordinary VTN data reports.

D03 One shot metadata request

P0 | M | CR 301, 308, 311, 324 and 327

Stimulus: Send oadrCreateReport with reportSpecifierID="METADATA", rID="0", reportBackDuration=PT0S, reportRequestID MR1, and requestID MREQ1. Repeat using another rID value to verify it is ignored for metadata requests.

Expected: The VEN sends oadrCreatedReport and a current oadrRegisterReport. The top-level reportRequestID is MR1; each contained metadata reportRequestID remains "0". Unsolicited registration omits the top-level reportRequestID. Response requestIDs correlate, and metadata is not returned as ordinary oadrUpdateReport data.

D04 Piggyback report requests

P0 | M | Section 8.3.2.1; CR 344

Stimulus: Respond to the VEN's oadrRegisterReport with oadrRegisteredReport containing one or more valid oadrReportRequest objects. Repeat without such a request.

Expected: The VEN acts on embedded requests, posts oadrCreatedReport, and starts the accepted reports. Receiving this form is mandatory even if the VEN never sends it. With no request, it does not invent a data subscription merely because a capability was advertised.

D05 Telemetry status data

P0 | M | CR 313, 331, 334 and 510

Stimulus: Request TELEMETRY_STATUS for an advertised resourceID. Toggle the test resource's online and manual-override states. If offered, change load-control-state values.

Expected: Reports contain oadrPayloadResourceStatus with correct oadrOnline and oadrManualOverride, matching times and IDs. Status metadata uses the CR 331 reportType and readingType and no fabricated measurement units. If load-control children appear, the same set occurs in every interval in a report.

D06 Periodic usage reports

P0 | M/S | CR 300, 313-317, 324-325 and 331

Stimulus: Request an advertised TELEMETRY_USAGE data point at supported granularity G, reportBackDuration B, and finite reporting duration. Drive a known sequence of readings. Repeat with overall reporting duration zero and later cancel explicitly.

Expected: oadrCreatedReport lists scheduled reports. oadrUpdateReport carries the requested points, chronological intervals, IDs, and values. Delivery follows B and sampling follows G within agreed tolerances. Use CR 317's interval-count guidance, not an unconditional exact B/G count. Finite reporting should end after the requested period, allowing final-report timing tolerance; zero overall duration continues indefinitely until canceled.

D07 One shot point and interval telemetry

P0 | M | CR 324 and 337

Stimulus: Request each supported telemetry kind with reportBackDuration=PT0S, varying granularity and supplying or omitting reportInterval for point data.

Expected: One report is sent immediately, containing one interval of data for each requested point; granularity is ignored. Point-data one-shots ignore an included reportInterval. Interval usage uses the latest buffered sample and includes interval duration when required by CR 337. The request does not become periodic.

D08 On change reporting

P1 | C/M/S | CR 317-318

Stimulus: For a supporting data point, request granularity=PT0S with nonzero reportBackDuration. Change its value, hold it constant across a reporting period, and change it rapidly.

Expected: Change-based data is handled according to CR 317's recommendation, with regular report delivery at reportBackDuration. Reported values are not sampled more frequently than the metadata minimum period. With no new value, the data-quality field indicates "No New Value - Previous Value Used".

D09 Subsets and concurrent report requests

P1 | M | CR 300, 303, 322 and 329

Stimulus: Request different subsets of rIDs from one reportSpecifierID under distinct reportRequestIDs; include multiple requests in one payload and different reportBackDurations.

Expected: Each accepted subscription preserves its own parameters and IDs. Reports contain all requested points and maintain correct association. oadrPendingReports lists all reports scheduled for future delivery, not just the latest request. Empty requestID is permitted; nonempty requestID is echoed.

D10 Unknown reporting identifiers

P1 | S/M/R | CR 300, 304 and 336

Stimulus: Separately request an unknown reportSpecifierID, an unadvertised rID, or use an unexpected reportRequestID/venID in an operation that requires an existing association.

Expected: The VEN does not produce a fictitious successful report. ID validation and a 452 response follow CR 304's recommendation; a suitable application error identifies unsupported or invalid requests. A recognized report not supported in the current configuration is distinguished from an unknown report identity.

D11 Capacity and invalid reporting intervals

P1 | M/R | CR 312, 317 and 340

Stimulus: Request a duration exceeding offered data capacity; separately request granularity below oadrMinPeriod, granularity above reportBackDuration, and a periodic request missing required timing fields.

Expected: Excess requested data returns oadrCreatedReport with application 454 as CR 340 requires. Other semantic faults receive appropriate error handling without silently creating an impossible schedule. Do not treat invalid harness requests as normal interoperability fixtures.

D12 Report interval and payload structure

P1 | M | CR 313-316, 319, 331 and 342

Stimulus: Inspect point and interval reports with one and several samples; exercise supported explicit-time and inherited-time encodings.

Expected: Data reports have dtstart, intervals, and the non-METADATA reportName, and omit reportDescription. Explicit interval times and durations satisfy the all-or-none rules. UIDs start at zero when interval dtstart is absent. Point/interval combinations obey CR 342; report duration, if present, spans the data. Accuracy, if included, uses the value's units.

D13 Unavailable data and quality

P1 | M | CR 300, 318 and 319

Stimulus: Remove a reading for a data point that was advertised and requested, then restore it. Inject a known bad-quality interval through the measurement adapter.

Expected: The VEN includes a placeholder for the missing requested point with an appropriate Quality Bad value rather than omitting it or reporting an unqualified good reading. Subsequent valid data is reported with correct quality and timestamps. Values and units remain consistent.

D14 Cancel periodic reports

P0 | M | Section 8.3.2.3; CR 329 and 336

Stimulus: Schedule at least two reports; send oadrCancelReport with reportToFollow=false for one and then for multiple valid reportRequestIDs.

Expected: The VEN posts oadrCanceledReport with a correct pending list. Canceled reports stop, unrelated reports continue, and no final additional report is required. Account for an already completed/in-flight report at the cancellation boundary instead of counting it as newly scheduled output.

D15 Final report and piggyback cancellation

P0 | M | CR 328, 335 and 345

Stimulus: Cancel a periodic report with reportToFollow=true. Repeat with cancellation embedded in oadrUpdatedReport responding to a VEN update; reference only IDs in that update.

Expected: The VEN processes both forms and sends one final additional report when requested, then stops the subscription. If cancellation was not honored, the harness repeats with a standalone oadrCancelReport as CR 335 directs. Verify the final report is additional to the update that triggered piggyback cancellation.

D16 Cancellation with invalid report IDs

P1 | M/R | CR 336

Stimulus: Cancel a set containing one valid and one unknown reportRequestID, or make one report uncanceled through supported instrumentation.

Expected: oadrCanceledReport contains a 4xx if all requested cancellations cannot succeed. Its pending list reflects actual remaining schedules. Do not invent an atomic-all-or-nothing requirement; document which valid reports were canceled and preserve consistency on subsequent exchanges.

D17 Metadata replacement and subscription invalidation

P1 | M | Section 8.3.2.1; CR 306

Stimulus: Establish data subscriptions, then exchange a changed complete metadata catalog in each supported logical direction. Re-request desired reports after the replacement.

Expected: The recipient replaces the previous catalog and invalidates its existing data subscriptions against that catalog, preserving periodic metadata requests. Thus a VEN catalog refresh requires the VTN to re-request VEN data; a VTN catalog refresh requires the consuming VEN to re-request VTN data. No stale rID/reportSpecifierID association survives. Do not implement catalog replacement as a partial merge.

D18 Periodic metadata requests

P1 | M | CR 306, 324 and 327

Stimulus: Request METADATA with nonzero reportBackDuration. Allow multiple deliveries, replace metadata, and then explicitly cancel the periodic metadata request.

Expected: The VEN delivers metadata periodically using oadrRegisterReport with the proper request association. Periodic metadata requests survive implicit cancellation caused by metadata replacement and stop after explicit cancellation. Apply the CR 510 no-report periodic-metadata exception only to the VTN, not generally to the VEN.

D19 Historical usage reports

P1 | C/M | CR 317, 324-325, 340-341 and 510

Stimulus: If HISTORY_USAGE is supported, request a known past window with granularity=PTOS and reportBackDuration=PTOS. Repeat with duration zero, then a window beyond available history.

Expected: Return a one-shot chronological report at stored granularity; zero duration means all history from dtstart. Excess data requests receive 454. HISTORY_USAGE is mandatory for a Report Only VEN, conditional for a full VEN. Do not substitute forecast samples for requested historical data.

D20 Recent data buffering

P0 | M/R | CR 510

Stimulus: Accumulate at least 100 timestamped data points through the test data source. Interrupt delivery temporarily, restore communication, and request the retained recent window through the supported reporting interface.

Expected: At least 100 recent data points are recoverable with correct timestamps and values. Use internal evidence if the interface cannot expose the entire buffer in one request. Separate this storage requirement from optional HISTORY_USAGE support. Any undisclosed loss within advertised capacity is a failure of the applicable storage assertion.

D21 VEN consumes VTN reports

P1 | C/M | Section 8.3.2; CR 303-304 and 328-329

Stimulus: With a declared report-consumer capability, advertise a VTN report, have the VEN request it, deliver oadrUpdateReport by poll, and then have the VEN cancel it. Also exercise a VEN piggyback request if it can send one.

Expected: The VEN sends requests/cancellation to EiReport, accepts valid associated data, and posts oadrUpdatedReport for polled updates. Correlation survives separate HTTP transactions. Do not send unsolicited data and label its rejection a failure of the normal consumer path.

D22 Corrected and repeated report data

P1 | C/S/R | CR 339, 341 and 346

Stimulus: Delay a measurement or initially supply bad-quality data, then provide a corrected value. If corrected-report generation is supported, observe an early additional report. For a report-consuming VEN, resend and correct a VTN report.

Expected: Corrected data replaces the corresponding previous values and uses the original request association and interval structure. Duplicate report delivery alone is not a protocol violation. eiReportID is not used as a required OpenADR correlation key; it should be ignored. Normal reporting remains predictable within allowed timing variation.

Opt operations

O01 Event specific opt qualification

P0 | M | CR 206-208 and 510

Stimulus: Deliver an always event, receive initial oadrCreatedEvent, then trigger a resource opt-state change through the VEN interface.

Expected: The full VEN can send oadrCreateOpt with optID, venID, requestID, optType, reason, correct qualifiedEventID including modificationNumber, and target. It omits vavailability and marketContext when eventID is present. The VTN returns oadrCreatedOpt with matching correlation.

O02 Resource opt precedence survives updates

P0 | M/C | CR 206

Stimulus: Qualify RES_1 using oadrCreateOpt, then change the broader event opt state via oadrCreatedEvent, modify the event, and finally change RES_1 again.

Expected: Once qualified, RES_1 is changed only through another oadrCreateOpt for that resource. Broad created-event changes may still affect unqualified resources. Event modification does not reset this precedence. Use a multi-resource fixture only when supported.

O03 Latest response requirement controls opt changes

P0 | M | CR 12, 42 and 206

Stimulus: Change a local opt decision after an always delivery, then distribute a new version marked never and attempt another change. Subsequently restore always in a later valid version.

Expected: Allowed asynchronous changes reference the appropriate event version and received requestID for created-event responses. No event-specific oadrCreateOpt or oadrCreatedEvent is sent while the latest requirement is never. An initial always acknowledgment is never replaced by only oadrCreateOpt.

O04 Opt targets and rejected qualification

P1 | M/C | CR 206 and 209

Stimulus: Qualify all resources, a valid subset, and device-class-filtered resources. Return an application 4xx to an attempted invalid subset or disallowed opt change.

Expected: No target or only venID applies to all event resources; a more specific target is a subset of the event targets. Device classes use the allowed encoding. The VEN recognizes VTN rejection and does not report the rejected state as confirmed.

O05 Availability schedules

P1 | C/M | CR 200-202, 205 and 210

Stimulus: If schedules are supported, create optIn and optOut availability windows, an open-ended zero-duration window, and targeted/market-context-qualified schedules permitted by the program.

Expected: Without eventID, oadrCreateOpt includes vavailability and may include marketContext. It excludes tolerance, notification, ramp-up, and recovery in availability properties. Empty target covers all VEN resources; multiple target criteria are ORed. Zero duration is open ended.

O06 Schedule replacement and overlap policy

P1 | C/M | CR 201 and 203

Stimulus: Create a schedule, replace future availability for all resources, and then replace it for a subset. Add overlapping optIn/optOut schedules under the declared program policy.

Expected: The harness models the specified replacement scope; unrelated resources retain their schedules. VEN messages reflect the requested local changes. Default availability and overlap precedence follow the configured program contract, not an invented universal rule. Keep optIn and optOut schedules independent for replacement.

O07 Cancel schedules and reuse opt IDs

P1 | C/M | CR 204, 207 and 211

Stimulus: Cancel a schedule, cancel an already superseded schedule, and reuse one optID for several target sets before canceling that ID. Attempt cancellation of an optID used for an active/pending event qualification.

Expected: Valid schedule cancellation uses oadrCancelOpt/oadrCanceledOpt and matching IDs. A reused optID covers all associated resources. Superseded-schedule cancellation is not a VTN error. The VEN must not use oadrCancelOpt to undo an event qualification; another oadrCreateOpt changes that state.

O08 Opt response errors and idempotence

P1 | M/R | CR 207-208; section 9.1.4

Stimulus: Return a mismatched optID or requestID, an explicit application rejection, and a lost response after accepting an opt creation. Use empty and nonempty requestIDs in valid runs.

Expected: Nonempty requestIDs are correlated; empty ones remain permissible. Invalid or rejected replies do not confirm a different operation. A retry does not create conflicting availability state. Exact recovery and UI presentation are checked against the product contract where the profile is silent.

Recovery and sustained operation

F01 Reboot and new registration recovery

P1 | M/S/R | CR 301, 309, 401 and 405-407

Stimulus: Reboot with pending events and periodic reports. Run with retained VEN configuration, then with factory-cleared protocol state. Let the VTN retain or clear its state in separate runs.

Expected: An unregistered configured VEN attempts registration. A new relationship exchanges metadata and synchronizes events; a retained registration follows the documented re-registration policy. No duplicate active schedules appear. Persistent storage details are product policy, not an assumed protocol requirement.

F02 Lost acknowledgment and repeated controls

P1 | M/R | Section 9.1.4; CR 12, 62 and 346

Stimulus: Lose acknowledgments after committed event responses, report creation, report cancellation, and opt changes. Redeliver the corresponding VTN controls after timeout where appropriate.

Expected: The system tolerates repeated delivery without contradictory state, while still acknowledging every event marked always. Repeated reports are allowed. Do not confuse business-state idempotence with suppressing protocol responses that the specification requires.

F03 Outage across event and report boundaries

P1 | M/R | CR 31-32, 61, 325 and 510

Stimulus: Disconnect during an active event and periodic reporting, let the event end or be canceled at the VTN, then restore service and provide an authoritative snapshot and report controls.

Expected: Polling recovers without request flooding; the VEN reconciles to received state and retained data is handled within declared capacity. Observe the documented local behavior during disconnection without assigning an electrical fail-safe policy to OpenADR. Account for missed events and samples explicitly.

F04 VTN state loss and rejected associations

P1 | R/M | CR 401-405, 500 and application parts of 517

Stimulus: Clear the harness registration or report association while the VEN remains running; return 463 or an appropriate ID error. Make the VTN ready to accept a new relationship afterward.

Expected: The VEN does not continue treating rejected transactions as successful. Its documented recovery reaches a consistent registration and subscription state and performs bootstrap if newly registered. Automatic re-registration on every 463 is not asserted as a universal MUST.

F05 Mixed service endurance

P2 | R | Engineering acceptance criterion

Stimulus: Run a 24-hour mixed workload of polling, events, report updates, opt changes, metadata requests, and occasional recoverable failures at supported rates. Use bounded outstanding work and seeded repeatable stimuli.

Expected: No deadlock, unbounded retry queue, lost acknowledged state, or monotonically growing memory/connection use beyond agreed limits. Preserve per-operation ordering and correlation. Record latency distributions and resource trends; the 24-hour duration is a proposed campaign target, not a profile requirement.

F06 Declared capacity boundaries

P2 | C/R | XSD cardinalities; product limits

Stimulus: Exercise zero, one, typical, maximum declared, and one-beyond-limit counts for events, signals, intervals, report requests, data points, and XML body size. Include concurrent polling and report delivery near capacity.

Expected: Supported limits function correctly; excess load is handled without silent truncation, partial success misreporting, starvation, or corruption. No arbitrary finite limit is derived from an XSD unbounded cardinality. Document throughput and response-time acceptance values before judging performance.

6 Execution order and completion criteria

First commission the harness and capability declaration, then run a smoke sequence: R02, R03, P01, E01, E02, E06, D01, D03-D07, D14-D15, O01, and R06. Include T01-T03 and X01-X02 assertions continuously. A successful smoke run proves that each essential service has a working exchange path.

Next run all applicable P0 and P1 groups, starting with valid behavior and then semantic errors, duplicates, and transport failures. Execute conditional feature vectors for every capability declared or advertised. Finish with P2 malformed-input, endurance, and limit testing. Repeat a compact mixed-service sequence after fixes to shared parsing, correlation, transport, or scheduling code.

Family	Scenario groups	Primary requirement coverage
HTTP transport	T01-T12	Section 9.1; CR 37, 51, 53, 511
XML and common rules	X01-X08	XSD; CR 1, 21, 33, 43, 45-48, 508-509, 512
Registration	R01-R10	Section 8.4; CR 400-407
Polling	P01-P08	Section 8.6; CR 500-502; Annex C
Events	E01-E24	Sections 8.1-8.2; applicable CR 1-66 and 100-116; CR 510
Reporting	D01-D22	Section 8.3; applicable CR 300-346; CR 510
Opt operations	O01-O08	Section 8.5; CR 200-211; CR 510
Recovery and endurance	F01-F06	Cross-service requirements and declared engineering targets

Completion requires every applicable mandatory assertion to pass, all conditional features to have a documented applicability decision, and every failure to retain reproducible evidence. No unexplained Inconclusive or Blocked result may be counted as passed. Resolve or explicitly disposition SHOULD departures and product-policy deviations. Record remaining gaps, especially state assertions unavailable through wire observation.

The final result should state the tested VEN build and configuration, counts of expanded vectors and verdicts, covered service paths, defect summary, and excluded capabilities. Report mandatory conformance findings separately from interoperability recommendations and robustness observations. Do not report full security conformance or Alliance certification from this campaign.

7 Implementation guidance for the case runner

Represent each expanded case as data: case ID, vector ID, applicability predicate, priority, rule references, initial state, VTN actions, VEN stimuli, expected message predicates, forbidden messages, timing bounds, state

observations, cleanup, and evidence paths. Use predicates over XML values, not whole-document byte equality. Preserve original bytes for auditing.

Correlate event responses by eventID, modificationNumber, and the received requestID; correlate reports by reportRequestID, reportSpecifierID, and rID; correlate opt operations by optID and requestID. Maintain a separate HTTP transaction identifier. A single global "last request" variable is insufficient when polling and reporting overlap.

The harness should enforce its own VTN duties during every positive scenario, including correct snapshot contents, valid response payloads, supported report selections, and accurate identifiers. Label tests that intentionally violate a VTN rule as negative VEN robustness or semantic-validation tests. The verdict then evaluates the VEN's specified response, not whether the invalid VTN behavior was conformant.

For reproducibility, save the exact serialized fixture after all dynamic IDs and timestamps have been resolved. For timing tests, retain both UTC and monotonic timestamps. For local opt or measurement changes, log the stimulus and the moment the adapter confirms it. Keep physical load response out of the oracle unless a later test campaign explicitly adds it.

8 Source baseline and interpretation decisions

Documents and schema

Primary reference: OpenADR_2_0b_Profile_Specification_v1_1.pdf, OpenADR Alliance, 2015, 110 pages.

References in this plan use its printed section numbers, figure numbers, and CR identifiers. Key pages are 23-50 for service exchanges, 51-54 for HTTP, 66-104 for conformance, and 108-109 for pull sequencing.

Schema reference: Schema_oadr_2_0b.zip. The validation entry point is the top-level oadr_20b.xsd with its packaged imports, including oadr_ei_20b.xsd and oadr_pyld_20b.xsd. That entry point compiled successfully during preparation of this plan. This verifies the schema dependency set, not execution of the proposed VEN tests.

The bundle also contains an older ZIP archive named oadr_20b_Schema_061813.zzz. Do not recursively unpack it over the selected schema set. The supplied top-level schema differs from that archive, including disabled Green Button imports and adapter declarations. Green Button extension testing is excluded from this baseline unless a separately agreed schema set enables it.

Decisions that prevent false failures

1. Use the XSD spelling simpleHttp on the wire. Prose uses simpleHTTP in places. Use ELECTRICITY_PRICE from the schema and Table 1; CR 510 contains a spelling error in that signal name.
2. XSD is necessary but insufficient. schemaVersion and some conditional fields are optional in XSD yet required by conformance rules. Conversely, never add a field to a message that its schema cannot carry merely because a general rule appears broader. oadrQueryRegistration has only requestID; initial registration has explicit ID exceptions.
3. Preserve SHOULD versus MUST. Immediate poll draining, some ID-error codes, and parts of version recovery are recommendations. Longer-than-requested polling is explicitly allowed for constraints; the 20%-of-event-duration guidance is not a hard protocol deadline.
4. CR 301 calls for metadata on re-registration, while CR 405 explicitly says metadata exchange and oadrRequestEvent are not required for re-registration. Accept re-registration without forced bootstrap as the CR 405 path; record the interpretation and test repeated metadata if the product sends it. Do not silently turn this inconsistency into a product failure.

5. CR 19's mixed-event language and CR 48's per-event error guidance must be read together. Preserve valid events and evaluate invalid-event diagnostics; do not reject an entire schema-valid distribution just because one event is logically invalid. `responseRequired=never` still suppresses event responses.
6. Use Figure 22's empty HTTP 200 completion for a VEN-posted re-registration acknowledgment. Generic poll-response prose does not justify responding forever to an `oadrResponse` with another `oadrResponse`.
7. Avoid truncated event snapshots in ordinary tests. CR 27 allows `replyLimit` on `oadrRequestEvent`, while CR 50/61/502 describe complete snapshots and implied cancellation. Omit `replyLimit` for normal synchronization. If the product uses a limiting value, test that mode separately and disposition the snapshot interaction before issuing a mandatory verdict.
8. Treat metadata as a complete catalog, and preserve the periodic-metadata exception in CR 306. Section 8.3.2.1 has potentially confusing source/target wording about cancellation; record both report directions explicitly and apply the conformance rule to the schedules affected by replacement.
9. The exact behavior of overlap, offline physical control, some malformed-response errors, and recovery after rejected registration is not universally prescribed. Use declared product/program policies for those assertions and label them accordingly.

Reproducibility fingerprints

SHA-256 values identify the exact baseline used to prepare this plan.

OpenADR_2_0b_Profile_Specification_v1_1.pdf

80c844e0900881f6c63ccb93f92e2a5f6bdb2814ababcfe2f6eba5ce65997ef7

Schema_oadr_2_0b.zip

2eb63b0c230ba01013826dc1f85420fca3bd721d9a0c18795f5d2984955f9b46

oadr_20b.xsd

96060f01ab3d890e16866d4bfa9f98094d9ac54dc3915f423216ecc89ccc1d03