AAS JSON (AASJ) Exchange Package File Format Specification

Version 2.0, https://aasj.basyx-enterprise.net

Alexander Gordt, objective partner AG

2026-02

Table of Contents

1. General

1.1 Scope

This specification defines the AAS JSON (AASJ) Exchange Package Format as an alternative to the AASX format specified in IDTA-01005. The AASJ format prioritizes simplicity, debuggability, and flexible security while maintaining full compatibility with Asset Administration Shell data models.

1.2 Normative References

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document:

- **[RFC 7515]** JSON Web Signature (JWS)
- **[RFC 7518]** JSON Web Algorithms (JWA)
- **[RFC 4648]** Base64 Data Encodings
- **[RFC 8259]** JavaScript Object Notation (JSON)
- **[FIPS 180-4]** Secure Hash Standard (SHA-256)
- **[IDTA-01001]** Asset Administration Shell Part 1
- **[ZIP]** .ZIP File Format Specification (PKWARE Inc.)

1.3 Terms and Definitions

Term	Definition
Identifiable	An AAS element with a globally unique identifier (Shell, Submodel, ConceptDescription)
Binary Content	Supplementary file referenced by an Identifiable (thumbnail, attachment)
Binary Reference	Pointer to existing Binary Content for deduplication
Content Hash	SHA-256 hash of file content for integrity verification
JWS	JSON Web Signature providing cryptographic authenticity
Package Metadata	Index file (metadata.json) containing package structure

1.4 License

2. Package Structure

2.1 Archive Format

The AASJ package SHALL use the ZIP archive format as specified in the .ZIP File Format Specification.

- The file extension for the AASJ format SHALL be `*.aasj`
- The compression method MAY be DEFLATE, STORE, or any ZIP-supported algorithm
- The archive SHALL NOT use ZIP encryption if JWS signatures are present

2.2 File Organization

The package SHALL contain files organized in a flat structure at the root level:

File Pattern	Cardinality	Description
`metadata.json`	1..1	Package metadata and index (mandatory)
`shell-{hash}`	0..*	Asset Administration Shell files
`submodel-{hash}`	0..*	Submodel files
`concept_description-{hash}`	0..*	Concept Description files
`binary-{hash}`	0..*	Binary attachment files

The package MAY contain fully signed content (with JWS), but in that case it also MUST contain the decoded content as listed above.

File Pattern	Cardinality	Description
`shell-{hash}.jws`	0..*	Asset Administration Shell files as embedded JWS
`submodel-{hash}.jws`	0..*	Submodel files as embedded JWS
`concept_description-{hash}.jws`	0..*	Concept Description files as embedded JWS
`binary-{hash}.jws`	0..*	Binary attachment files as embedded JWS

2.3 Path Generation Algorithm

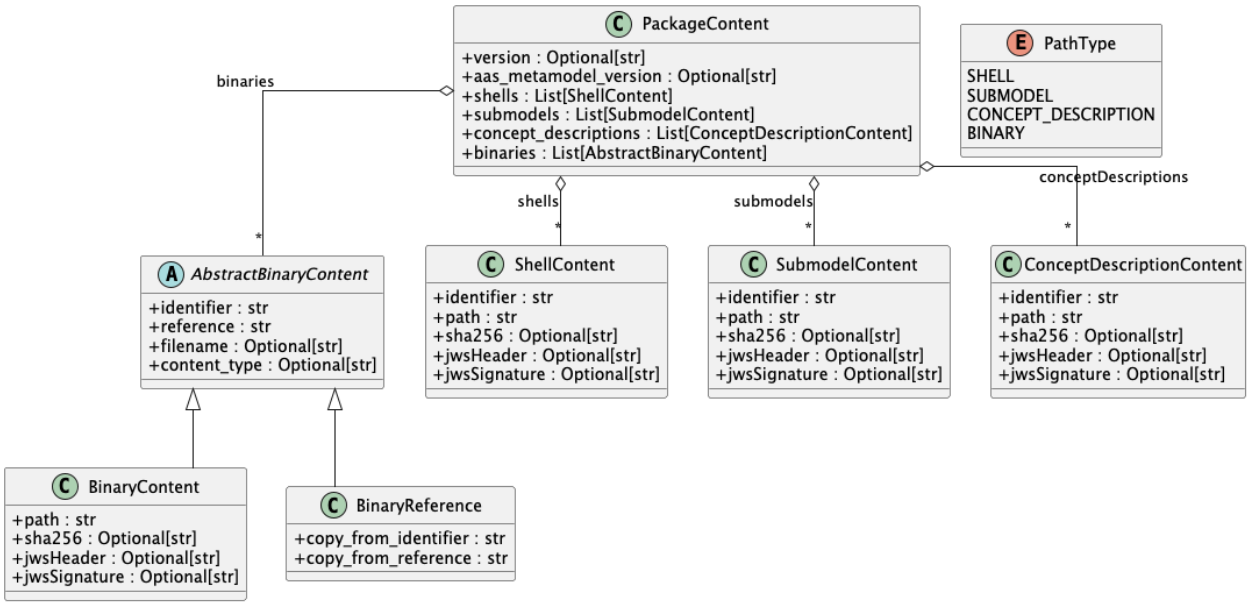
File paths SHALL be generated using the following algorithm:

```
path = type + "-" + SHA256(identifier [+ reference])
```

Where: - `type` in {`shell`, `submodel`, `concept_description`, `binary`} - `identifier` = Identifiable.id value - `reference` = idShort path or “thumbnail” (binary files only)

3. Metadata Specification

3.1 Metadata Structure



The `metadata.json` file SHALL contain a JSON object with the following top-level fields:

Field	Type	Cardinality	Description
`version`	string	0..1	Package format version (“1” or “2”)
`aasMetamodelVersion`	string	0..1	AAS metamodel version (default: “3.0”)
`shells`	array	1..1	Array of ShellContent objects
`submodels`	array	1..1	Array of SubmodelContent objects
`conceptDescriptions`	array	1..1	Array of ConceptDescriptionContent objects
`binaries`	array	1..1	Array of BinaryContent or BinaryReference objects

3.2 Content Object Specifications

3.2.1 ShellContent Object

Field	Type	Cardinality	Version	Description
`identifier`	string	1..1	1,2	Shell identifier (IRI/URI)
`path`	string	1..1	1,2	File path within archive
`sha256`	string	0..1*	2	SHA-256 hash of content
`jwsHeader`	string	0..1**	2	Base64url-encoded JWS header
`jwsSignature`	string	0..1**	2	Base64url-encoded JWS signature
`jwsAttached`	boolean	0..1	2	Flag if redundant JWS file is included, default = false

*Required if jwsHeader or jwsSignature present **Ether both or neither must be present

3.2.2 SubmodelContent Object

Field	Type	Cardinality	Version	Description
`identifier`	string	1..1	1,2	Submodel identifier (IRI/URI)
`path`	string	1..1	1,2	File path within archive
`sha256`	string	0..1*	2	SHA-256 hash of content
`jwsHeader`	string	0..1**	2	Base64url-encoded JWS header
`jwsSignature`	string	0..1**	2	Base64url-encoded JWS signature
`jwsAttached`	boolean	0..1	2	Flag if redundant JWS file is included, default = false

*Required if jwsHeader or jwsSignature present **Ether both or neither must be present

3.2.3 ConceptDescriptionContent Object

Field	Type	Cardinality	Version	Description
`identifier`	string	1..1	1,2	ConceptDescription identifier
`path`	string	1..1	1,2	File path within archive
`sha256`	string	0..1*	2	SHA-256 hash of content
`jwsHeader`	string	0..1**	2	Base64url-encoded JWS header
`jwsSignature`	string	0..1**	2	Base64url-encoded JWS signature
`jwsAttached`	boolean	0..1	2	Flag if redundant JWS file is included, default = false

*Required if jwsHeader or jwsSignature present **Ether both or neither must be present

3.2.4 BinaryContent Object

Field	Type	Cardinality	Version	Description
`identifier`	string	1..1	1,2	Parent Identifiable identifier
`reference`	string	1..1	1,2	Reference path within parent
`path`	string	1..1	1,2	File path within archive
`filename`	string	0..1	2	Original filename
`contentType`	string	0..1	2	MIME type
`sha256`	string	0..1*	2	SHA-256 hash of content
`jwsHeader`	string	0..1**	2	Base64url-encoded JWS header
`jwsSignature`	string	0..1**	2	Base64url-encoded JWS signature
`jwsAttached`	boolean	0..1	2	Flag if redundant JWS file is included, default = false

*Required if jwsHeader or jwsSignature present **Ether both or neither must be present

3.2.5 BinaryReference Object

Field	Type	Cardinality	Version	Description
`identifier`	string	1..1	2	Target Identifiable identifier
`reference`	string	1..1	2	Target reference path
`copy_from_identifier`	string	1..1	2	Source Identifiable identifier
`copy_from_reference`	string	1..1	2	Source reference path
`filename`	string	0..1	2	Original filename (inherited)
`contentType`	string	0..1	2	MIME type (inherited)

4. Content Files

4.1 Identifiable Content Files

Files containing Identifiables (Shells, Submodels, ConceptDescriptions) SHALL:

1. Be serialized as JSON according to the AAS metamodel specification
2. Use UTF-8 encoding without BOM
3. Contain exactly one complete Identifiable element
4. NOT include referenced Binary Content inline

4.2 Binary Content Files

Binary files SHALL:

1. Be stored in their original format without modification
2. Be referenced by exactly one BinaryContent entry in metadata
3. MAY be referenced by multiple BinaryReference entries

5. Security and Integrity

5.1 Security Tiers

The package format SHALL support four security tiers:

Tier	SHA256	JWS	Use Case
0	Optional	No	Legacy compatibility, trusted environments
1	Required	No	Basic integrity verification
2	Required	Required	Full cryptographic validation
3	Required	Required	Full cryptographic validation without logic

5.2 Hash Computation

When SHA256 hashes are used:

1. The hash SHALL be computed over the raw file bytes
2. The hash SHALL be stored as lowercase hexadecimal string
3. The hash SHALL NOT include any prefix (e.g., “sha256:”)

5.3 JWS Signature Requirements

When JWS signatures are used:

1. The sha256 field MUST be present and valid
2. The jwsHeader field MUST contain a valid JWS header
3. The jwsSignature field MUST contain a valid signature
4. The signature SHALL be computed according to RFC 7515
5. The complete compact representation of “header.payload.signature” MAY be stored in a separate .jws file if jwsAttached is set to true

5.4 JWS Processing

For signed content, the following process SHALL be applied:

Export (Signing): 1. Obtain JWS with embedded payload from signing service 2. Split JWS at period delimiters: `header.payload.signature` 3. Store base64url-decoded payload as file content 4. Store header in jwsHeader field 5. Store signature in jwsSignature field 6. Compute and store SHA256 of decoded payload 7. Optionally include redundant .jws file and set jwsAttached to true

Import (Validation) - jwsAttached is false: 1. Verify SHA256 hash of file content 2. Base64url-encode file content to recreate payload. *ATTENTION*: Pure byte processing must be used to avoid byte manipulation by any string / text / JSON processing. 3. Reconstruct JWS: `{jwsHeader}.{payload}.{jwsSignature}` 4. Validate signature using appropriate algorithm and key

Import (Validation) - jwsAttached is true: 1. Validate signature using appropriate algorithm and key

6. Version Compatibility

6.1 Version Identification

Version	Identification Criteria
1	`version` absent or “1”, no security fields
2	`version` = “2”, security fields optional

6.2 Forward Compatibility

Version 1 readers: - SHALL ignore unknown fields in metadata.json - SHALL process content files normally - SHALL NOT validate security fields

Version 2 readers: - SHALL process Version 1 packages - SHALL apply security validation when fields present - MAY upgrade Version 1 packages to Version 2

6.3 Migration Rules

Version 1 to Version 2: 1. Set `version` to "2" 2. Add `aasMetamodelVersion`, default is "3.0" 3. Optionally compute sha256 for content 4. Optionally add filename/contentType to binaries

7. Security Considerations

7.1 Threat Model

The package format addresses these security concerns:

Threat	Mitigation	Security Tier
Content tampering	SHA256 hash validation	1, 2
Origin spoofing	JWS signature validation	2
Replay attacks	Timestamp validation in JWS	2
Information disclosure	ZIP encryption (optional)	All
Resource exhaustion	Size limits, streaming	All

7.2 Security Recommendations

1. Implementations SHOULD validate content before processing
2. Untrusted packages SHOULD use Security Tier 2
3. Signing keys SHOULD be properly managed and rotated
4. Implementations SHOULD limit package and file sizes
5. Sensitive content SHOULD use additional encryption

8. Conformance

8.1 Conformance Classes

Class	Requirements
Producer-Basic	Generate valid Version 1 packages
Producer-Standard	Generate Version 2 packages with SHA256
Producer-Full	Generate Version 2 packages with JWS
Consumer-Basic	Read Version 1 packages
Consumer-Standard	Read Version 2, validate SHA256
Consumer-Full	Read Version 2, validate JWS

Annex A: Example Metadata

```
{
  "version": "2",
  "aasMetamodelVersion": "3.0",
  "shells": [
    {
      "identifier": "https://example.com/aas/123",
      "path": "shell-c51b17f0c6525f4e81843ccd438cc77f70df39113530fc87182027b400de1ac3",
      "sha256": "a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890",
```

```
      "jwsHeader": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9",
      "jwsSignature": "SflKxwRJSMeKKF2QT4fwpMeJf36POk",
      "jwsAttached": false
    }
  ],
  "submodels": [
    {
      "identifier": "https://example.com/sm/456",
      "path": "submodel-c1e09905a53cdcf8e2e5aa0a2335b4f9e3f3a921ca0e438d4cef8ec58c5162bc",
      "sha256": "b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890a1"
    }
  ],
  "conceptDescriptions": [],
  "binaries": [
    {
      "identifier": "https://example.com/aas/123",
      "reference": "thumbnail",
      "path": "binary-a7814558759e609ab85d4ff296f7cebd2c71aae5e1fa848a25593057bbaaf830",
      "filename": "thumbnail.png",
      "contentType": "image/png",
      "sha256": "c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2"
    },
    {
      "identifier": "https://example.com/sm/456",
      "reference": "Documents[7].DocumentVersions[0].PreviewFile",
      "path": "binary-65c4166574278e6eebf24bbb8d55aafa655f8b6a963301cfef3bccba3290d961",
      "filename": "preview.pdf",
      "contentType": "application/pdf",
      "sha256": "d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3"
    }
  ]
}
```


Annex B: Path Generation Examples

Shell

- Identifier: `https://example.com/aas/123`
- Reference: —
- Path: `shell-c51b17f0c6525f4e81843ccd438cc77f70df39113530fc87182027b400de1ac3`

Submodel

- Identifier: `https://example.com/sm/456`
- Reference: —
- Path: `submodel-c1e09905a53cdcf8e2e5aa0a2335b4f9e3f3a921ca0e438d4cef8ec58c5162bc`

Binary (Thumbnail)

- Identifier: `https://example.com/aas/123`
- Reference: `thumbnail`
- Path: `binary-a7814558759e609ab85d4ff296f7cebd2c71aae5e1fa848a25593057bbaaf830`

Binary (Submodel Attachment)

- Identifier: `https://example.com/sm/456`
- Reference: `Documents[7].DocumentVersions[0].PreviewFile`

- Path: `binary-65c4166574278e6eebf24bbb8d55aafa655f8b6a963301cfef3bccba3290d961`

Annex C: Security Examples

C.1 Tier 0 - No Verification (ShellContent)

```
{
  "identifier": "https://example.com/aas/123",
  "path": "shell-c51b17f0c6525f4e81843ccd438cc77f70df39113530fc87182027b400de1ac3"
}
```

C.2 Tier 1 - SHA256 Only (ShellContent)

```
{
  "identifier": "https://example.com/aas/123",
  "path": "shell-c51b17f0c6525f4e81843ccd438cc77f70df39113530fc87182027b400de1ac3",
  "sha256": "a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890"
}
```

C.3 Tier 2 - SHA256 + JWS (ShellContent)

```
{
  "identifier": "https://example.com/aas/123",
  "path": "shell-c51b17f0c6525f4e81843ccd438cc77f70df39113530fc87182027b400de1ac3",
  "sha256": "a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890",
  "jwsHeader": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9",
  "jwsSignature": "SflKxwRJSMeKKF2QT4fwpMeJf36POk",
  "jwsAttached": false
}
```

C.4 Tier 3 - SHA256 + JWS + Redundant JWS File (ShellContent)

```
{
  "identifier": "https://example.com/aas/123",
  "path": "shell-c51b17f0c6525f4e81843ccd438cc77f70df39113530fc87182027b400de1ac3",
  "sha256": "a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890",
  "jwsHeader": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9",
  "jwsSignature": "SflKxwRJSMeKKF2QT4fwpMeJf36POk",
  "jwsAttached": true
}
```

Implicit path for redundant JWS file: `shell-c51b17f0c6525f4e81843ccd438cc77f70df39113530fc87182027b400de1a`

C.5 BinaryContent - Thumbnail

```
{
  "identifier": "https://example.com/aas/123",
  "reference": "thumbnail",
  "path": "binary-a7814558759e609ab85d4ff296f7cebd2c71aae5e1fa848a25593057bbaaf830",
  "filename": "thumbnail.png",
  "contentType": "image/png",
  "sha256": "c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2"
}
```

C.6 BinaryContent - Submodel Attachment


```
{
  "identifier": "https://example.com/sm/456",
  "reference": "Documents[7].DocumentVersions[0].PreviewFile",
  "path": "binary-65c4166574278e6eebf24bbb8d55aafa655f8b6a963301cfef3bccba3290d961",
  "filename": "preview.pdf",
  "contentType": "application/pdf",
  "sha256": "d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3d4e5f67890a1b2c3"
}
```