Finland EMSWe RIM – Domibus Sender Integration Guide (C1 + C2 Domain)

1. Scope and Purpose

This document provides practical guidance for senders (C1 + C2 domain) on how to transmit messages via Domibus in compliance with the European Maritime Single Window environment (EMSWe) and its Reporting Interface Module (RIM) specification. It is based on the European Commission's RIM Architecture and the Domibus 5.1.9 documentation.

The guide describes how to:

- Prepare and package a formality message (ASiC-E container);
- Build and submit a valid AS4 message through Domibus;
- Configure `pmode.xml`, certificates, and security settings;
- Maintain interoperability and compliance with Member State RIM systems.

This guide applies exclusively to the sender side (C1 + C2). The receiver (C3 + C4) operations are described in the RIM Architecture document.

1.1 Supported Formalities when reporting in Finand

Here's a list of Formalities that Finland will receive:

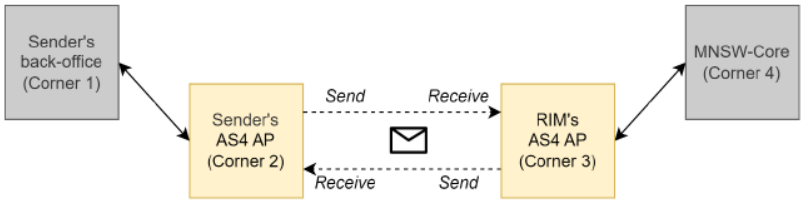
Code	Description from EMSA list
NOA	Notice of pre arrival

ATA	Notification of actual arrival
NOD	Notice of pre departure
ATD	Notification of actual departure
CGA	Cargo declaration at arrival
CGD	Cargo declaration at departure
CWA	Crew list at arrival
CWD	Crew list at departure
DUE	Fairway and port dues declaration
EFF	Crew's effects declaration
HZA	Notification of hazardous materials (dangerous and polluting goods) on board at arrival
HZD	Notification of hazardous materials (dangerous and polluting goods) on board at departure
MDD	Maritime declaration of health details
MDH	Maritime declaration of health
CRT	Ship certificates
SHP	Ship information
SID	Ship identifiers notification
SRV	Request for service
SSA	Ship-to-Ship Activity declaration
STA	Declaration of stores on board at arrival
STD	Declaration of stores on board at departure
SEC	Notification of security information
PXA	Passenger list at arrival
VID	Request for VisitID
PXD	Passenger list at departure
WAS	Advance notification for waste delivery to port reception facilities
WAR	Waste delivery receipt

2. The Four-Corner Model Overview

In the EMSWe context, data exchange follows a four-corner model:

Corner	Role	Description
C1	Sender backend	Generates formalities and prepares data payloads
C2	Sender Access Point (Domibus)	Packages payloads as AS4 messages and sends to Member State RIM
C3	Member State RIM (Domibus)	Receives, validates, authenticates, and forwards to C4
C4	Member State MNSW backend	Processes the message (formalities, certificates, etc.)



Four corner model

The sender controls the C1–C2 domain only. All communication to C3 and beyond is through secure AS4 exchange. Setting up C2, hints and tips can be found here.

3. Message Preparation and Structure

3.1 ASiC-E Container

Before transmission, the sender (C1) must create an ASiC-E container that includes:

- The XML formality document;
- A qualified eIDAS signature file (CAdES, stored in `META-INF/signatures.p7b`);
- The manifest (`manifest.xml`) and mimetype (`application/vnd.etsi.asic-e+zip`);
- Optional signed attachments (e.g. `.pdf`, `.xml`, `.jpg`, `.png`).

The container must be signed with an eIDAS Qualified or Advanced Electronic Signature (QES or AdES). Hashing algorithm: SHA-256.

3.2 Mapping to AS4 Message

Domibus (C2) wraps the ASiC-E container into an AS4 UserMessage. The required fields are defined below.

Header Elements

Field	Description	Example

`Sender`	Identifier of AS4 submitter	`sender-ap`
`Receiver`	Identifier of recipient (Member State RIM)	`rim-mnsw-fi`
`Authorization.Identifier`	Sender's EORI number	`BE1000000001`
`Authorization.Type`	Sender type (`DECL` or `DSP`)	`DECL`
`Authorization.SubDomain`	Country code of sender	`BE`
`originalSender`	Business identifier of sender backend (C1)	`urn:emswe:rim:emsa`
`finalRecipient`	Identifier of Member State MNSW backend	`urn:emswe:mnsw:fi`
`CollaborationInfo.Service`	Constant: `rim-messaging-service`	
`CollaborationInfo.Action`	Type of message (e.g., `emswe-formality-request`)	
`MessageInfo.MessageId`	Unique message identifier	UUID format
`MessageInfo.Timestamp`	Message creation timestamp	ISO 8601 format

> **EORI Number** The `Authorization.Identifier` field must contain the sender's EORI number. This is the primary business identifier used by RIM and URAM to authenticate the sender. It must match the EORI value registered in URAM.

Formality Info Metadata

Property	Description	Example

`FormalityInfo.TypeUrn`	Formality type identifier	`urn:emswe:formality:CGD`
`FormalityInfo.TypeVersion`	MIG version	`1.0`
`FormalityInfo.FileName`	Formality filename	`CGD_001.xml`
`FormalityInfo.MimeType`	Payload type	`application/vnd.etsi.asic-e+zip`
`FormalityInfo.CreationDateAndTime`	Timestamp	`2025-01-17T10:25:41Z`
`FormalityInfo.AttachmentCount`	Number of attachments	`1`
`FormalityInfo.CorrelationId`	Optional link to previous message	UUID reference

Example of AS4 message with values mapped to RIM message:

```
<eb:UserMessage mpc="http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/defaultMPC">
    <eb:MessageInfo>
        <eb:Timestamp>2025-01-17T10:25:41.000Z</eb:Timestamp>
        <eb:MessageId>7e63a507-29d5-4bdc-be01-8eed0164bdb9</eb:MessageId>
    </eb:MessageInfo>
    <eb:PartyInfo>
        <eb:From>
            <eb:PartyId type="urn:oasis:names:tc:ebcore:partyid-type:unregistered">RimMessage.Header.Sender</eb:PartyId>
            <eb:Role>http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/initiator</eb:Role>
        </eb:From>
        <eb:To>
            <eb:PartyId type="urn:oasis:names:tc:ebcore:partyid-type:unregistered">RimMessage.Header.Receiver</eb:PartyId>
            <eb:Role>http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/responder</eb:Role>
        </eb:To>
    </eb:PartyInfo>
    <eb:CollaborationInfo>
        <eb:Service type="RimMessage.Header.collaborationInfo.service.type">RimMessage.Header.collaborationInfo.service</eb:Service>

<eb:Action>RimMessage.Header.collaborationInfo.action</eb:Action>
```

```
        <eb:ConversationId>c3681e40-b522-11ee-86a1-
0a2fc46c16f1@domibus.eu</eb:ConversationId>
    </eb:CollaborationInfo>
    <eb:MessageProperties>
        <eb:Property type="RimMessage.Header.originalSender.type"
name="originalSender">RimMessage.Header.originalSender</eb:Property>
        <eb:Property type=" RimMessage.Header. finalRecipient.type"
name="finalRecipient">RimMessage.Header.finalRecipient</eb:Property>
    </eb:MessageProperties>
    <eb:PayloadInfo>
        <eb:PartInfo href="cid:message">
            <eb:PartProperties>
                <eb:Property
name="Authorization.Identifier">RimMessage.Header.Authorization.identi
fier</eb:Property>
                <eb:Property
name="Authorization.Type">RimMessage.Header.Authorization.type</eb:Pro
perty>
                <eb:Property
name="Authorization.SubDomain">RimMessage.Header.Authorization.subDoma
in</eb:Property>
                <eb:Property
name="FormalityInfo.CorrelationId">RimMessage.Header.FormalityInfo.Cor
relationId</eb:Property>
                <eb:Property
name="FormalityInfo.TypeUrn">RimMessage.Header.FormalityInfo.Formality
.formalityTypeUrn</eb:Property>
                <eb:Property
name="FormalityInfo.FileName">RimMessage.Header.FormalityInfo.Formalit
y.fileName</eb:Property>
                <eb:Property
name="FormalityInfo.CreationDateAndTime">RimMessage.Header.FormalityIn
fo.creationDateAndTime</eb:Property>
                <eb:Property
name="FormalityInfo.TypeVersion">RimMessage.Header.FormalityInfo.forma
lityVersion</eb:Property>
                <eb:Property
name="FormalityInfo.MimeType">RimMessage.Header.FormalityInfo.mimeType
</eb:Property>
                <eb:Property
name="FormalityInfo.AttachmentCount">RimMessage.Header.FormalityInfo.f
ormalityAttachments.size</eb:Property>
                <eb:Property
name="FormalityInfo.FormalityBody.Attachment[i]">
RimMessage.Header.FormalityInfo.formalityAttachments [i]</eb:Property>
            </eb:PartProperties>
        </eb:PartInfo>
    </eb:PayloadInfo>
</eb:UserMessage>
```

3.3 Validation Constraints

The Member State RIM validates all incoming AS4 messages. Key validation points:

- `PartyInfo.From.PartyId` and `To.PartyId` ≤ 255 chars;
- `Authorization.Identifier`, `Authorization.Type`, `Authorization.SubDomain` mandatory;
- `FormalityInfo.TypeUrn`, `TypeVersion`, `MimeType`, `FileName`, `CreationDateAndTime` mandatory;
- Unique `MessageId` (UUID);
- Optional `CorrelationId` ≤ 255 chars.

Failure to comply will trigger a control message (FRM) from the RIM with codes such as:

- `RIM_PLUGIN_0006` – Missing or invalid mandatory field;
- `RIM_PLUGIN_0009` – Central Authentication failure;
- `RIM_PLUGIN_0019` – Schematron validation failed.

4. Certificates, eIDAS and PKI Security

ℹ Asiakas itse joutuu hankkimaan omat varmenteensa (eIDAS claim, AS4 signing sertti ja https sertti).

Domibus requires secure key management on both ends.

Purpose	Certificate Type	Description
Message signing (C1)	eIDAS QES/AdES	Used to sign ASiC-E container
AS4 signing/encryption (C2)	Domibus keystore private key	Used for WS-Security message-level signing
Transport security	TLS server/client certificates	Used for HTTPS mutual authentication
Truststore	X.509 certificates of partner APs	Must include Member State RIM certificates

Algorithm requirements: RSA ≥ 2048 bits or ECDSA ≥ 256 bits, SHA-256 hash.

4.1 Public Key Infrastructure (PKI) Overview

The European Maritime Single Window environment (EMSWe) relies on a **Public Key Infrastructure (PKI)** for trust, authentication, confidentiality, and non-repudiation in message exchange.
According to RIM Architecture v1.1 section 8.1, PKI defines the roles, policies, and mechanisms to create, issue, manage, and revoke digital certificates used for secure electronic communication between Senders (C1/C2) and Member State RIM (C3/C4).

Each participant in the RIM four-corner model must possess appropriate **digital certificates** issued by trusted **Certificate Authorities (CA)**, preferably listed in the **EU/EEA eIDAS Trusted Lists**.

4.2 Certificate Types in the eDelivery (AS4) Context

Purpose	Certificate Type	Used By	Context / Usage
eIDAS Signature Certificate	Qualified or Advanced (QES/AdES)	C1 (Declarant/DSP system)	Used to digitally sign the ASiC-E container and registration PDF. Ensures message integrity and non-repudiation. Issued by an **EU Qualified Trust**

			Service Provider (QTSP).
AS4 Signing and Encryption Certificate	X.509 v3	C2 (Domibus Access Point)	Used by Domibus for message-level WS-Security signing and encryption in AS4 messages.
TLS/SSL Certificate	X.509 v3	C2 ↔ C3 (Domibus endpoints)	Used for HTTPS transport encryption and mutual authentication between Access Points.
CA Certificates	Trusted Root or Intermediate	All nodes	Used to verify the validity of other certificates; must be trusted by the systems involved.

These types align with RIM Architecture section 8.2.1–8.2.3, which explicitly defines TLS, signing, encryption, and CA certificate purposes.

4.3 eIDAS Certificates and QTSPs

Per RIM Architecture v1.1 §8.2.3, eIDAS certificates are issued by **Qualified Trust Service Providers (QTSPs)** accredited under EU eIDAS Regulation (EU) 910/2014.
The sender (Declarant or DSP) must request one from an approved QTSP listed in the EU Trusted List Browser.

- **Advanced Electronic Signature (AdES):** based on a qualified certificate ensuring authenticity and integrity.
- **Qualified Electronic Signature (QES):** created with a **Qualified Signature Creation Device (QSCD)** and provides the highest legal assurance level.
- **Qualified Seal:** can be used by organizations instead of individuals.

Accepted algorithms under eIDAS (RIM Architecture v1.1 §8.2.3.2):

- RSA (RSASSA-PSS): ≥ 3072 bits, SHA-256 or stronger
- ECDSA: ≥ 256 bits, SHA-256 or stronger

4.4 How Senders Obtain Certificates

Step 1: Choose QTSP

Select a **Qualified Trust Service Provider (QTSP)** listed under the EU/EEA Trusted List.
Examples (Finland and EU region):

- Posti Messaging / Posti Trust Services
- Signicat
- DigiCert Europe
- GlobalSign EU

Step 2: Request eIDAS QES/AdES Certificate

Provide:

- Organization name and EORI number
- Legal representative identification
- Public key or CSR (Certificate Signing Request)
- Optional QSCD device or HSM for private key generation

Step 3: Register Certificate with URAM

During onboarding, the **National Coordinator** registers the sender's details and certificate in URAM:

- EORI number (Authorization.Identifier)
- Type (DECL or DSP)
- Subdomain (country code)
- Certificate subject, serial, and fingerprint

RIM verifies these values automatically against URAM upon message submission. A mismatch triggers a control message (e.g. `RIM_PLUGIN_0009`).

4.5 PKI Usage per Domain

Based on the four-corner model and RIM Architecture v1.1 §8.3 diagram (p. 62–63):

Domain	Certificate Usage	Description
C1 (Sender Backend)	eIDAS QES/AdES	Signs the ASiC-E container. Private key stored in QSCD/HSM.
C2 (Sender Domibus)	AS4 Signing + TLS	Signs and encrypts the AS4 message and uses mutual TLS for transport.
C3 (RIM)	AS4 Decryption + TLS	Verifies AS4 signatures and decrypts payload. Maintains truststore of senders' public keys.
C4 (MNSW Backend)	TLS or Optional Signing	Receives the validated payload from RIM over HTTPS.

4.6 Key Management Requirements

Aspect	Recommendation

Private Key Protection	Use HSM or QSCD. Private keys must not be exportable.
Truststore Configuration	Include Member State RIM and URAM CA certificates.
Algorithm Policy	RSA ≥ 2048 bits or ECDSA ≥ 256 bits, SHA-256 minimum.
Renewal and Revocation	Renew before expiry, update fingerprints in bilateral agreements, and revoke compromised keys immediately.
Audit Trail	Keep versioned records of certificate changes and validation results.

4.7 Common Issues and Error Codes

Error	Description	RIM Error Code	Resolution
Missing or invalid certificate	Sender's certificate not registered in URAM	`RIM_PLUGIN_0009`	Re-register the correct certificate in URAM.
Expired certificate	Certificate validity date passed	`RIM_PLUGIN_0009`	Renew and update PMode and URAM record.
Mismatch between EORI and certificate	EORI/certificate pair not valid	`RIM_PLUGIN_0013`	Correct Authorization.Identifier or reissue certificate.
Invalid TLS configuration	Non-HTTPS endpoint	`RIM_PLUGIN_0011`	Use HTTPS and trusted CA chain.

5. Correlation and Control Messages

If a message fails RIM validations, a control message (FRM) is generated and sent back to the sender. The FRM includes:

- `FormalityInfo.CorrelationId` – referencing the failed message;
- Error code and text description;
- Encapsulated as ZIP/AS4 message.

Example structure:

```
<ValidationError>
  <errorCode>RIM_PLUGIN_0009</errorCode>
  <error>Central Authentication validation failed: Certificate not registered</error>
</ValidationError>
```

6. Authentication and Registration

When registering organisation to URAM, organisation must also think what is the “type” of organisation, there are two types DECL and DSP.

6.1 Relationship Between DECL and DSP

- A **Declarant (DECL)** can submit messages directly via its own integration or through a **Data Service Provider (DSP)**.
- If a DSP submits on behalf of multiple declarants, it must:
  - Be **registered in URAM** as a data service provider;
  - Include the correct **Authorization.Identifier (EORI)** and **SubDomain (country)** in every AS4 message, so that RIM can authenticate the sender properly.

6.2 Registration

A sender (declarant or data service provider) must be registered in URAM (Central Authentication Service) before transmission.

- EORI number;
- Contact details;
- Type of sender;
- Subdomain (country code);
- eIDAS certificate (when requested using PDF).

6.3 Process

1. National coordinator creates sender entry in URAM based on details provided by sender.
2. Coordinator sends registration PDF for eIDAS signature to sender.
3. Declarant signs with qualified eIDAS certificate and returns it.
4. Coordinator verifies signature and activates URAM registration.

The RIM (C3) validates every incoming message by querying URAM (Central Authentication Service).

6.4 Use of EORI Number

Each sender registered in URAM (Central Authentication Service) must have a unique **EORI number** (Economic Operators Registration and Identification).
The EORI is the **primary business identifier** in EMSWe and is used to authenticate and trace all RIM messages.

Rules

- The EORI is placed in the AS4 message under `Authorization.Identifier`.
- It must exactly match the EORI value registered in URAM.
- For DSPs (Data Service Providers), the EORI identifies the provider; URAM maintains delegations linking that DSP to its Declarants (DECL).
- The combination **EORI + certificate + SubDomain** must be unique and active in URAM.

- Any mismatch between the EORI and the certificate causes authentication failure and a control message (`RIM_PLUGIN_0009` or `RIM_PLUGIN_0013`).

7. Sender–Member State Agreement and Configuration Updates

Before production exchange, the sender and the Member State must establish a bilateral AS4 agreement defining:

Area	Description
Access Endpoints	URLs and ports for AS4 message exchange
Party Identifiers	Sender and Receiver `PartyId`s as defined in PMode
Certificates	Exchange of public keys (signing, TLS) via secure channel
PMode File	Technical configuration: service/action, security, reliability, and bindings
Security Policy	WS-Security profile, e.g. `eDeliveryAS4Policy_BST.xml`
Contact Points	Support and operational contacts for both parties
Versioning	Agreement version and effective date

When access endpoints or public keys change:

1. Notify counterpart in advance (30 days recommended);
2. Exchange and verify new certificates;
3. Update and re-upload the PMode file;
4. Test connection in staging environment;
5. Activate change in production;
6. Record the change (agreement version, date, fingerprints).

Each update must be documented with:

- New and old certificate fingerprints;
- Updated endpoint URLs;
- Change description, validation evidence, and approval.

8. Documentation and Compliance

All configuration and registration records must be version-controlled. The following artifacts are recommended:

- AS4 Agreement Sheet (technical configuration summary);

- Change Log Register (history of endpoint and certificate changes);
- Test Evidence (connectivity and validation reports);
- Registration Archive (signed registration PDFs and URAM confirmations).

These measures ensure compliance with:

- Regulation (EU) 2019/1239, recital (12);
- Commission Implementing Regulation (EU) 2023/204, Article 2;
- Commission Implementing Regulation (EU) 2023/2790, Annex II.