e-QR Technical Specification v0.1

Pan-European A2A Payment Initiation for SEPA Instant Credit Transfers (SCT Inst)
QR Carrier Profile aligned with EPC MSCT QR standardisation

Document name	e-QR Technical Specification
Version	v0.1 (Draft for consultation)
Date	2026-01-12
Primary use case	Merchant-presented A2A initiation of SCT Inst at the POI
Primary carrier	QR (URL-based, EPC MSCT-aligned)
Owner	e-QR Payment Standard MTÜ

Version history

Version	Date	Change
v0.1	2026-01-10	Initial consultation draft

1. Introduction

e-QR is an interoperability specification for initiating SEPA Instant Credit Transfers (SCT Inst) using Mobile Initiated SEPA (Instant) Credit Transfer (MSCT) patterns. It standardises how merchant-presented data (QR in v0.1) is encoded, routed, and resolved so that payer applications can consistently prefill an SCT Inst initiation across PSPs and merchant solutions.

e-QR does not create new payment rails. Execution, authentication (including Strong Customer Authentication where applicable), risk controls, and settlement remain with regulated PSPs and the SCT Inst scheme.

1.1 Scope

- A QR carrier profile aligned with EPC MSCT QR-code standardisation (URL-based format with version, type and routing identifier).
- Two payload profiles for merchant-presented initiation at the POI: proxy-based (MID) and token-based (tok).
- A resolver interface (HTTPS) that resolves proxy/token into verified merchant and transaction details for SCT Inst prefill.
- An Operator Directory (trust anchor) published by the Governance Authority for host/OPID verification and key distribution.
- Security, privacy and conformance requirements sufficient for independent implementations to interoperate.

1.2 Out of scope

- Bank/PSP internal payment initiation APIs (e.g., PSD2/PSR interfaces).
- Inter-PSP hub messaging for MSCTs where deployed by a scheme/framework.
- Commercial terms, fees, or enforcement procedures.
- Detailed merchant onboarding procedures beyond the minimum integrity requirements for registry records.

1.3 Conformance language

The key words MUST, MUST NOT, SHOULD, SHOULD NOT, and MAY are to be interpreted as described in RFC 2119 and RFC 8174 when, and only when, they appear in all capitals.

2. Normative references

- EPC024-22 v2.10 - Standardisation of QR-codes for MSCTs.
- EPC269-19 v3.0 - MSCT interoperability guidance.
- EPC193-22 v1.2 - Specification of QR-codes for mobile (instant) credit transfers in ISO format.
- RFC 2119 / RFC 8174 - Requirement levels.
- RFC 7515 - JSON Web Signature (JWS).
- RFC 7517 - JSON Web Key (JWK).
- RFC 8785 - JSON Canonicalization Scheme (JCS).
- RFC 3339 - Date and time formats (for timestamps).
- ISO 13616 - IBAN (format).

3. Definitions and acronyms

Term	Definition
A2A	Account-to-Account payment using SCT/SCT Inst rails.
ASPSP	Account Servicing Payment Service Provider.
MSCT	Mobile Initiated SEPA (Instant) Credit Transfer.
POI	Point of Interaction.
SCT Inst	SEPA Instant Credit Transfer.
Governance Authority	Entity maintaining the specification and trust anchor (Operator Directory).
Operator	Accredited entity operating resolver/registry endpoints in the e-QR interoperability domain.
OPID	3-character Operator ID used for routing (aligned with EPC MSCT service provider ID).
Payload issuer (PI)	3-character identifier of the entity issuing payload content under an Operator (e.g., POS vendor).
MID	Merchant Identifier used as proxy for resolving verified merchant identity and payee account details.
Token (tok)	Opaque identifier resolving to a specific transaction request (typically short-lived and single-use).

4. Alignment with EPC MSCT QR standardisation

EPC MSCT QR standardisation defines a URL-based QR structure with the following segments: domain, version, type, routing identifier (service provider ID), and payload carried as a query string.

e-QR adopts this structure and profiles it for SCT Inst initiation at the POI (type '/m/'). For POI contexts ('/m/'), this profile restricts clear-text payee account details in the QR; proxy/token approaches reduce tampering and privacy risks.

5. Architecture and roles

High-level process:

- Merchant presents a QR code containing routing (OPID) and minimal payload (proxy MID or token).
- Payer application validates the QR against the signed Operator Directory (host + OPID trust check).
- Payer application resolves merchant/transaction details from the responsible Operator resolver endpoint.
- Payer application initiates SCT Inst with the payer's PSP using resolved details as prefill inputs.
- Settlement occurs on SCT Inst rails; e-QR components do not clear or settle payments.

Role	Responsibilities
Governance Authority	Publishes the specification; accredits Operators; publishes a signed Operator Directory (trust anchor).
Operator (resolver/registry)	Hosts resolver endpoints; maintains MID registry integrity; manages OPID/PI namespaces under governance; publishes signing keys; signs resolver responses.
Payload issuer	Generates payload content (e.g., POS vendor or merchant system) under an Operator; identified by PI.
Payer application	Parses, validates and resolves QR payload; presents transaction to user; initiates SCT Inst via regulated PSP channels.
Merchant systems	POS/checkout components that request tokens (dynamic) or render proxy QRs and display amount at POI.

5.1 Registry Integrity and Signing Model

The Operator registry MUST enforce merchant payout integrity using a separation-of-duties model. Merchant payout identifiers (e.g., IBAN) are treated as high-risk attributes and MUST NOT be trusted solely based on database state.

Operators MUST ensure that resolver responses containing payee account details are cryptographically attested using Operator signing keys. Resolver signatures MUST only be produced for merchant records that have been explicitly approved under Operator controls independent of the registry data store.

This design ensures that compromise of registry storage or resolver infrastructure cannot silently redirect payments; at worst, such compromise results in denial of service. Payer applications MUST verify resolver response signatures using Operator public keys distributed via the Operator Directory.

6. Identifiers

6.1 Operator ID (OPID)

OPID MUST be 3 alphanumeric characters (A-Z, 0-9). OPIDs are issued under Governance Authority accreditation.

6.2 Payload issuer ID (PI)

PI MUST be 3 alphanumeric characters (A-Z, 0-9). Operators manage PI assignment for payload issuers operating under their OPID.

6.3 Merchant Identifier (MID)

MID is a proxy identifier used to resolve merchant identity and IBAN details from the Operator. MID MUST be 1-70 alphanumeric characters.

For interoperability, e-QR RECOMMENDS: MID = OPID + IssuerSpecificId (i.e., embed the OPID as the first 3 characters).

6.4 Token (tok)

Tokens MUST be 1-300 alphanumeric characters (A-Z, 0-9). Tokens used for POI C2B contexts MUST be unpredictable and MUST provide at least 128 bits of entropy.

As a practical guideline, if tokens are restricted to (A-Z, 0-9), Operators SHOULD use a minimum token length of 25 characters (base36) to reach >=128 bits of entropy. Tokens MUST be short-lived and single-use for POI contexts.

7. QR carrier profile

7.1 URL structure

QR codes MUST encode an HTTPS URL structured as:

```
https://<host>/<version>/<type>/<opid>?<query>
```

Definitions:

- **<host>**: resolver host under the Operator; MUST appear in the signed Operator Directory (lowercase DNS host name).
- **<version>**: QR-format version. The first version SHALL be '1' (segment '/1/').
- **<type>**: payment context code. This v0.1 profile supports '/m/' only.
- **<opid>**: routing identifier (3-character OPID).
- **<query>**: payload as URL query parameters as specified in Section 8.

7.2 Supported type codes in v0.1

This profile supports merchant-presented POI initiation using type '/m/'. Other type codes are reserved for future versions.

7.3 Domain trust and legitimacy checks

Payer applications MUST reject QR codes whose <host> and <opid> are not present (as an active entry) in the current signed Operator Directory.

Payer applications MUST verify that the scanned <host> is listed in the *same* Operator Directory entry as the scanned <opid> (i.e., <host> MUST be present in operators[].hosts[] for the operators[] object whose opid equals the scanned <opid>). If not, reject.

Payer applications MUST reject QR codes using non-default HTTPS ports, IP-literal hosts, userinfo components, or URL fragments. Host name comparison MUST be case-insensitive and performed on the lowercase ASCII form.

7.4 Clear-text restriction for '/m/'

For POI context ('/m/'), this profile restricts QR payloads to proxy or token modes. All payee account details (IBAN) MUST be obtained via resolution (not embedded as clear-text in the QR).

8. Payload profiles

Payload is carried as a URL query string in the QR. Parameter names are case-sensitive and SHALL be lowercase in this profile. Unless stated otherwise, parameter values MUST be URL-safe and percent-encoded where required.

8.0 Parsing and validation rules

- Payer applications MUST parse the QR URL using a standards-compliant URL parser.
- Query parameter values MUST be interpreted as UTF-8 after percent-decoding. Invalid percent-encoding MUST cause rejection with error invalid_request.
- If the query contains duplicate parameter names, payer applications MUST reject the QR as invalid_request.
- For type '/m/', exactly one of **mid** or **tok** MUST be present. If both or neither are present, reject as invalid_request.

8.1 Common parameters

Param	Presence	Applies	Encoding	Meaning / Notes
pi	M	proxy+token	3 an	Payload issuer ID. Operators MUST validate that pi is registered under the OPID; otherwise error invalid_request.
instr	M	proxy+token	4 an	Payment instrument. Values: SCTI (SCT Inst).
ccy	O	proxy	3 an	Currency (ISO 4217). For SCT Inst, if present, ccy MUST be 'EUR'. If omitted, payer apps MUST assume EUR.
amt	O	proxy	1-12 n	Amount (minor units). Integer minor units (Section 8.4). If omitted, payer app MUST prompt user to enter amount.
mcc	O	proxy	4 n	Merchant Category Code. If known, Operators SHOULD provide merchant.mcc in the resolver response.
rmt	O	proxy	1-140	Remittance info (unstructured). Payload issuers SHOULD keep it short when encoded in QR to limit QR size.
ref	O	proxy	1-35 an	Structured creditor reference (e.g., ISO 11649 'RF...') for reconciliation.
purp	O	proxy	4 an	Purpose code (ISO 20022) if used.

8.2 Proxy mode payload (MID)

Proxy mode is used when the QR code contains a merchant proxy (MID) that resolves to verified merchant identity and IBAN.

Param	Presence	Encoding	Meaning / Notes
mid	M	1-70 an	Merchant Identifier (proxy). Resolved via Operator to merchant identity + IBAN.

8.3 Token mode payload (dynamic)

Token mode is used when the QR code contains a token resolving to a specific transaction request. Token resolution returns merchant details, amount, reference and expiry.

Param	Presence	Encoding	Meaning / Notes
tok	M	1-300 an	Transaction token. MUST meet entropy requirements (Section 6.4). SHOULD be single-use and short-lived. Token mode is intended for dynamic POI contexts.

8.4 Amount encoding

Amount MUST be encoded as integer minor units for the currency to avoid decimal ambiguity. Example: EUR 12.34 -> ccy=EUR&amt=1234

9. Resolver interface

The scanned QR URL identifies the Operator resolver endpoint. For security and privacy reasons, payer applications MUST NOT transmit payload parameters (including tokens) in the HTTP request URL. Instead, payer applications MUST send the payload parameters in the HTTP request body as defined below.

9.1 Request

Payer applications MUST perform an HTTPS POST to the resolver endpoint URL consisting of the scanned URL with any query string removed (i.e., the substring before the first '?' character).

Required headers:

- Accept: application/json
- Content-Type: application/json

The JSON request body MUST contain the decoded payload parameters from the QR query. For type '/m/', the body MUST include pi, instr, and exactly one of mid or tok. Optional proxy parameters (ccy, amt, rmt, ref, purp) MAY be included.

Payer applications MUST NOT follow HTTP redirects for type '/m/'. Any 3xx response MUST be treated as an error.

Rationale: full URLs are frequently recorded by application servers, reverse proxies, CDNs, WAFs, and monitoring systems. Transmitting tokens or transaction parameters in the URL increases the risk of unintended disclosure. Using a POST body reduces this risk.

Proxy QR example (carrier):

```
https://qr.example.org/1/m/ABC?pi=POS&instr=SCTI&mid=ABC000000123456&ccy=EUR&amt=1234&rmt=INV12
3
```

HTTP request:

```
POST https://qr.example.org/1/m/ABC
Accept: application/json
Content-Type: application/json

{
  "pi": "POS",
  "instr": "SCTI",
  "mid": "ABC000000123456",
  "ccy": "EUR",
  "amt": 1234,
  "rmt": "INV123"
}
```

Token QR example (carrier):

```
https://qr.example.org/1/m/ABC?pi=POS&instr=SCTI&tok=ABCD1234EFGH5678
```

HTTP request:

```
POST https://qr.example.org/1/m/ABC
Accept: application/json
Content-Type: application/json

{
  "pi": "POS",
  "instr": "SCTI",
  "tok": "ABCD1234EFGH5678"
}
```

9.2 Response (success)

On success, resolver returns HTTP 200 and JSON. The response MUST include a signature object (Section 9.4).

Minimum response data for SCT Inst prefill:

- merchant.mid
- merchant.name (display name)
- merchant.account_name (OPTIONAL; name intended to match beneficiary account for Verification-of-Payee where applicable)
- merchant.iban (payee account for SCT Inst initiation; uppercase, no spaces; IBAN checksum valid)
- transaction.ccy (EUR) and transaction.amt (minor units) *if amount is known*. For proxy mode without amt provided, transaction.amt MAY be omitted.
- transaction.rmt / transaction.ref / transaction.purp (if provided)
- transaction.expires_at (REQUIRED for token mode; RFC 3339 UTC 'Z')

Example response (signature value omitted):

```
{
  "spec": "e-qr-resolver-0.1",
  "opid": "ABC",
  "mode": "proxy",
  "status": "ok",
  "merchant": {
    "mid": "ABC000000123456",
    "name": "Example Merchant", "account_name":
    "Example Merchant OÜ", "iban":
    "EE001234567890123456",
    "mcc": "5411"
  },
  "transaction": {
    "ccy": "EUR",
    "amt": 1234,
    "rmt": "INV123",
    "ref": "RF18539007547034"
  },
  "sig": {
    "jws": "<JWS compact string>"
  }
}
```

9.3 Caching

For token mode, Operators MUST set HTTP cache controls to prevent caching (e.g., 'Cache-Control: no-store'). For proxy mode, Operators MAY allow short caching when responses are signed (e.g.,’Cache-Control: private, max-age=300').

9.4 Response integrity (mandatory for '/m/')

Operators MUST sign resolver success responses for type '/m/' using JWS (RFC 7515) with alg ES256. Payer applications MUST verify resolver signatures using Operator public keys distributed via the signed Operator Directory.

Signing input:

The JWS payload MUST be the UTF-8 bytes of the JSON Canonicalization Scheme (JCS, RFC 8785) applied to the resolver response JSON with the 'sig' member removed.

JWS requirements:

- JWS serialization: Compact.
- alg: ES256.
- kid: REQUIRED and MUST match an active key for the Operator in the Operator Directory.
- The application MUST verify that the decoded JWS payload bytes are identical to the canonicalized JSON bytes computed from the received response (with 'sig' removed). If not identical, reject.

9.5 Errors

Errors MUST use the HTTP status codes and JSON error objects below:

error	HTTP status	Notes
invalid_request	400	Malformed/unsupported parameters; unknown or unauthorised pi; missing required fields; duplicate parameters.
not_found	404	Unknown MID or token.
expired	410	Expired token (or single-use token already consumed).
rate_limited	429	Too many requests. Operators SHOULD include Retry-After.
server_error	500	Unhandled error.
unavailable	503	Temporary outage/maintenance.

Error object:

```
{
  "status": "error",
  "error": "invalid_request|not_found|expired|rate_limited|server_error|unavailable", "message":
  "...",
  "trace_id": "..."
}
```

The 'message' field is diagnostic and MUST NOT be presented to end users verbatim.

10. Operator Directory (trust anchor)

The Governance Authority SHALL publish a signed Operator Directory listing accredited Operators, their resolver hosts, and public signing keys.

Directory endpoint:

```
https://<governance-domain>/.well-known/eqr/operators.json
```

10.1 Directory signature

The Operator Directory MUST be signed by the Governance Authority using JWS Compact (RFC 7515) with alg ES256. The directory JSON MUST include a 'sig.jws' value.

The JWS payload MUST be the UTF-8 bytes of the JSON Canonicalization Scheme (JCS, RFC 8785) applied to the directory JSON with the 'sig' member removed.

Payer applications MUST verify the directory signature before using any contents. If verification fails, the directory MUST be treated as invalid.

The Governance Authority public verification key(s) used for directory signature verification MUST be distributed to payer applications via an out-of-band trusted channel (e.g., embedded trust store / application configuration) for pilots.

10.2 Freshness and expiry

The directory MUST include published_at and valid_until timestamps (RFC 3339 UTC 'Z'). Payer applications MUST refresh the directory at least every 24 hours and MUST NOT trust a directory past valid_until.

In emergency revocation scenarios, the Governance Authority MAY publish a directory with shorter validity. Payer applications SHOULD refresh more frequently when next_update is present.

10.3 Minimum directory fields

- spec_version
- published_at
- valid_until
- next_update (OPTIONAL; RFC 3339 UTC 'Z')
- operators[].opid
- operators[].status (active|suspended|revoked)
- operators[].hosts[] (lowercase DNS hosts allowed for resolver URLs)
- operators[].signing_keys[] (JWK, RFC 7517) including kid, kty, crv, x, y, alg=ES256, use=sig, and optional not_before/not_after
- sig.jws

10.4 Key status, rotation and revocation

- Operator status: payer applications MUST treat operators with status other than 'active' as untrusted.
- Key selection: resolver response JWS header 'kid' MUST match an Operator signing key in the directory.
- Key validity: if key attributes not_before and/or not_after are present, payer applications MUST enforce them.
- Rotation: Operators MAY publish multiple active keys during rotation; payer applications SHOULD accept any active key for the Operator.
- Revocation: compromised keys MUST be removed or marked invalid in the directory and the directory MUST be republished.

Example directory (signature value and key material abbreviated):

```
{
  "spec_version": "e-qr-directory-0.1",
  "published_at": "2026-01-10T00:00:00Z",
  "valid_until": "2026-01-11T00:00:00Z",
  "operators": [
    {
      "opid": "ABC",
      "status": "active",
      "hosts": ["qr.example.org"],
      "signing_keys": [
        {
          "kid": "abc-2026-01",
          "kty": "EC",
          "crv": "P-256",
          "x": "...",
          "y": "...",
          "use": "sig",
          "alg": "ES256"
        }
      ]
    }
  ],
  "sig": { "jws": "<JWS compact string>" }
}
```

11. Security and privacy

11.1 Transport security

- All resolver and directory endpoints MUST use HTTPS.
- Endpoints MUST support TLS 1.2 or TLS 1.3 and MUST NOT support TLS 1.0/1.1.
- Clients MUST validate TLS certificates.
- Operators SHOULD deploy HSTS and a modern TLS configuration.

11.2 QR and URL handling hardening

- Payer applications MUST reject URLs with userinfo, fragments, IP-literal hosts, or non-default ports.
- Payer applications MUST NOT follow redirects for '/m/' resolution.
- Payer applications MUST perform resolver calls using an application HTTP client and MUST NOT launch an external browser to resolve '/m/' URLs.
- Payer applications MUST NOT transmit payload parameters (including tok, rmt, ref) in the HTTP request URL. Payload parameters MUST be sent in the POST body as defined in Section 9.1.
- Operators SHOULD include trace_id for all errors.

11.3 Token and URL disclosure considerations

Tokens and other security-sensitive values embedded in URLs are commonly exposed via infrastructure logging (application servers, reverse proxies, WAFs, CDNs), browser history, or referrer headers. For this reason:

- Payer applications MUST treat tok as a secret and MUST NOT write tok to application logs or analytics events.
- Operators MUST treat tok as secret and SHOULD ensure access logs and monitoring do not record tok. If request bodies are logged, Operators SHOULD redact tok and other sensitive fields.
- Token mode SHOULD be short-lived and single-use. If a single-use token is presented after successful use, Operator MUST return expired (HTTP 410).

11.4 User confirmation

Payer applications MUST display resolved merchant name and amount (if known) to the user and require explicit confirmation before initiating SCT Inst.

12. Conformance requirements

12.0 Conformance profiles

This specification defines two payload profiles (proxy and token). Conformance is defined as follows:

- **Core conformance**: support for proxy mode (MID) for type '/m/'.
- **Token extension**: additional support for token mode (tok) for type '/m/'.

12.1 Operator conformance

- MUST operate a resolver endpoint for type '/m/' supporting proxy mode.
- MAY additionally support token mode. If token mode is supported, Operator MUST enforce token expiry and MUST return expired (410) for already-consumed single-use tokens.
- MUST publish hosts and signing keys through the signed Operator Directory.
- MUST maintain MID-to-IBAN integrity and merchant record status.
- MUST validate that pi is authorised under the Operator; otherwise return invalid_request.
- MUST return clear errors for invalid/expired tokens and unknown MIDs.
- MUST sign resolver success responses for '/m/' as defined in Section 9.4.
- MUST reject requests with both mid and tok present.

12.2 Payer application conformance

- MUST parse and validate URL structure and reject non-HTTPS URLs.
- MUST verify directory signature and enforce directory expiry (valid_until).
- MUST verify that the scanned host is authorised for the scanned opid in the Operator Directory and reject if not trusted.
- MUST resolve details prior to payment initiation and MUST NOT follow redirects.
- MUST send resolver requests using HTTPS POST and MUST NOT transmit payload parameters in the request URL (Section 9.1).
- MUST verify resolver response signatures as defined in Section 9.4.
- MUST display merchant + amount (if known) and require confirmation.
- If token mode is not implemented, the payer application MUST reject tok-based QRs with a user-friendly error indicating that the QR type is not supported.

13. Test vectors (non-normative)

The following examples are intended to support interoperability testing.

Valid proxy QR

QR input:

```
https://qr.example.org/1/m/ABC?pi=POS&instr=SCTI&mid=ABC000000123456&ccy=EUR&amt=1234&rmt=INV12
3
```

Expected outcome: App validates host<->opid binding in directory, POSTs body to https://qr.example.org/1/m/ABC, expects HTTP 200 with signed response; app verifies and prefills SCT Inst.

Valid token QR

QR input:

```
https://qr.example.org/1/m/ABC?pi=POS&instr=SCTI&tok=ABCD1234EFGH5678
```

Expected outcome: App validates directory, POSTs body to https://qr.example.org/1/m/ABC with tok in JSON body, expects HTTP 200 signed response with expires_at; app verifies signature and prefill.

Untrusted host

QR input:

```
https://evil.example/1/m/ABC?pi=POS&instr=SCTI&mid=ABC000000123456
```

Expected outcome: App MUST reject before network call (host not in directory).

Wrong OPID/host binding

QR input:

```
https://qr.example.org/1/m/ZZZ?pi=POS&instr=SCTI&mid=ABC000000123456
```

Expected outcome: App MUST reject (host is not authorised for OPID ZZZ).

Expired or consumed token

QR input:
```
https://qr.example.org/1/m/ABC?pi=POS&instr=SCTI&tok=EXPIREDTOKEN
```
Expected outcome: Expect HTTP 410 expired.

Unsigned response

QR input:

```
(any)
```

Expected outcome: App MUST reject any HTTP 200 response missing sig.jws for '/m/'.

Bad signature

QR input:

```
(any)
```

Expected outcome: App MUST reject if JWS verification fails or payload mismatch is detected.

Appendix A. Schemas (informative)

A.0 Resolver request body fields

```
pi: string (3 an)
instr: string (must equal 'SCTI')
mid: string (proxy mode) OR tok: string (token mode)
ccy: string (optional; if present must equal 'EUR') amt:
integer (optional; minor units)
rmt: string (optional)
ref: string (optional)
purp: string (optional)
```

A.1 Resolver success response fields

```
spec: string (must equal 'e-qr-resolver-0.1')
opid: string (3 an)
mode: string ('proxy'|'token')
status: string ('ok')
merchant.mid: string
merchant.name: string (display name)
merchant.account_name: string (optional)
merchant.iban: string (IBAN, uppercase, no spaces)
merchant.mcc: string (optional, 4 digits)
transaction.ccy: string (must equal 'EUR')
transaction.amt: integer (optional; minor units)
transaction.rmt: string (optional) transaction.ref:
string (optional) transaction.purp: string
(optional)
transaction.expires_at: string (required for token mode; RFC3339 UTC 'Z') sig.jws:
string (required; JWS compact ES256)
```

A.2 Resolver error response fields

```
status: string ('error')
error: string ('invalid_request'|'not_found'|'expired'|'rate_limited'|'server_error'|'unavailable')
message: string
trace_id: string
```

A.3 Operator Directory fields

```
spec_version: string (must equal 'e-qr-directory-0.1') published_at:
string (RFC3339 UTC 'Z')
valid_until: string (RFC3339 UTC 'Z') next_update:
```

```
string (optional; RFC3339 UTC 'Z')
operators[].opid: string (3 an)
operators[].status: string
('active'|'suspended'|'revoked')
operators[].hosts[]: string (lowercase DNS host)
operators[].signing_keys[]: object (JWK public key with kid, kty, crv, x, y, use='sig', alg='ES256')
sig.jws: string (required; JWS compact ES256)
```

14. Reference documents

- EPC024-22 v2.10 - Standardisation of QR-codes for MSCTs.
- EPC269-19 v3.0 - MSCT interoperability guidance.
- EPC193-22 v1.2 - Specification of QR-codes for mobile (instant) credit transfers in ISO format.
- EPC069-12 v3.1 - QR Code guidelines to enable data capture for initiation of an SCT.
- RFC 7515, RFC 7517, RFC 8785, RFC 3339.