ETSI TS 119 524-2 V1.2.1 (2023-12)



Electronic Signatures and Infrastructures (ESI); Testing Conformance and Interoperability of Electronic Registered Delivery Services; Part 2: Test suites for interoperability testing of Electronic Registered Delivery Service Providers 2

Reference RTS/ESI-0019524-2v121

Keywords

electronic registered delivery, interoperability, testing

ETSI

650 Route des Lucioles F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B Association à but non lucratif enregistrée à la Sous-Préfecture de Grasse (06) N° w061004871

Important notice

The present document can be downloaded from: <u>https://www.etsi.org/standards-search</u>

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the prevailing version of an ETSI deliverable is the one made publicly available in PDF format at www.etsi.org/deliver.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at <u>https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx</u>

If you find errors in the present document, please send your comment to one of the following services: <u>https://portal.etsi.org/People/CommiteeSupportStaff.aspx</u>

If you find a security vulnerability in the present document, please report it through our Coordinated Vulnerability Disclosure Program: https://www.etsi.org/standards/coordinated-vulnerability-disclosure

Notice of disclaimer & limitation of liability

The information provided in the present deliverable is directed solely to professionals who have the appropriate degree of experience to understand and interpret its content in accordance with generally accepted engineering or other professional standard and applicable regulations.

No recommendation as to products and services or vendors is made or should be implied.

No representation or warranty is made that this deliverable is technically accurate or sufficient or conforms to any law and/or governmental rule and/or regulation and further, no representation or warranty is made of merchantability or fitness for any particular purpose or against infringement of intellectual property rights.

In no event shall ETSI be held liable for loss of profits or any other incidental or consequential damages.

Any software contained in this deliverable is provided "AS IS" with no warranties, express or implied, including but not limited to, the warranties of merchantability, fitness for a particular purpose and non-infringement of intellectual property rights and ETSI shall not be held liable in any event for any damages whatsoever (including, without limitation, damages for loss of profits, business interruption, loss of information, or any other pecuniary loss) arising out of or related to the use of or inability to use the software.

Copyright Notification

No part may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm except as authorized by written permission of ETSI.

The content of the PDF version shall not be modified without the written authorization of ETSI.

The copyright and the foregoing restriction extend to reproduction in all media.

© ETSI 2023. All rights reserved.

Contents

Intell	ntellectual Property Rights					
Forev	² oreword5					
Moda	al verbs terminology	5				
1	Scope	6				
2	References	6				
2.1 2.2	Normative references	6 7				
3	Definition of terms, symbols and abbreviations	7				
3.1	Terms	/ / 7				
3.3	Abbreviations	7				
4	Technical approach	8				
4.1	Components of test cases and their identifiers					
4.2	Adding new test cases to the test suite	9				
5	Scenarios	10				
5.1	Introduction					
5.2	Abbreviations used in tables defining scenarios					
5.3	Black-box model scenarios.					
532	Introduction.	12				
533	Scenarios with potification for acceptance					
5.4	Scenarios for 4-corner model	30				
5.4.1	Introduction					
5.4.2	Scenarios where RERDS does not use notification for acceptance					
5.4.3	Scenarios where RERDS uses notification for acceptance	44				
5.5	Scenarios for extended model					
5.5.1	Introduction.					
5.5.2 5.5.3	Scenarios where RERDS does not use notification for acceptance	57				
6	EDD Massages instances	92				
0 6 1	Introduction and technical approach					
6.2	Combinations of fields for headers in FRD envelopes	83 83				
6.2.1	Introduction					
6.2.2	Combinations of AS4 metadata profiled in ETSI EN 319 522-4					
6.2.3	Combinations of components of relay metadata					
6.2.4	Combinations of AS4 metadata profiled and relay metadata					
6.3	Instances of ERD payload					
6.4	Instances of ERDS receipts					
6.5 7	Instances of ERD dispatch					
/	Other named sets					
7.1 7.2	Named sets of additional requirements					
7.2	Named sets of additional requirements					
8	Test cases definition	90				
8.1	Introduction					
8.1.1	General					
8.1.2	Notation for black box model scenarios					
8.1.3	Notation for 4 corner and extended models scenarios					
0.1.4 8.2	Reasons for black has model					
8.3	Test cases for 4-cornel and extended models					
8.3.1	General					

Annex B	(informative):	Change History	96
Annex A	(informative):	Bibliography	95
8.3.7	Rules for param		
8.3.6	Rules for param	94	
8.3.5	Rules for param	netrizing entities at receiving side	
8.3.4	Rules for param	netrizing ERDS receipts	
8.3.3	Rules for param	netrizing ERD payloads	
8.3.2	Rules for param		

4

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: "Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards", which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECTTM, **PLUGTESTSTM**, **UMTSTM** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPPTM** and **LTETM** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2MTM** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**[®] and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Electronic Signatures and Infrastructures (ESI).

The present document is part 2 of a multi-part deliverable. Full details of the entire series can be found in part 1 [i.1].

Modal verbs terminology

In the present document "shall", "shall not", "should", "should not", "may", "need not", "will", "will not", "can" and "cannot" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"must" and "must not" are NOT allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document defines:

- A test suite for supporting interoperability tests within the field of Electronic Registered Delivery Services (ERD services or ERDS hereinafter) as specified in ETSI EN 319 522 parts 1 [1], 2 [2], 3 [3], 4 [4], [5] and [6]. The test suite defines test cases for the following environments:
 - Environments that correspond to the basic model as defined in ETSI EN 319 522-1 [1] where sender and all the entities at receiving side are subscribed to the same ERDS.
 - Environments that correspond to the 4-corner model as defined in ETSI EN 319 522-1 [1] where sender is subscribed to one ERDS and the entities at receiving side are subscribed to another one, and no intermediate ERDS is required for relaying ERD messages between them.
 - Environments that correspond to the extended model as defined in ETSI EN 319 522-1 [1] where sender is subscribed to one ERDS and the entities at receiving side are subscribed to another one, and intermediate ERDSs are required for relaying ERD messages between them.
- 2) A mechanism for documenting new test cases and expanding the aforementioned test suite.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference/.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

[1]	ETSI EN 319 522-1: "Electronic Signatures and Infrastructures (ESI); Electronic Registered Delivery Services; Part 1: Framework and Architecture".
[2]	ETSI EN 319 522-2: "Electronic Signatures and Infrastructures (ESI); Electronic Registered Delivery Services; Part 2: Semantic contents".
[3]	ETSI EN 319 522-3: "Electronic Signatures and Infrastructures (ESI); Electronic Registered Delivery Services; Part 3: Formats".
[4]	ETSI EN 319 522-4-1: "Electronic Signatures and Infrastructures (ESI); Electronic Registered Delivery Services; Part 4: Bindings; Sub-part 1: Message delivery bindings".
[5]	ETSI EN 319 522-4-2: "Electronic Signatures and Infrastructures (ESI); Electronic Registered Delivery Services; Part 4: Bindings; Sub-part 2: Evidence and identification bindings".
[6]	ETSI EN 319 522-4-3: "Electronic Signatures and Infrastructures (ESI); Electronic Registered Delivery Services; Part 4: Bindings; Sub-part 3: Capability/requirements bindings".
[7]	ETSI EN 319 532-3: "Electronic Signatures and Infrastructures (ESI); Registered Electronic Mail (REM) Services; Part 3: Formats".
[8]	OASIS Standard (January 2013): "AS4 Profile of ebMS 3.0 Version 1.0".

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

- [i.1] ETSI TS 119 524-1: "Electronic Signatures and Infrastructures (ESI); Testing Conformance and Interoperability of Electronic Registered Delivery Services; Part 1: Testing conformance".
- [i.2] OASIS Standard (October 2007): "<u>ebXML Messaging Services Version 3.0: Part 1, Core</u> <u>Features</u>".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI EN 319 522-1 [1] apply.

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

ACCeptance REJection EXPiry
Applicability Statement 4
CONSignment ACCeptance
CONSignment NOTification
CONSignment NOTification FAILure
CONSignment REJection
CONTent ACcess TRACKing
CONTent CONSignment
CONTent CONSignment FAILure
CONTent HANDover
CONTent HANDover FAILure
ebXML Messaging Services
electronic business using eXtensible Markup Language
Electronic Registered Delivery
Electronic Registered Delivery Service
Evidence - SET
Intermediate Electronic Registered Delivery Service
NOTification ACcess TRACKing
NOTification DELivered
NOTification For ACCceptance
NOTification For ACCceptance FAIL
Organization for the Advancement of Structured Information Standards
RECeived From Non ERDS
RELay ACCeptance
RELay FAILure

REL REJ	RELav REJection
REL T NERDS	RELay To Non ERDS
REL_T_NERDS_FAIL	RELay To Non ERDS FAILure
REM	Registered Electronic Mail
REMS	Registered Electronic Mail Service
RERDS	Recipient's Electronic Registered Delivery Service
SCN_ID	Scenario IDentifier
SERDS	Sender's Electronic Registered Delivery Service
SUB_ACC	SUBmission ACCeptance
SUB_REJ	SUBmission REJection
URI	Universal Resource Identifier
XML	eXtensible Mark-up Language

4 Technical approach

4.1 Components of test cases and their identifiers

As it has been mentioned before the present document defines:

- 1) A test suite for supporting interoperability tests within the field of Electronic Registered Delivery (ERD hereinafter) as specified in ETSI EN 319 522 parts 1 [1], 2 [2], 3 [3], 4 [4], [5] and [6].
- 2) A mechanism for documenting new test cases and expanding the aforementioned test suite.

The present document follows a layered approach for building the definition of the test cases in the test suite, which can be summarized as follows:

 Clause 5 defines a number of parameterized scenarios. A scenario consists of a number of entities, namely: sender, one or more ERDSs, and the entities at receiving side (one or more recipients and/or one or more recipients' delegates), which exchange different ERD messages with time. Each scenario corresponds to one of the three models presented in ETSI EN 319 522-1 [1]. This clause presents a template for defining one scenario, in a way that resembles to some templates used for defining use cases scenarios in software engineering.

This template:

- Includes the enumeration of the original message and all the ERD messages exchanged by the participating entities. This list of exchanged ERD messages is one of the parameters of the scenario.
- Also includes a list of ERDS evidence sets, which, in the scenario, are incorporated in some ERD messages.

One scenario may be used for defining several test cases depending on:

- The specific components of each exchanged ERD message (suppressing or adding an optional metadata component, or changing the value of a certain metadata component results in a different ERD message and consequently a different test case).
- The entities at receiving part (for instance, changing one recipient by one recipient's delegate, or two recipients and one recipient's delegate results in a different the test case).
- A named set of additional requirements (for instance details of the original message, like whether it contains or not attachments, is signed, is encrypted, etc.).

This means that one test case corresponds to one scenario where all the exchanged ERD messages have been completely defined in terms of their components, all the participating entities have been established, and all the additional requirements have also been defined. Taking the functional notation this can be expressed as follows:

TestCase#i = Scenario_id(,<Receiving side identifier>, <ERD message 1 details>, <ERD message 2 details>, ..., <ERD message N details>, <additional requirements set identifier)

Where:

- <Receiving side identifier> is the identifier assigned to a certain set of entities at receiving side;

9

- <message identifier I> is the identifier of a specific instantiation of the aforementioned message, defined in clauses 6.3, 6.4, and 6.5 These clauses define specific instantiations of ERD payloads, ERD receipts and ERD dispatches respectively.
- <additional requirements set identifier> is the identifier of a named set of additional requirements. Clause 7.2 defines a number of these named sets.
- 2) Clauses 6.3, 6.4, and 6.5 define specific instantiations of ERD payloads, ERD receipts and ERD dispatches respectively. Each type of ERD message is composed by several components, with their metadata components and payloads as specified in ETSI EN 319 522-4-1 [4] and ETSI EN 319 522-4-2 [5]. The present document defines a number of combinations of metadata components in clauses 6.2.2 and 6.2.3, and assigns to each one a unique identifier. This allows to use again the functional notation, and define one instantiation of a certain type of ERD message as follows:

ERD message instance = Sequence(Metadata(<AS4 profiled metadata combination details>, <payload for ERDS relay metadata combination details>) <payload for User Content>* <payload for ERDS Evidence>*)

Where '*' stands for 0 or more occurrences of the payload.

- NOTE: The payloads for user content and for ERDS evidence can be present at certain types of ERD messages but are forbidden in other types.
- 3) Clauses 6.2.2 and 6.2.3 define named combinations of metadata components defined in OASIS: "AS4 Profile of ebMS 3.0 Version 1.0" [8] and profiled in ETSI EN 319 522-4-1 [4] and ETSI EN 319 522-4-2 [5], and the relay metadata components defined in ETSI EN 319 522-3 [3] respectively. Each clause define different instances of the aforementioned components and assigns them unique identifiers that are used for defining specific instances of the different ERD messages as shown above. Once this level is reached, the specific test case is fully defined as: a scenario where fully defined, ERD messages are exchanged between a specific set participating entities, and where a specific set of additional requirements are imposed.

4.2 Adding new test cases to the test suite

The strategy followed for building the definitions of the test cases makes it easy to expand the test suite by incorporation of new test cases.

For defining a new test case the following steps are required:

- 1) Identify the **set of receiving entities**. If none of the predefined set of entities at the receiving side is the one required, assign a name to this set (**<RECEIVING_ENTITIES >**) and incorporate it to the repertoire of named sets as specified in clause 7.3). The sender is always present by default.
- 2) Define the ERDSs that will participate in the test case.
- 3) If the set of participating ERDSs is not equal to none of the scenarios already identified in the present document, the new scenario will require to be defined in a new template.
- 4) Identify the **sequence of actions** performed by each actor and their order of occurrence and assign a new unique identifier (**<SCN_ID** >) to the scenario.
- 5) Identify **all the ERD messages** generated by the actors as they go through the sequence of actions. For each message:
 - a) Identify its ebMS payloads, e.g. the parts of the user content or XML document with relay meta-data.
 - b) Check if the combinations of metadata components have already been defined in the present document. If not, add the required combination of metadata components to the repertoire of named combinations to the corresponding clause (clause 6.2.2 or 6.2.3).
 - c) List all the ERD messages exchanged as parameters of the scenario.

- d) Identify the ERDS evidence format and the set of ERDS evidence for each ERD message including them and add the names of the ERDS evidence sets to the Var section of the template.
- 6) Identify and define any other additional requirement for completely define the test case. If the set of requirements is different than all the sets already define, assign a name to it (**ADD_REQ_COMB**>) and add it to the repertoire of named sets of additional requirements in Table 12 (clause 7.2).

5 Scenarios

5.1 Introduction

The present clause defines a number of selected scenarios that will be used in clause 8.

Clause 5.3 defines scenarios where sender and recipient(s) are subscribed to the same ERDS (black-box model described in ETSI EN 319 522-1 [1]).

Clause 5.4 defines scenarios where the sender and the recipient(s) are subscribed to different ERDSs and there are not intermediate ERDSs between them (4-corner model described in ETSI EN 319 522-1 [1]).

Clause 5.5 defines scenarios where sender is subscribed to a ERDS and the recipient(s) is(are) not subscribed to the same ERDS and there are one or more intermediate ERDSs (extended model described in ETSI EN 319 522-1 [1]).

Figure 1 of clause 4 of ETSI EN 319 522-2 [2] shows three structures being exchanged between ERD-UAs and ERDSs, namely:

- 1) The structure {submission metadata, user content}, which receives the name of original message in Table 1 of clause 4 of ETSI EN 319 522-2 [2].
- 2) The structure {ERDS handover metadata, ERDS evidence} for allowing access to ERDS evidences to users.
- 3) The structure {ERDS handover metadata, user content, ERDS evidence} for allowing the R-ERDS the submission of the user content (and optionally ERDS evidences) to the recipient.

Because of that the following decisions have been adopted for building the scenarios:

- Neither S-ERDS nor R-ERDS will submit {ERDS handover metadata, ERDS evidence} structures to their users, except when the ERDS evidence is an evidence of some kind of relevant rejection by the ERDS (see the first scenario, for instance). Identical scenarios including the submission of such structures can be easily defined and used in interoperability test events.
- 2) The scenarios will show the R-ERDS submitting {ERDS handover metadata, user content, ERDS evidence} or {ERDS handover metadata, user content} structures to the receiving side.
- 3) The acronym hndvMet is used for ERDS handover metadata.

Table 1 shows the template for defining one scenario.

Scenario id: <scn_id></scn_id>				Purpose	
Par	Parameter: <erds_receipt>_WITH_XML_SUB_REJ Var SET_EV#1 = {,}</erds_receipt>				
<pa< td=""><td>arameter 1 that helps to fully the scenario.</td><td>Their number</td><td>Named sets of ERDS e</td><td>vidence used in</td><td></td></pa<>	arameter 1 that helps to fully the scenario.	Their number	Named sets of ERDS e	vidence used in	
dep	pends on the specific scenario>		the definition of the sce	nario.	
Par	ameter: < Parameter 2>		Var SET_EV#2 = {	}	
Par	ameter: < Parameter N>		Var SET_EV#N = {	.}	
	Sec	quence of actions			
<si< td=""><td>EQUENCE OF ACTIONS. THERE IS ONE</td><td>COLUMN PER PAI</td><td>RTICIPATING ACTOR></td><td></td><td></td></si<>	EQUENCE OF ACTIONS. THERE IS ONE	COLUMN PER PAI	RTICIPATING ACTOR>		
#	Sender		ERDS	Receiving side	
The	e sequence is composed of a number of nu	merated steps, whe	n the actors perform cer	tain actions as	
Sor	me frequent actions: send original message	e accent submission	n reject submission con	sian aenerate	
one	ERDS evidence, generate one ERD mess	aae. etc.		olgn, gonorato	
1	Sender sends original message				
2		Rejects submissior	n. Generates		
		XML_SUB_REJ EF	RDS evidence		
3	3 Generates				
<erds_receipt>_WITH_XML_SUB_REJ</erds_receipt>					
4	4 Sends				
		<erds_receipt>_\</erds_receipt>	NITH_XML_SUB_REJ		
5	Receives				
	<erds_receipt>_WITH_XML_SUB_RE</erds_receipt>				
	J				

Table 1: Template for the tabular definition of one scenario

Each scenario is assigned a unique identifier <SCN_ID>. The reasons why the scenario has been defined are shown in column "Purpose".

The definition of each scenario requires that parties exchange a number of ERD messages, which appear listed as parameters in the rows immediately below the headers row. Its number depends on the specific scenario.

Below the list of parameters, the table shows a sequence of actions performed by different involved entities, which results in that a set of ERD messages is generated and exchanged.

The definition of each scenario also can use a number of named ERDS evidence sets, which are listed in cells started with Var. Each ERDS evidence set is given a name EV_SET#<i>, being <i> a non-negative integer number.

The involved entities are sender (or sender's delegate, the scenario definition does not make any distinction between them), one or more ERDSs, and the entities at the receiving side (for the same scenario there may be different sets of entities, for instance one recipient, one recipient's delegate, one or more recipients, or one or more recipients and one or more recipients' delegates).

Each step is assigned a positive integer number. The actions performed include: submission of messages, generation of ERD messages, generation of ERDs evidence, acceptance of ERD messages, rejection of ERD messages, consignment of ERD messages, retrieval of ERD messages by entities at receiving side, failures, etc.

5.2 Abbreviations used in tables defining scenarios

This clause shows some abbreviations (which have already been anticipated in clause 3.3) used in the tables that define the scenarios.

Table 2 shows the abbreviations used for the different ERDS evidence.

ERDS Evidence name	ERDS Evidence abbreviation
SubmissionAcceptance	SUB_ACC
SubmissionRejection	SUB_REJ
RelayAcceptance	REL_ACC
RelayRejection	REL_REJ
RelayFailure	REL_FAIL
NotificationForAcceptance	NOT_F_ACC
NotificationForAcceptanceFailure	NOT_F_ACC_FAIL
ConsignmentAcceptance	CONS_ACC
ConsignmentRejection	CONS_REJ
AcceptanceRejectionExpiry	ACC_REJ_EXP
ContentConsignment	CONT_CONS
ContentConsignmentFailure	CONT_CONS_FAIL
ConsignmentNotification	CONS_NOT
ConsignmentNotificationFailure	CONS_NOT_FAIL
ContentHandover	CONT_HAND
ContentHandoverFailure	CONT_HAND_FAIL
RelayToNonERDS	REL_T_NERDS
RelayToNonERDSFailure	REL_T_NERDS_FAIL
ReceivedFromNonERDS	REC_F_NERDS
NotificationDelivered	NOT_DEL
NotificationAccessTracking	NOT_AC_TRACK
ContentAccessTracking	CONT_AC_TRACK

Table 2: ERDS evidence abbreviations

12

ETSI EN 319 522-1 [1] specify a XML format for ERDS evidence, but also allows that ERDS Evidences are signed PDF documents. The notation defined in the present document makes it clear that all the test cases are defined for XML ERDS Evidence using the **XML** prefix for the ERDS evidence abbreviations.

- EXAMPLE: The abbreviation for the XML SubmissionAcceptance ERDS evidence will be **XML_SUB_ACC**.
- NOTE: In case some format for PDF ERDS Evidence is defined and ERDS providers need to test interoperability with them, it is always possible to replace the test cases defined in the present document by identical test cases where PDF ERDS Evidences are generated and exchanged instead XML ERDS Evidences.

The tables defining the Scenarios use the following abbreviations for the different participating ERDSs:

- **SERDS** stands for the ERDS serving the sender, in the scenarios where it is different than the ERDS serving the entities at receiving side.
- **RERDS** stands for the ERDS serving the entities at receiving side, in the scenarios where it is different than the ERDS serving the sender.
- **IERDS** stands for a ERDS that does not directly serves neither to the sender nor to the recipient(s)/recipient's delegate, but instead is an intermediate ERDS that relies ERD messages from SERDS to RERDS and from RERDS to SERDS.

5.3 Black-box model scenarios

5.3.1 Introduction

The present clause defines scenarios where the sender and the entities at the receiving side are subscribed to the same ERDS and consequently the user content is not relayed between different ERDSs.

Clause 5.3.2 defines scenarios where the ERDS operates in Store and Forward style.

Clause 5.3.3 defines scenarios where the ERDS operates in Store and Notify style.

5.3.2 Scenarios without notification for acceptance

Table 3 defines a number of scenarios for the case where sender and the entities at the receiving side are subscribed to the same ERDS and the ERDS does not send notification for acceptance to the entities at the receiving side.

13

Table 3: Scenarios for intra-ERDS without notifications for acceptance (1/10)

		Purpose			
Para	imeter: {hndvMet,XML_SUB_REJ}				
		I ne simplest negative scenario:			
#	Sender	Where the ERDS rejects the original message submitted by			
1	Sender sends original message				the sender and sends back a
2		Rejects submission. Ge ERDS evidence	nerates XML_SUB_REJ		structure {ERDS handover metadata, XML_SUB_REJ}
3		Sends {hndvMet, XML_ sender	SUB_REJ} structure to the		ERDS evidence.
4	Receives {hndvMet, XML_SUB_REJ} structure				

Table 3a: Scenarios for intra-ERDS without notifications for acceptance (2/10)

		Purpose		
Para	ameter: {hndvMet, user content}			
Para	ameter: XML_SUB_ACC			The simplest positive scenario:
Para	ameter: XML_CONT_CONS]
		Sequence of actions		Where the ERDS accepts the
#	Sender	ERDS	Receiving side	original message, then it
1	Sender sends original message			structure {Including bandover
2		Accepts submission		metadata and user content} to N
3		Generates and stores XML_SUB_ACC ERDS evidence		entities on the receiving side.
4		Generates {hndvMet, user content} structure		Lastly, ERDS generates and
5		Consigns {hndvMet, user content} structure to the N		stores the ContentConsignment
		entities at the receiving side		ERDS evidence.
6			{hndvMet, user content} structure	
			correctly consigned to the N entities at	
			the receiving side	
7		Generates and stores XML_CONT_CONS ERDS evidence		

Table 3b: Scenarios for intra-ERDS without notifications for acceptance (3/10)

		Scenario id: ERDS_BB_NO_NOT_F_ACC#3		Purpose
Para	ameter: {hndvMet, user content}			
Para	ameter: XML_SUB_ACC			The positive scenario:
Para	ameter: XML_CONT_CONS			
Para	ameter: XML_CONT_HAND			Where the ERDS accepts the
		Sequence of actions		original message, then it
#	Sender	ERDS	Receiving side	generates and consigns the
1	Sender sends original message			metadata and user contents to N
2		Accepts submission.		entities on the receiving side. In
3		Generates and stores XML_SUB_ACC ERDS evidence		the next step, ERDS generates
4		Generates {hndvMet, user content} structure		evidence. After the successful
5		Consigns {hndvMet, user content} structure to the N entities at the receiving side		retrieval of the ERD payload on the receiving side, ERDS
6		Generates and stores XML_CONT_CONS ERDS evidence	{hndvMet, user content} structure correctly consigned to N entities at receiving side	generates and stores ContentHandover ERDS evidence for the N handovers.
7			All the entities retrieve the ERD dispatch	
8		Generates and stores XML_CONT_HAND ERDS evidence for the N handover and stores	N handovers of {hndvMet, user content} structure succeed	

Table 3c: Scenarios for intra-ERDS without notifications for acceptance (4/10)

	Scenario ic	Purpose			
Para	meter: {hndvMet, user content}				
Para	meter: XML_SUB_ACC				A negative scenario of
Para	meter: XML_CONT_HAND				ERDS_BB_NO_NOT_F_ACC#3
Para	meter: XML_CONT_HAND_FAIL				
		Sequence o	factions		Where one of the entities on the
#	Sender		ERDS	Receiving side	receiving side fails to retrieve the
1	Sender sends original message				and stores two evidences:
2		Accepts sub XML_SUB_/	mission. Generates and stores ACC ERDS evidence		ContentHandover ERDS evidence
3		Generates {	nndvMet, user		ContentHandoverFailure ERDS
		content}strue	cture		evidence for one entity.
4		Consigns {h	ndvMet, user content}		
		structure to t side	the N entities at the receiving		
5		Generates a	nd stores XML_CONT_CONS	{hndvMet, user content} structure	
		ERDS evide	nce	successfully consigned to N entities	
<u> </u>				at the receiving side	-
6				Entitles try to retrieve the ERD	
7		0		dispatch, but one fails	-
1		Generates o	N 1 antitian and and	N 4 handevers of (had) Met weer	
		evidence for	IN-1 entities and one	IN-1 nandovers of (nndviviet, user	
		XML_CONT	_HAND_FAIL ERDS evidence	content} structure succeed. One fails	
1		for one entity	y and stores them		

Table 3d: Scenarios for intra-ERDS without notifications for acceptance (5/10)

		Purpose		
Para	ameter: {hndvMet, user content, XML_S	A positive scenario of		
Para	ameter: XML_CONT_CONS			ERDS_BB_NO_NOT_F_ACC#2:
		Sequence of actions		
#	Sender	ERDS	Receiving side	Where ERDS generates and
1	Sender sends the original message			consigns the structure (including
2		Accepts submission.		consignment metadata user
3	Generates and stores XML_SUB_ACC ERDS evidence			content and
4		Generates {hndvMet, user content, XML_SUB_ACC} structure		SubmissionAcceptance
5		Consigns {hndvMet, user content, XML_SUB_ACC} structure to the receiving side		receiving side and generates the ContentConsignment ERDS
6		Generates and stores XML_CONT_CONS ERDS evidence	{hndvMet, user content, XML_SUB_ACC} structure successfully consigned to the N entities at the receiving side	evidence.

Table 3e: Scenarios for intra-ERDS without notifications for acceptance (6/10)

		Purpose		
Para	ameter: {hndvMet, user content}			
Para	ameter: XML_SUB_ACC	A positive scenario of		
Para	ameter: {NotificationOfConsignment}			ERDS_BB_NO_NOT_F_ACC#2:
Para	ameter: XML_CONS_NOT			
Para	ameter: XML_CONT_CONS			Now the ERDS generates and
		Sequence of actions		sends a notification of
#	Sender	ERDS	Receiving side	side When the notification of
1	Sender sends the original message			consignment is received by N
2		Accepts submission. Generates and stores XML_SUB_ACC ERDS evidence		entities at the receiving side, ERDS generates and stores
3 4	-	Generates {hndvMet, user content} structure		ConsignmentNotification ERDS
5		Consigns {hndvMet, user content} structure to receiving side		
6			{hndvMet, user content} structure successfully consigned to the N entities of the receiving side	
7		Generates and stores XML_CONT_CONS ERDS evidence		
8		Generates {NotificationOfConsignment} for N entities		
9		Sends {NotificationOfConsignment} to receiving side		
10		Generates and stores XML_CONS_NOT ERDS evidence	{NotificationOfConsignment} received by N entities at the receiving side	

Table 3f: Scenarios for intra-ERDS without notifications for acceptance (7/10)

Scenario id: ERDS_BB_NO_NOT_F_ACC#7					Purpose
Par	ameter: {hndvMet, user content}				
Par	ameter: XML_SUB_ACC				A negative scenario of
Par	ameter: {NotificationOfConsignment}				ERDS_BB_NO_NOT_F_ACC#2:
Par	ameter: XML_CONS_NOT				
Par	ameter: XML_CONS_NOT_FAIL				When the notification of
		Sequence	of actions		consignment is not received by
#	Sender		ERDS	Receiving side	receiving side Then FRDS
1	Sender sends the original message				denerates and stores two
2		Accepts sub	mission		evidences:
3		Generates a	nd stores XML_SUB_ACC		ConsignmentNotificationFailure
		ERDS evide	nce		ERDS evidence for one entity
4		Generates {	nndvMet, user content}		and ConsignmentNotification
_		structure for	N entities at the receiving side		ERDS evidence for N-1 entities.
5		Consigns (hi	ndviviet, user content}		
0		structure to	receiving side		_
0				consignments of {muvivel, user	
7		Generates (NotificationOfConsignment} for		_
<i>'</i>		N entities in	receiving side		
8		Consigns {N	otificationOfConsignment} for		
		N entities in	receiving side but one fails		
9				N-1 {NotificationOfConsignment} are successfully received; one is not received	
10		Generates a XML_CONS for one entity ERDS evide	nd stores one _NOT_FAIL ERDS evidence y and XML_CONS_NOT nce for N-1 entities		

Table 3g: Scenarios for intra-ERDS without notifications for acceptance (8/10)

		Purpose		
Para	ameter: {hndvMet, user content}			
Para	ameter: XML_SUB_ACC			A negative scenario of
Para	ameter: XML_CONT_CONS			ERDS_BB_NO_NOT_F_ACC#2:
Para	ameter: XML_CONS_FAIL			
		Sequence of actions		Where one of the ERD dispatch
#	Sender	ERDS	Receiving side	consignments fails. Now, ERDS
1	Sender sends original message			generates and stores two
2		Accepts submission.		ERDS evidence for N-1 entities
3		Generates and stores XML_SUB_ACC ERDS evidence		and ContentConsignmentFailure
4		Generates {hndvMet, user content} structure		FRDS evidence for one entity
5		Consigns {hndvMet, user content} structure to receiving		
		side		
6			N-1 {hndvMet, user content} structures	
			are successfully consigned. One fails	
7		Generates XML_CONT_CONS ERDS evidence for N-1		
		entities and one XML_CONS_FAIL ERDS evidence for one		
		entity and stores them		

Table 3h: Scenarios for intra-ERDS without notifications for acceptance (9/10)

		Purpose		
Para	ameter: {hndvMet, user content}			
Para	ameter: XML_SUB_ACC			The positive scenario:
Para	ameter: XML_CONT_CONS			
Para	ameter: XML_CONT_AC_TRACK			Where the ERDS accepts the
		Sequence of actions		original message, then it
#	Sender	ERDS	Receiving side	generates and consigns the
1	Sender sends original message			metadata and user content) to N
2		Accepts submission.		entities on the receiving side. In
3		Generates and stores XML_SUB_ACC ERDS evidence		the next step, ERDS generates
4		Generates {hndvMet, user content} structure		avidence After the successful
5		Consigns {hndvMet, user content} structure to the N entities at the receiving side	1	retrieval of the ERD payload on
6		Generates and stores XML_CONT_CONS ERDS	{hndvMet, user content} structure correctly consigned to N entities at receiving side	generates and stores ContentAccessTracking ERDS evidence for the N handovers.
7			All the entities retrieve the ERD dispatch	Here, the scenario includes
8		Generates and stores XML_CONT_AC_TRACK ERDS evidence for the N handovers performed	N handovers of {hndvMet, user content} structure succeed	ContentAccessTracking event instead of the handover, as both events provide evidence of the successful delivery of the user content.

Table 3i: Scenarios for intra-ERDS without notifications for acceptance (10/10)

		Purpose			
Para	ameter: {hndvMet, user content, XML_S	UB_ACC}			
Para	ameter: XML_CONT_HAND				A positive scenario of
		Sequence of actions			ERDS_BB_NO_NOT_F_ACC#2:
#	Sender	ERDS		Receiving side	Where ERDS generates and
1	Sender sends the original message				consigns the structure (including
2		Accepts submission.		handover metadata user	
3		Generates and stores XML_SUB_ACC ERDS evidence			content, and
4		Generates {hndvMet, user content, XML_SUB_	ACC}		SubmissionAcceptance
		structure			evidence} to N entities on the
5		Hands over {hndvMet, user content, XML_SUB	_ACC}		receiving side and generates the
		structure to the receiving side			ContentHandover ERDS
6				{hndvMet, user content,	evidence.
		Generates and stores XML_CONT_HAND ERE	DS evidence	handed over to the N entities at the	
				receiving side	

5.3.3 Scenarios with notification for acceptance

Table 4 defines a number of scenarios for the case where sender and the entities at receiving side are subscribed to the same ERDS and the ERDS sends notification for acceptance to the entities at the receiving side.

		Purpose			
Para	meter: {hndvMet, user content}				
Para	meter: {NotificationForAcceptanc	The simplest positive scenario:			
Para	meter: XML_NOT_F_ACC				
Para	meter: XML_CONS_ACC				Where the ERDS accepts the
Para	meter: XML_NOT_DEL				original message, then it
Para	meter: XML_CONT_CONS				generates and sends a structure
		Sequence of	factions		containing
#	Sender	ERDS		Receiving side	NotificationForAcceptance to the
1	Sender sends original				Tecenning side.
1	message				When the ERDS has succeeded
2		Accepts submission			in delivering these
3		Generates and stores XML_SUB_A	CC ERDS evidence		NotificationForAcceptance, it
4		Generates (NotificationForAcceptan	ice}		generates one
5		Generates and stores XML_NOT_F	_ACC ERDS evidence		NotificationDelivered for each
6		Sends {NotificationForAcceptance}			entity in the receiving side.
7				All entities in receiving side receive one	
				{NotificationForAcceptance} and	When N entities successfully
				answer positively	receive and respond positively to
8		Generates and stores XML_NO1_D	EL evidence for the N		the notification of acceptance at
0		entities at receiving side	and Constant		the receiving side, ERDS
9		Receives positive responses from re	eceiving side and Generates		Generates and stores their
		and stores XML_CONS_ACC ERDs	S evidence for the in entities		evidence
10		Generates (bndy/Met_user content)	structure		
11		Consigns (hndy/Met, user content) to	N entities in receiving side		Afterwards, ERDS generates
12					and consigns the structure
12					{Including handover metadata
					and user content} to N entities
		Constates and stores XML_CONT	CONS ERDS ovidence	N consignments of {hndvMet, user	on the receiving side. The cycle
		Generates and stores XML_CONT_	CONS ERDS evidence	content} succeed	of message events is finalized
					with the generation of the
					ContentConsignment ERDS
1					levidence for N entities.

Table 4: Scenarios for intra-ERDS with notifications for acceptance (1/7)

23

Table 4a: Scenarios for intra-ERDS with notifications for acceptance (2/7)

		Purpose		
Para	meter: {hndvMet, user content}			
Para	meter: {NotificationForAcceptance}	A negative scenario of		
Para	meter: XML_SUB_ACC			ERDS_BB_NOT_F_ACC#1:
Para	meter: XML_NOT_F_ACC			
Para	meter: XML_CONS_ACC			The notification of acceptance is
Para	meter: XML_CONS_REJ			successfully received by N
Para	meter: XML_NOT_DEL			entities on the receiving side,
Para	meter: XML_CONT_CONS			where one entity rejects the
		Sequence of actions		consignment. Now ERDS
#	Sender	ERDS	Receiving side	evidences.
1	Sender sends original message			ConsignmentAcceptance FRDS
2		Accepts submission.		evidence for N-1 entities and
3		Generates and stores XML_SUB_ACC ERDS evidence		ConsignmentRejection ERDS
4		Generates {NotificationForAcceptance} for N entities at		evidence for 1 entity.
		receiving side		
5 Sends {NotificationForAcceptance} to N entities at receiving			Afterwards, ERDS generates	
		side		and consigns the structure
6			N entities correctly receive	{Including handover metadata
		Generates and stores XML_NO1_F_ACC ERDS evidence	{NotificationForAcceptance}. N-1	and user content} to N-1 entities
for N entit		for N entities	accept. One does not accept	of massage events is finalized
7		Concretes and stores XML NOT DEL suidenes for the N	consignment	with the deperation of the
ľ		Generates and stores XML_NOT_DEL evidence for the N		ContentConsignment ERDS
8		Receives N-1 positive answers and one pegative answer		evidence for N-1 entities.
q		Generates XML_CONS_ACC_ERDS evidence for N-1		
J J		entities and one XML_CONS_RELERDS evidence for 1		
		entity		
10 Generates {hndvMet, user content} structure for N-1 entities				
11 Consigns them to the N-1 entities at receiving side				
12		7		
		Generates XML_CONT_CONS for N-1 entities	consigned to N-1 entities at receiving	
			side	
13		Generates XML_CONT_CONS ERDS evidence for N-1		
		entities		

Table 4b: Scenarios for intra-ERDS with notifications for acceptance (3/7)

Scenario id: ERDS_BB_NOT_F_ACC#3					Purpose
Param	neter: {hndvMet, user content}				
Param	neter: {NotificationForAcceptance}	A negative scenario similar to			
Param	neter: XML_SUB_ACC				ERDS_BB_NOT_F_ACC#1:
Param	neter: XML_NOT_F_ACC				
Param	neter: XML_CONS_ACC				Where ERDS does not receive
Param	neter: XML_ACC_REJ_EXP				response from one entity after
Param	neter: XML_NOT_DEL				NotificationEarAcceptance
Param	neter: XML_CONT_CONS				ERDS generates and stores two
		Sequence of ac	tions		evidences.
#	Sender	ERDS		Receiving side	ConsignmentAcceptance ERDS
1	Sender sends original message				evidence for N-1 entities and
2		Accepts submission			ConsignmentRejection ERDS
3		Generates and stores XML_SUE	3_ACC ERDS evidence		evidence for 1 entity.
4		Generates {NotificationForAccep	otance} for N entities at		
		receiving side			Afterwards, ERDS generates
5		Sends {NotificationForAcceptance	ce} to N entities at		and consigns the structure
		receiving side			{Including handover metadata
6		Generates and stores XML NOT	T F ACC ERDS	N entities correctly receive	and user content} to N-1 entities
		evidence for N entities		{NotificationForAcceptance}. N-1	on the receiving side. The cycle
7			T DEL ovidence for the	accept. One does not answer in time	with the dependion of the
'		Generates and stores AML_NOT	I_DEL evidence for the		ContentConsignment ERDS
8		Receives N-1 positive answers			evidence for N-1 entities and
0	Concretes (hpd)/Mot_user content) structure for N 1			AcceptanceRejectionExpirv	
9		entities			ERDS evidence for one entity.
10		Consigns {hndvMet_user conten	ot} structure to the N-1		-
		entities that have accepted			
11		When the expiration time is reac	hed Generates and		
		stores XML_CONS_ACC ERDS	evidence for N-1 entities	{hndvMet, user content} structure	
		and one XML_ACC_REJ_EXP E	ERDS evidence for one	successibility consigned to IN-1 entitles	
		entity		at receiving side	

Table 4c: Scenarios for intra-ERDS with notifications for acceptance (4/7)

		Purpose			
Para	ameter: {hndvMet, user content}				
Parameter: {NotificationForAcceptance}					A positive scenario of
Para	ameter: XML_NOT_F_ACC				ERDS BB NOT F ACC#3:
Para	ameter: XML_CONS_ACC				
Para	ameter: XML_CONT_CONS				After the successful handover of
Para	ameter: XML_NOT_DEL				the payload at receiving side, the
Para	ameter: XML_CONT_HAND				cycle of message events is
		Sequence of	f actions		completed with the generation
#	Sender	ERDS	i	Receiving side	EPDS ovidence for N optitios
4	Sender sends original				ERDS evidence for N entities.
1	message				
2		Accepts submission.]
3		Generates and stores XML_SUB_A	CC ERDS evidence		
4		Generates {NotificationForAcceptar	nce}		
5		Generates and stores XML_NOT_F	_ACC ERDS evidence		
6		Sends {NotificationForAcceptance}			
7				All entities in receiving side receive one	
				{NotificationForAcceptance} and	
				answer positively	_
8		Generates and stores XML_NOT_D	DEL evidence for the N		
		entities at receiving side			-
9		Receives positive responses from re	eceiving side and Generates		
		and stores XML_CONS_ACC ERDS	S evidence for the N entities		
4.0		at receiving side	- 1		4
10		Generates {hndviviet, user content}	structure		4
11		Consigns (nndviviet, user content) to	o in entities in receiving side		4
12		Generates and stores XML_CONT_	CONS ERDS evidence	IN consignments of (nnaviviet, user	
12				N handovers of (hndvMet, uper content)	4
13				etructure succeed	
14		Concrates and stores XML_CONT	HAND ERDS evidence for N		4
14		entities			

Table 4d: Scenarios for intra-ERDS with notifications for acceptance (5/7)

		Purpose		
Para	meter: {hndvMet, user content}			•
Parameter: {NotificationForAcceptance}				A negative scenario of
Para	meter: XML_NOT_F_ACC			ERDS BB_NOT_F_ACC#4:
Para	meter: XML_CONS_ACC			
Para	meter: XML_CONT_CONS			After the occurrence of N-1
Para	meter: XML_CONT_HAND			successful and 1 failed handover
Para	meter: XML_NOT_DEL			of the payload at the receiving
Para	meter: XML_CONT_HAND_FAIL			side, the cycle of message
		Sequence of actions		events is completed with the
#	Sender	ERDS	Receiving side	ContentHandover ERDS
	Sender sends original			evidence for N-1 entities and
1	message			ContentHandoverFailure for 1
2		Accepts submission.		entity.
3		Generates and stores XML_SUB_ACC ERDS evidence		
4		Generates {NotificationForAcceptance}		
5		Generates and stores XML_NOT_F_ACC ERDS evidence		
6		Sends {NotificationForAcceptance}		
7			All entities in receiving side receive one	
			{NotificationForAcceptance} and	
			answer positively	
8		Generates and stores XML_NOT_DEL evidence for the N		
		entities at receiving side		
9		Receives positive responses from receiving side and Generates		
		and stores XML_CONS_ACC ERDS evidence for the N entities		
10		at receiving side		-
10		Generates {hndvMet, user content} structure		-
11		Consigns {hndvMet, user content} to N entities in receiving side		-
12		Generates and stores XML_CONT_CONS ERDS evidence	N consignments of {hndvMet, user content} structure succeed	
13			N-1 handovers of {hndvMet, user	
			content} structure succeed. One fails	
14		Generates and stores XML_CONT_HAND ERDS evidence for N-1 entities, and XML_CONT_HAND_FAIL for 1 entity		

Table 4e: Scenarios for intra-ERDS with notifications for acceptance (6/7)

		Purpose			
Para Para Para Para Para Para Para Para	meter: {hndvMet, user content} meter: {NotificationForAcceptance} meter: XML_SUB_ACC meter: XML_NOT_F_ACC meter: XML_CONS_ACC meter: XML_CONS_REJ meter: XML_CONT_CONS meter: XML_CONT_HAND meter: XML_NOT_DEL	Scenario id: ERDS_	BB_NOT_F_ACC#6		Purpose A negative scenario similar to the combination of ERDS_BB_NOT_F_ACC#2 and ERDS_BB_NOT_F_ACC#5: The notification of acceptance is successfully received by N entities on the receiving side, one entity rejects the
Para	meter: XML_CONT_HAND_FAIL	Soguonoo	of actions		consignment. Now, ERDS
#	Sender	Sequence	of actions	Receiving side	generates and stores two
1	Sender sends original message		5	Receiving side	evidences;
2	Gender sends ofiginal message	Accepts submission.			evidence for N-1 entities and
3		Generates and stores XML_SL	JB_ACC ERDS evidence		ConsignmentRejection ERDS
4		Generates {NotificationForAcce	eptance} for N entities at		evidence for 1 entity.
		receiving side			
5		Sends {NotificationForAccepta	nce} to N entities at receiving		Afterwards, ERDS generates
6 6 Generates and stores XML_NOT_F_ACC ERDS evid for N entities		DT_F_ACC ERDS evidence	N entities correctly receive {NotificationForAcceptance}. N-1 accept. One does not accept consignment	{Including handover metadata and user content} to N-1 entities on the receiving side. Then, ERDS generates	
7		Generates and stores XML_NC entities at receiving side	DT_DEL evidence for the N		ContentConsignment ERDS evidence for N-1 entities and
8		Receives N-1 positive answers	and one negative answer		ConsignmentRejection ERDS
9		Generates XML_CONS_ACC I entities and one XML_CONS_I entity	ERDS evidence for N-1 REJ ERDS evidence for 1		evidence for one entity. After the N-2 successful and 1
10		Generates {hndvMet, user con	tent} structure for N-1 entities		failed handover of the payload at
11		Consigns them to the N-1 entit	ies at receiving side		receiving side, the cycle of
12		Generates XML_CONT_CONS entities	SERDS evidence for N-1	N-1 {hndvMet, user content} correctly consigned to N-1 entities at receiving side	with the generation and storing of ContentHandover ERDS
13				N-2 handovers of {hndvMet, user	ContentHandoverFailure for 1
14		Generates and stores XML_CO	DNT_HAND ERDS evidence	content} structure succeed. One fails	entity.

Table 4f: Scenarios for intra-ERDS with notifications for acceptance (7/7)

		Purpose			
Para	ameter: {hndvMet, user content}				
Para	ameter: {NotificationForAcceptance	e}			A positive scenario of
Para	ameter: XML_NOT_F_ACC				ERDS_BB_NOT_F_ACC#3:
Para	ameter: XML_CONS_ACC				After the successful handover of
Para	ameter: XML_CONT_CONS				the payload at receiving side, the
Para	ameter: XML_NOT_DEL				cycle of message events is
Para	ameter: XML_CONT_AC_TRACK				completed with the generation
		Sequence	of actions		and storing of
#	Sender	ERD	S	Receiving side	ContentAccess Fracking ERDS
	Sender sends original			.	evidence for N entities.
1	message				Here, the scenario includes
2		Accepts submission.			ContentAccessTracking event
3		Generates and stores XML_SUB_	ACC ERDS evidence		instead of the handover, as both
4		Generates {NotificationForAccepta	ance}		events provide evidence of
5		Generates and stores XML_NOT_	F_ACC ERDS evidence		the successful delivery of the
6		Sends (NotificationForAcceptance)	}		user content.
7				All entities in receiving side receive one	
				{NotificationForAcceptance} and	
				answer positively	
8		Generates and stores XML_NOT_	DEL evidence for the N		
_		entities at receiving side			-
9		Receives positive responses from	receiving side and Generates		
		and stores XML_CONS_ACC ERL	DS evidence for the N entities		
10		at receiving side			-
10		Generates (Indviviet, user content	<pre>} structure to N antitica in reactiving cide</pre>		-
11		Consigns {nnaviviet, user content}	to in entities in receiving side	N consignments of (hnd)/Mat. user	
12		Generates and stores XML_CONT	_CONS ERDS evidence	in consignments of (movivel, user	
13				N handovers of (hndvMet user content)	4
13				structure succeed	
14		Generates and stores XML_CONT	AC TRACK FRDS evidence		4
' ⁻		for N entities			

5.4 Scenarios for 4-corner model

5.4.1 Introduction

The present clause defines test cases for scenarios that take place when the sender and the entities at the receiving side are subscribed to different ERDSs but there are not intermediate ERDSs between the SERDS and the RERDS.

30

Clause 5.4.2 defines test cases when RERDS does not notify for acceptance.

Clause 5.4.3 defines test cases when RERDS notifies for acceptance.

5.4.2 Scenarios where RERDS does not use notification for acceptance

Table 5 defines a number of scenarios for the case where RERDS does not notify for acceptance.

The scenarios are based on scenarios at Table 3 adding the relay acceptance and relay rejection events at the some of the RERDSs and their corresponding ERDS evidences.

For the sake of simplicity, it will be supposes that all the entities at receiving side are served by the same RERDS. It could be possible to use the templates defined in the present document for defining scenarios where the aforementioned entities are served by different RERDSs.

Table 5: Scenarios for RERDS not using notification for acceptance (1/7)

		Purpose			
Parameter: {hndvMet, EV_SET#1} Var EV_SET#1 = {XML_SUB_ACC, XML_REL_REJ}					
		The simplest negative scenario:			
#	Sender	SERDS	RERDS	Receiving side	W/bara SERDS apports the
1	Sender sends original message				where SERDS accepts the
2		Accepts submission			relies FRD dispatch message to
		Generates and stores XML_SUB_ACC ERDS evidence			RERDS. RERDS rejects the
3		Generates <erd_dispatch>_WITH_XML_SUB_ ACC</erd_dispatch>			submitted by the SERDS and sends back the sender a structure with the set of ERDS
4		Relies the <erd_dispatch>_WITH_XML_SUB_ ACC to the RERDS</erd_dispatch>			evidences generated by SERDS. (Including the XML_REL_REJ).
5	Receives ERDS receipt		The RERDS Rejects the <erd_dispatch>_WITH_XML_SU B_ACC</erd_dispatch>		
6		Generates and stores XML_REL_REJ ERDS evidence			
7		Generates {hndvMet, EV_SET#1} structure]	
8		Sends {hndvMet, EV_SET#1} back to sender			
9	Sender receives {hndvMet, EV_SET#1}				

Table 5a: Scenarios for RERDS not using notification for acceptance (2/7)

		Purpose			
Para	ameter: <erd_dispatch_1>_WITH</erd_dispatch_1>	_XML_SUB_ACC			
Para	ameter: <erds_receipt_1>_WITH</erds_receipt_1>	The simplest positive scenario:			
Para	ameter: {hndvMet, user content}				
Para	ameter: <erds_receipt_2>_WITH</erds_receipt_2>		The SERDS accepts the		
			submission of the original		
#	Sender	SERDS	RERDS	Receiving side	message, then it generates and
1	Sender sends original message				message RERDS a dispatch
2		Accepts submission. Generates and			relay message and sends back a
		stores XML_SUB_ACC ERDS			receipt message containing
		evidence			RelayAcceptance evidence.
3		Generates			Afterwards, RERDS generates
		<pre> <erd_dispatcn_1>_WITH_XIVIL_SU A C C</erd_dispatcn_1></pre>			and consigns a structure
1	+	Balias			containing user content to N
4		<pre><frd 1="" dispatch=""> WITH XML_SUL</frd></pre>			recipients on receiving side. If
		B ACC to RERDS			consigned successfully to N
5			Accepts		RERDS generates and sends
			<erd_dispatch_1>_WITH_XML_S</erd_dispatch_1>		back the receipt containing
			UB_ACC and generates and stores		ContentConsignment evidence
			XML_REL_ACC ERDS evidence		to SERDS. SERDS generates
6			Generates		and sends back the sender ERD
			<pre><erds_receipt_1>_WITH_XML_R</erds_receipt_1></pre>		receipt message with
7	+		EL_ACC		ContentConsignment ERDS
1			Sends		evidence.
			EL ACC to SERDS		Clourse 4.2.2 of ETSI EN 210
8		Receives	Generates {hndvMet, user content}		522-1 [1] shows a variation of
Ũ		<pre><erds 1="" receipt=""> WITH XML RE</erds></pre>	struct		this scenario where RERDS
		L_ACC from RERDS			sends XML REL ACC and
9			Consigns {hndvMet, user content}		XML_CONT_CONS in different
			to the receiving side		ERD receipts.
10			Generates XML_CONT_CONS	N consignments of {hndvMet,	
			ERDS evidence	user content} structure	
4.4	+	+	Conceptor	succeed	4
1.1			CERDS receipt 2 WITH VML C		
			ONT CONS		
12	+	+	Sends	1	4
			<pre><erds 2="" receipt=""> WITH XML C</erds></pre>		
			ONT_CONS back to the SERDS		

13	Recei <erd NT_C sende</erd 	eives DS_receipt_2>_WITH_XML_CO CONS and sends it back to the der		
14	Store ERDS	es the XML_CONT_CONS DS evidence for N entities		

Table 5b: Scenarios for RERDS not using notification for acceptance (3/7)

Scenario id: 4C_RERDS_NO_NOT_F_ACC#3					Purpose
Parameter: <erd_dispatch_1>_WITH_XML_SUB_ACC Var EV_SET#1 = {XML_CONT_CONS}</erd_dispatch_1>					•
Parameter: {hndvMet, user content}					A positive scenario of
Para	ameter: <erds_receipt_1>_WITH</erds_receipt_1>	_EV_SET#1			4C_RERDS_NO_NOT_F_ACC
Parameter: Generates <erd_receipt_2>_WITH_CONT_HAND</erd_receipt_2>					<u>#2</u> :
#	Sender	ERDS	RERDS	Receiving side	Vvnere both the
1	Sender sends original message			Sender sends original	ContentConsignment evidences
2		Accepts submission Generates and			are included in the same ERD
–		stores XML_SUB_ACC_ERDS			te SERDS, When the user
		evidence			to SERDS. When the user
3		Generates			by entities on the receiving side
		<erd_dispatch_1>_WITH_XML_SU</erd_dispatch_1>			the cycle of message events is
		B_ACC			completed with the generation of
4		Relies			a receipt message containing
		<pre> <erd_dispatch_1>_WITH_XML_SU</erd_dispatch_1></pre>			ContentHandover ERDS
5			Acconto		evidence and sends it back to
5			<pre><frd 1="" dispatch=""> WITH XML S</frd></pre>		SERDS.
			UB ACC and generates and stores		
			XML_REL_ACC ERDS evidence		
6			Generates {hndvMet, user content}		
			structure for N recipients		
7			Consigns {hndvMet, user content}		
			structure to the receiving side		
8			Generates XML_CONT_CONS	N consignments of {hndvMet, user content} structure	
			ERDS evidence for N entities	succeed	
9			Generates		
			<erds_receipt>_WITH_XML_EV_</erds_receipt>		
			SET#1		_
10				Entities in receiving side	
			<pre><erds_receipt_1>_WITH_XML_E</erds_receipt_1></pre>	retrieve user content	
11		Pacaivas	V_3E1#110 3ERD3		-
1		<pre><frds 1="" receipt=""> WITH EV SET</frds></pre>	Generates XML_CONT_HAND		
		#1 and stores the evidences within	ERDS evidence for N entities		
		the set			
12			Generates		
1			<erd_receipt_2>_WITH_CONT_H</erd_receipt_2>		
1			AND		

Scenario id: 4C_RERDS_NO_NOT_F_ACC#3					Purpose
13			Sends <erd_receipt_2>_WITH_CONT_H AND to SERDS</erd_receipt_2>		
14		Receives <erds_receipt_2>WITH_XML_C ONT_HAND and stores the ERDS evidence</erds_receipt_2>			

Table 5c: Scenarios for RERDS not using notification for acceptance (4/7)

Scenario id: 4C_RERDS_NO_NOT_F_ACC #4					Purpose
Parameter: <erd_dispatch_1>_WITH_XML_SUB_ACC Var EV_SET#1 = {XML_REL_ACC, XML_CONT_CONS}</erd_dispatch_1>					1
Parameter: {hndvMet, user content} Var EV_SET#2 = {XML_CONT_HAND, XML_CONT_HAND_FAIL}					A negative scenario of
Parameter: <erds_receipt_1>_WITH_XML_EV_SET#1</erds_receipt_1>					4C_RERDS_NO_NOT_F_ACC#
Para	ameter: ERDS_receipt_2>_WITH_	<u>3</u> :			
#	Sender	SERDS	RERDS	Receiving side	Where one of the handovers
1	Sender sends original message				Talls, the RERDS sends back a
2		Accepts submission. Generates			containing a set of evidence
		XML_SUB_ACC ERDS evidence			(including ContentHandover and
3		Generates			ContentHandoverFailure
		<erd_dispatch_1>_WITH_XML_SU</erd_dispatch_1>			evidence).
<u> </u>		B_ACC			-
4		Relies			
		B ACC to RERDS			
5			Accepts		1
Ŭ			<pre><erd 1="" dispatch=""> WITH XML S</erd></pre>		
			UB_ACC and generates and stores		
			XML_REL_ACC ERDS evidence		
6			Generates {hndvMet, user content}		
			structure for N recipients		
7			Consigns {hndvMet, user content}		
			to the receiving side		-
8			Generates XML CONT CONS	N consignments of {hndvMet,	
			ERDS evidence	user content} structure	
0			Concretes	succeed	-
9			ERDS receipt 1 WITH EV SE		
10			Sends	N – 1 entities in receiving	1
			<pre><erds 1="" receipt=""> WITH EV SE</erds></pre>	side successfully retrieve	
			T#1 to SERDS	user content. One fails	
11			Generates XML_CONT_HAND		1
		Receives	ERDS evidence for N-1 entities		
		<erds_receipt>_WITH_XML_EV_S</erds_receipt>	and one XML_CONT_HAND_FAIL		
		ET#1 and stores ERDS evidences	ERDS evidence for one entity and		
			stores them		4
12			Generates		
12			#2 Sands		1
15			<pre>CERD receipt 2> WITH EV SET</pre>		
			#2 to SERDS		
	Scenario id: 4C_RERDS_N	Purpose			
----	---	---------	--		
14	Receives <erd_receipt_2>_WITH_EV_SET#2 and stores ERDS evidences</erd_receipt_2>				

Table 5d: Scenarios for RERDS not using notification for acceptance (5/7)

		Purpose			
Para	ameter: <erd_dispatch_1>_WITH</erd_dispatch_1>	i			
Para	ameter: {hndvMet, user content}		•	÷	A negative scenario
Para	ameter: <erds_receipt_1>_WITH</erds_receipt_1>	_XML_REL_ACC to SERDS			4C_RERDS_NO_NOT_F_ACC
Para	ameter: <erds_receipt_2>_WITH</erds_receipt_2>	_XML_EV_SET#1			<u>#2</u> :
		Sequence of	actions		Whore one of the EPD dispatch
#	Sender	SERDS	RERDS	Receiving side	consignments fails. The RERDS
1	Sender sends original message				sends back a set of evidence to
2		Accepts submission. Generates and stores XML_SUB_ACC ERDS			SERDS (including N-1 ContentConsignment and 1
		evidence			ContentConsignmentFailure
3		Generates			evidence) in a receipt message.
		B_ACC			
4		Relies			
		<pre> <erd_dispatch_1>_WITH_XML_SU B_ACC to RERDS</erd_dispatch_1></pre>	J		
5			Accepts		
			<pre><erd_dispatch_1>_WITH_XML_S</erd_dispatch_1></pre>		
			UB_ACC and generates and stores		
6			Generates		
0			<pre><frds 1="" receipt=""> WITH XML R</frds></pre>		
			EL_ACC		
7			Sends		
			<erds_receipt_1>_WITH_XML_R</erds_receipt_1>		
			EL_ACC to SERDS		
8		Receives	Generates {hndvMet. user content}		
		<erds_receipt_1>_WITH_XML_RE</erds_receipt_1>	structure for N entities		
a			Consigns (hnd)/Met_user content)		
Ŭ			structure to receiving side		
10	1			N-1 consignments of	
				{hndvMet, user content}	
				structure succeed. One	
				consignment fails	
11			Generates XIVIL_CONT_CONS		
			entities		
			Generates		
			XML CONT CONS FAIL related		
			to one entity		

	Scenario id: 4C_RERDS_I	Purpose	
12		Generates <erds_receipt_2>_WITH_XML_E V_SET#1</erds_receipt_2>	
13		Sends <erds_receipt_2>_WITH_XML_E V_SET#1 back to the SERDS</erds_receipt_2>	
14	Receives <erds_receipt_2>_WITH_XML_EV _SET#1 and stores ERDS evidences</erds_receipt_2>		

The set of scenarios shown above could very easily be used for generating other scenarios where ERD payloads replace ERD dispatches.

Below follows the generation of a new scenario resulting from replacing ERD dispatches by ERD payloads in scenario 4C_RERDS_NO_NOT_F_ACC #5. Differences between both scenarios have been highlighted in yellow.

Table 5e: Scenarios fo	r RERDS not using	notification for	acceptance (6	5/7)
------------------------	-------------------	------------------	---------------	------

		Purpose			
Para	meter: <erd_payload_1></erd_payload_1>	Va	ar EV_SET#1 = {CONT_CONS, CON_(CONS_FAIL}	
Para	meter: {hndvMet, user content}				A scenario similar to
Para	meter: <erds_receipt_1>_WITH_</erds_receipt_1>	_XML_REL_ACC to SERDS			4C RERDS NO NOT F ACC
Para	meter: <erds_receipt_2>_WITH_</erds_receipt_2>	_XML_EV_SET#1			<u>#5</u> :
		Sequence of	actions		Where SERDS generates and
#	Sender	SERDS	RERDS	Receiving side	sends payload ERD message to
1	Sender sends original message				BERDS and BERDS accepts
2		Accepts submission. Generates and stores XML_SUB_ACC ERDS evidence			this relay message.
3		Generates <erd_payload_1></erd_payload_1>			
4		Relies < ERD_payload_1 > to RERDS	6		
5			Accepts <erd_payload_1> and generates and stores XML_REL_ACC ERDS evidence</erd_payload_1>		
6			Generates <erds_receipt_1>_WITH_XML_R EL_ACC</erds_receipt_1>		
7			Sends <erds_receipt_1>_WITH_XML_R EL_ACC to SERDS</erds_receipt_1>		
8		Receives <erds_receipt_1>_WITH_XML_RE L_ACC to SERDS</erds_receipt_1>	Generates {hndvMet, user content} structure for N entities		
9			Consigns {hndvMet, user content} structure to receiving side		
10				N-1 consignments of {hndvMet, user content} structure succeed. One consignment fails	
11			Generates XML_CONT_CONS ERDS evidence related to N-1 entities Generates XML_CONT_CONS_FAIL related to one entity		
12			Generates <erds_receipt_2>_WITH_XML_E V_SET#1</erds_receipt_2>		

	Scenario id: 4C_RER	Purpose		
13		Sends		
		<erds_receipt_2>_WITH_XML_E</erds_receipt_2>		
		V_SET#1 back to the SERDS		
14	Receives			
	<erds_receipt_2>_WITH_XML</erds_receipt_2>	EV		
	_SET#1 and stores ERDS evide	ences		

Table 5f: Scenarios for RERDS not using notification for acceptance (7/7)

		Purpose			
Para	ameter: <erd_dispatch_1>_WITH</erd_dispatch_1>				
Para	ameter: {hndvMet, user content}	Var	EV_SET#2 = {XML_CONT_AC_TRA	CK,	A negative scenario of
		XMI	CONT_HAND_FAIL}		4C_RERDS_NO_NOT_F_ACC#
Para	ameter: <erds_receipt_1>_WITH</erds_receipt_1>	_XML_EV_SET#1			<u>3</u> :
Para	ameter: ERDS_receipt_2>_WITH_	XML_EV_SET#2	-		Whore one of the reasilying
		Sequence of a	ctions		entities fails in retrieving the user
#	Sender	SERDS	RERDS	Receiving side	content, the RERDS sends back
1	Sender sends original message				a receipt message to SERDS
2		Accepts submission. Generates XML SUB ACC ERDS evidence			containing a set of evidence
3		Generates			ContentAccessTracking and
		<erd_dispatch_1>_WITH_XML_SU</erd_dispatch_1>			ContentConsignmentFailure
4		Belies			evidence).
		<pre><erd_dispatch_1>_WITH_XML_SU</erd_dispatch_1></pre>			Here, the scenario includes
F		B_ACC to RERDS	Accento		ContentAccessTracking event
э			<pre>Accepts <frd 1="" dispatch=""> WITH XML S</frd></pre>		instead of the handover, as both
			UB ACC and generates and stores		events provide evidence of
			XML REL ACC ERDS evidence		user content
6			Generates {hndvMet, user content}		
			structure for N recipients		
7			Consigns {hndvMet, user content} to the receiving side		
8			Generates XML_CONT_CONS	N consignments of {hndvMet,	
			ERDS evidence	succeed	
9			Generates		
			<erds_receipt_1>_WITH_EV_SE T#1</erds_receipt_1>		
10			Sends	N - 1entities in receiving side	
			<erds_receipt_1>_WITH_EV_SE</erds_receipt_1>	successfully retrieve user	
			T#1 to SERDS	content. One fails	-
11			Generates		
		Receives	XML_CONT_AC_TRACK ERDS		
		<erds_receipt>_WITH_XML_EV_S</erds_receipt>	XMI CONT HAND FAIL FRDS		
		ET#1 and stores ERDS evidences	evidence for one entity and stores		
1			them		
12			Generates]
			<erd_receipt_2>_WITH_EV_SET</erd_receipt_2>		
			#2		

	Scenario id: 4C_RER	Purpose		
13		Sends <erd_receipt_2>_WITH_EV_SET #2 to SERDS</erd_receipt_2>		
14	Receives <erd_receipt_2>_WITH_EV_S and stores ERDS evidences</erd_receipt_2>	ET#2		

5.4.3 Scenarios where RERDS uses notification for acceptance

Table 6 defines a number of scenarios for the case where RERDS uses notification for acceptance.

The scenarios are based on scenarios at Table 4 adding the relay acceptance and relay rejection events at the some of the RERDSs and their corresponding ERDS evidences.

44

For the sake of simplicity, it is supposed that all the entities at receiving side are served by the same RERDS. It could be possible to use the templates defined in the present document for defining scenarios where the aforementioned entities are served by different RERDSs.

Again, a similar set of scenarios can be obtained by replacing ERD dispatches with ERD payloads in the scenarios shown below.

Table 6: Scenarios where RERDS uses notification for acceptance (1/6)

		Purpose			
Para	ameter: <erd_dispatch_1></erd_dispatch_1>	The simplest positive scenario:			
Para	ameter: {hndvMet, user con	itent}			The SERDS accepts the
Para	ameter: <erds_receipt_1></erds_receipt_1>	-WITH_XML_REL_ACC			submission of the original
Para	ameter: {NotificationforAcce	eptance}			message, then it generates and
Para	ameter: <erds_receipt_2></erds_receipt_2>	-WITH_XML_EV_SET#1			relies RERDS a dispatch
		Sequence o	factions		message. RERDS accepts the
#	Sender	SERDS	RERDS	Receiving side	relay message and sends back a
	Sender sends original				receipt message containing
1	message				RelayAcceptance evidence.
2		Accepts submission. Generates and			-RERDS generates and sends
–		stores XML SUB ACC ERDS evidence			the NotificationForAcceptance to
3		Generates			- N entities at receiving side along
-		<erd 1="" dispatch=""> WITH XML SUB</erd>			NotificationEarAccontanco
		ACC			avidence RERDS generates
4		Relies			
		<erd_dispatch_1>_WITH_XML_SUB_</erd_dispatch_1>			evidence as all the entities on
		ACC to RERDS			receiving side responded
5			Accepts		positively
			<erd_dispatch_1>_WITH_XML_SUB</erd_dispatch_1>		peenivery
			_ACC to RERDS		Afterwards, RERDS generates
6			Generates and stores XML_REL_ACC		and consigns a structure
			ERDS evidence		containing user content to N
7			Generates		recipients on receiving side.
			<erds_receipt_1>_WITH_XML_REL</erds_receipt_1>		After the successful
			_ACC		consignment to N entities,
8			Sends		RERDS generates
			<erd_receipt_1>_WITH_XML_REL_</erd_receipt_1>		ContentConsignment evidence.
			ACC back to SERDS		RERDS generates
9		Receives	Generates (NotificationForAccentance)		ContentHandover evidence for
		<pre><erds_receipt_1>_WITH_XML_REL_A</erds_receipt_1></pre>	struct		all receiving entities upon
		CC from RERDS			successful retrieval of the user
10			Sends {NotificationForAcceptance}		content. RERDS generates and
			struct to receiving side		sends back the SERD a receipt
11			Generates XML_NOT_F_ACC ERDS	All entities in receiving side	message with the set of
			evidence	answer positively	
12			Generates XML_NOT_DEL ERDS		XML_NOT_F_ACC,
			evidence for the N entities at receiving		XIVIL_INUT_DEL,
			side.		XIVIL_CONS_ACC,
1					XIVIL_CONT_LONS,
1			Generates and stores		INIL_CONT_MAND,
1			IXML NOT AC TRACK ERDS	1	IVIL_INUT_AU_TRAUN).

	Scenario id: 4C_REDS_NOT_F_ACC#1				
		evidence for the N entities at receiving side			
13		Generates XML_CONS_ACC ERDS evidence			
14		Generates {hndvMet, user content} struct for N entities			
15		Consigns {hndvMet, user content} struct to receiving side			
16			N consignments of {hndvMet, user content} struct succeed		
17		Generates XML_CONT_CONS ERDS evidence	All the entities retrieve user content		
18		Generates XML_CONT_HAND ERDS evidence			
19		Generates <erds_receipt_2>_WITH_EV_SET#1</erds_receipt_2>			
20		Sends <erds_receipt_2>_WITH_EV_SET#1 to SERDS</erds_receipt_2>			
21	Receives <erds_receipt_2>_WITH_EV_SET#1 and stores ERDS evidences</erds_receipt_2>				

Table 6a: Scenarios where RERDS uses notification for acceptance (2/6)

		Purpose			
Para	Var EV_SET#1 = {XML_NOT_F_ACC, XML_NOT_DEL, Parameter: {hndvMet, user content} XML_CONS_ACC, XML_CONS_REJ, XML_CONT_CONS, XML_CONT_HAND}			L_NOT_DEL, CONT_CONS,	A negative scenario of <u>4C REDS NOT F ACC#1</u> :
Para	ameter: <erd_dispatch_1></erd_dispatch_1>	_WITH_XML_SUB_ACC			Where one of the entities on the
Para	ameter: <erds_receipt_1>_</erds_receipt_1>	_WITH_XML_REL_ACC			receiving side responds
Para	ameter: {NotificationforAcce	otance}			negatively to the
Para	ameter: <erds_receipt_2>_</erds_receipt_2>	_WITH_EV_SET#1			NoticationForAcceptance. The
		Sequence of	of actions		RERDS generates and stores
#	Sender	SERDS	RERDS	Receiving side	
1	Sender sends original message				ConsignmentRejection evidence.
2		Accepts submission. Generates and stores XML_SUB_ACC ERDS evidence			a structure containing user
3		Generates <erd_dispatch_1>_WITH_XML_SUB_ ACC</erd_dispatch_1>			receiving side. Upon the successful consignment to N -1
4		Relies <erd_dispatch_1>_WITH_XML_SUB_ ACC to RERDS</erd_dispatch_1>			RERDS generates ContentConsignment evidence.
5			Accepts <erd_dispatch_1>_WITH_XML_SUB _ACC to RERDS</erd_dispatch_1>		After the successful retrieval of the user content, RERDS generates ContentHandover
6			Generates XML_REL_ACC ERDS		evidence for N-1 receiving
7			Generates <erds_receipt_1>_WITH_XML_REL ACC</erds_receipt_1>		sends back the SERD a receipt message with the set of evidence (including
8			Sends <erd_receipt_1>_WITH_XML_REL_ ACC back to SERDS</erd_receipt_1>		XML_NOT_F_ACC, XML_NOT_DEL, XML_CONS_ACC,
9		Receives <erds_receipt_1>_WITH_XML_REL_/ CC and stores ERDS evidence</erds_receipt_1>	Generates {NotificationforAcceptance} structure		XML_CONS_REJ, XML_CONT_CONS, XML_CONT_HAND).
10			Sends {NotificationforAcceptance} structure to receiving side		
11			Generates and stores XML_NOT_F_ACC ERDS evidence	N-1 entities in receiving side answer positively. One answers negatively	
12			Generates and stores XML_NOT_DEL ERDS evidence for the N entities at receiving side		

	Scenario id: 4C_RERDS_NOT_F_ACC#2	Purpose
13	Generates and stores XML_CONS_ACC ERDS evidence for N-1 entities and one XML_CONS_REJ ERDS evidence for one entity	
14	struct	
15	Successfully consigns {hndvMet, user content} structure to N-1 entities at receiving side	
16	Generates XML_CONT_CONS ERDS N-1 entities retrieve user evidence for N-1 entities content. One fails	
17	Generates XML_CONT_HAND ERDS evidence for N-1 entities	
18	Generates <erds_receipt_2>_WITH_EV_SET#1</erds_receipt_2>	
19	Sends <erds_receipt_2>_WITH_EV_SET#1 to SERDS</erds_receipt_2>	
20	Receives <erds_receipt_2>_WITH_EV_SET#1, extracts ERDS evidences and stores them</erds_receipt_2>	

Table 6b: Scenarios where RERDS uses notification for acceptance (3/6)

		Purpose			
Para	ameter: <erd_dispatch_1>_</erd_dispatch_1>				
			<pre>(ML_ACC_REJ_EXP, XML_CONT_CONS)</pre>	, XML_CONT_HAND}	
Para	ameter: {hndvMet, user conte	ent}			A negative scenario of
<er< td=""><td>DS_receipt_1>_WITH_XML</td><td>_REL_ACC</td><td></td><td></td><td><u>4C_REDS_NOT_F_ACC#1</u>:</td></er<>	DS_receipt_1>_WITH_XML	_REL_ACC			<u>4C_REDS_NOT_F_ACC#1</u> :
Para	ameter: {NotificationforAccep				Where one of the entities does
Para	ameter: <erds_receipt_2>_</erds_receipt_2>	WITH_XML_EV_SET#1	<i></i>		not respond in time to the
<u> </u>		Sequence o	f actions	_	NoficationForAcceptance. The
#	Sender	SERDS	RERDS	Receiving side	RERDS generates and stores
1	Sender sends original				both N-1
0	message	Accento submission. Conceptos and			ConsignmentAcceptance and 1
2		stores XML SUB ACC ERDS evidence			AcceptanceRejectionExpiry
з	1	Generates			evidence. RERDS generates
5		<pre><frd 1="" dispatch=""> WITH XML SUB</frd></pre>			and consigns a structure
		ACC			recipients on receiving side
4		Relies			Upon successful consignment to
		<erd_dispatch_1>_WITH_XML_SUB_</erd_dispatch_1>			N -1 entities on the receiving
		ACC to RERDS			side, RERDS generates
5			Accepts		ContentConsignment evidence.
			<pre><erd_dispatch_1>_WITH_XML_SUB</erd_dispatch_1></pre>		
<u> </u>					After the successful retrieval of
6			Generates XML_REL_ACC ERDS		the user content, RERDS
7			Generates		generates ContentHandover
ľ			<pre><frds 1="" receipt=""> WITH XML REL</frds></pre>		entities RERDS generates and
			ACC		sends back the SERD a receipt
8			Sends		message with the set of
			<erd_receipt_1>_WITH_XML_REL_</erd_receipt_1>		evidence (including
			ACC back to SERDS		XML_NOT_F_ACC,
9		Receives	Generates (NotificationforAccentance)		XML_NOT_DEL,
		<pre> <erds_receipt_1>_WITH_XML_REL_A</erds_receipt_1></pre>	structure		XML_CONS_ACC,
4.0					XML_ACC_REJ_EXP,
10			Sends (NotificationforAcceptance)		XIVIL_CONT_CONS,
11				N-1 entities in receiving	
			Generates XML_NOT_F_ACC ERDS	side answer positively. One	
			evidence	does not answer in time	
12			Generates XML_NOT_DEL ERDS		1
			evidence for the N entities at receiving		
1			side		

	Scenario id: 4C_RERDS_NO	Purpose		
13	Gen evid XML for o	nerates XML_CONS_ACC ERDS dence for N-1 entities and one IL_ACC_REJ_EXP ERDS evidence one entity		
14	Gen struc	nerates {hndvMet, user content} ucture for N-1 entities		
15	Send struct rece	nds {hndvMet, user content} ucture to N-1 accepting entities at eiving side		
16			{hndvMet, user content} structure consigned to N-1 entities in receiving side	
17	Genevid	nerates XML_CONT_CONS ERDS dence	N-1 entities retrieve user content	
18	Gen evid	nerates XML_CONT_HAND ERDS dence for N-1 entities		
19	Gen <er< td=""><td>nerates RDS_receipt_2>_WITH_EV_SET#1</td><td></td><td></td></er<>	nerates RDS_receipt_2>_WITH_EV_SET#1		
20	Send <er to SI</er 	nds RDS_receipt_2>_WITH_EV_SET#1 SERDS		
21	Receives <erds_receipt_2>_WITH_EV_SET#, extracts ERDS evidences and stores them</erds_receipt_2>			

Table 6c: Scenarios where RERDS uses notification for acceptance (4/6)

		Purpose			
Para	meter: <erd_dispatch_1></erd_dispatch_1>	A negative scenario of 4C_REDS_NOT_F_ACC#1:			
Para	meter: {hndvMet, user conf	tent}			
Para	meter: <erds_receipt_1></erds_receipt_1>	_WITH_XML_REL_ACC			Where user content is consigned
Para	meter: {NotificationforAcce	ptance}			to N-1 entities but failed to one.
Para	meter: <erds_receipt_2></erds_receipt_2>	_WITH_EV_SET#1			The RERDS generates and
	·	Sequence o	factions		stores both N-1
#	Sender	SERDS	RERDS	Receiving side	ContentConsigment and 1
	Sender sends original			g	ContentConsignmentFailure
1	message				evidence. Upon the successful
2	message	Accepts submission Generates			retrieval of the user content,
2		XML SUB ACC ERDS evidence			RERDS generates
3		Generates			ContentHandover evidence for
5		<erd_dispatch_1>_WITH_XML_SUB_ ACC</erd_dispatch_1>			N-1 receiving entities. The RERDS generates and sends back the SERD a receipt
4		Relies <erd_dispatch_1>_WITH_XML_SUB_ ACC to RERDS</erd_dispatch_1>			message with the set of evidence (including
5			Accepts <erd_dispatch_1>_WITH_XML_SUB _ACC to RERDS</erd_dispatch_1>		XML_NOT_DEL, XML_CONS_ACC, XML_CONT_CONS.
6			Generates XML_REL_ACC ERDS evidence		XML_CONT_CONS_FAIL,
7			Generates <erds_receipt_1>_WITH_XML_REL _ACC</erds_receipt_1>		
8			Sends <erd_receipt_1>_WITH_XML_REL_ ACC back to SERDS</erd_receipt_1>		
9		Receives <erds_receipt_1>_WITH_XML_REL_4 CC</erds_receipt_1>	Generates {NotificationforAcceptance} structure for N entities		
10			Sends {NotificationforAcceptance} structure to receiving side		
11			Generates XML_NOT_F_ACC ERDS evidence	All the entities in receiving side answer positively	
12			Generates and stores XML_NOT_DEL ERDS evidence for the N entities at receiving side		
13			Generates XML_CONS_ACC ERDS evidence for all the entities in receiving side		

	Scenario id: 4C_RERDS_NOT_F_ACC#4						
14		Generates {hndvMet, user content} structure for N entities					
15		Sends {hndvMet, user content} structure to N entities					
16			N-1 {hndvMet, user content} structure consignments succeed. One fails				
17		Generates XML_CONT_CONS ERDS evidence for N-1 entities and XML_CONS_FAIL for one entity	N-1 entities retrieve user content				
18		Generates XML_CONT_HAND ERDS evidence					
19		Generates <erds_receipt_2>_WITH_EV_SET#1</erds_receipt_2>					
20		Sends <erds_receipt_2>_WITH_EV_SET#1 to SERDS</erds_receipt_2>]			
21	Receives <erds_receipt_2>_WITH_EV_SET#1, and stores the ERDS evidences</erds_receipt_2>						

Table 6d: Scenarios where RERDS uses notification for acceptance (5/6)

		Purpose			
Para	meter: <erd_dispatch_1>_</erd_dispatch_1>	A negative scenario of 4C_REDS_NOT_F_ACC#1:			
Para	meter: {hndvMet, user conte	ent}			
Para	meter: <erds_receipt_1>_</erds_receipt_1>	WITH_XML_REL_ACC			Where RERDS generates N-1
Para	meter: {NotificationforAccep	otance}			ContentHandover and 1
Para	meter: <erds_receipt_2>_</erds_receipt_2>	WITH_EV_SET#1			ContentHandoverFailure
		Sequence o	factions		evidence for receiving entities
#	Sender	SERDS	RERDS	Receiving side	when N-1 entitles retrieve the
4	Sender sends original				fails RERDS dependences and
1	message				sends back the SERD a receipt
2		Accepts submission. Generates and			message with the set of
		stores XML_SUB_ACC ERDS evidence			evidence (including
3		Generates <erd_dispatch_1>_WITH_XML_SUB_ ACC</erd_dispatch_1>			XML_NOT_F_ACC, XML_NOT_DEL, XML_CONS_ACC
4		Relies <erd_dispatch_1>_WITH_XML_SUB_ ACC to RERDS</erd_dispatch_1>			XML_CONT_CONS, XML_CONS_FAIL, XML_CONT_HAND
5			Accepts <erd_dispatch_1>_WITH_XML_SUB _ACC to RERDS</erd_dispatch_1>		XML_CONT_HAND_FAIL).
6			Generates XML_REL_ACC ERDS evidence		
7			Generates <erds_receipt_1>_WITH_XML_REL _ACC</erds_receipt_1>		
8			Sends <erd_receipt_1>_WITH_XML_REL_ ACC back to SERDS</erd_receipt_1>		
9		Receives <erds_receipt_1>_WITH_XML_REL_4 CC</erds_receipt_1>	Generates {NotificationforAcceptance} structure		
10			Sends {NotificationforAcceptance} structure to receiving side		
11			Generates XML_NOT_F_ACC ERDS evidence	All the entities in receiving side answer positively	
12			Generates and stores XML_NOT_DEL ERDS evidence for the N entities at receiving side		
13			Generates XML_CONS_ACC ERDS evidence for all the entities in receiving side		

	Scenario id: 4C_RERDS_NOT_F_ACC#5	Purpose
14	Generates {hndvMet, user content} for N entities	
15	Sends {hndvMet, user content} to N-1 accepting entities at receiving side	
16	N consignments of {hndvMet, user content} structure succeed	
17	Generates XML_CONT_CONS ERDS N-1 entities retrieve user content. One fails	
18	Generates XML_CONT_HAND ERDS evidence for N-1 entities and one XML_CONT_HAND_FAIL for one entity	
19	Generates <erds_receipt_2>_WITH_EV_SET#1</erds_receipt_2>	
20	Sends <erds_receipt_2>_WITH_EV_SET#1 to SERDS</erds_receipt_2>	
21	Receives <erds_receipt_2>_WITH_EV_SET#1 and stores ErDS evidences</erds_receipt_2>	

Table 6e: Scenarios where RERDS uses notification for acceptance (6/6)

		Purpose			
Para	ameter: <erd_dispatch_1>_</erd_dispatch_1>	_WITH_XML_SUB_ACC	/ar EV_SET#1 = {XML_NOT_F_ACC, XM KML_CONS_ACC, XML_CONT_CONS, X KML_CONT_AC_TRACK}	L_NOT_DEL, ML_CONT_CONS_FAIL,	A negative scenario of 4C_REDS_NOT_F_ACC#1:
Para	ameter: {hndvMet, user cont	ent}			
Para	ameter: <erds_receipt_1>_</erds_receipt_1>	_WITH_XML_REL_ACC			Where user content is consigned
Para	ameter: {NotificationforAccept	ptance}			to N-1 entities but failed to one.
Para	ameter: <erds_receipt_2>_</erds_receipt_2>	WITH_EV_SET#1			The RERDS generates and
	· · · · · ·	Sequence o	f actions		stores both N-1
#	Sender	SERDS	RERDS	Receiving side	ContentConsigment and 1
	Sender sends original				
1	message				evidence. Upon the successful
2	mooodgo	Accepts submission, Generates			retrieval of the user content,
_		XML_SUB_ACC_ERDS evidence			RERDS generates
3		Generates			
-		<erd 1="" dispatch=""> WITH XML SUB</erd>			entities The RERDS generates
		ACC			and sends back the SERD a
4		Relies			receipt message with the set of
		<erd_dispatch_1>_WITH_XML_SUB_ ACC to RERDS</erd_dispatch_1>			evidence (including
5			Accepts		XML_NOT_L_ACC
			<erd_dispatch_1>_WITH_XML_SUB</erd_dispatch_1>		XML_CONT_CONS.
			_ACC to RERDS		XML CONT CONS FAIL
6			Generates XML_REL_ACC ERDS		XML CONT AC TRACK).
			evidence		/
7			Generates		Here, the scenario includes
			<erds_receipt_1>_WITH_XML_REL</erds_receipt_1>		ContentAccessTracking event
			_ACC		instead of the handover, as both
8			Sends		events provide evidence of
			<erd_receipt_1>_WITH_XML_REL_</erd_receipt_1>		the successful delivery of the
			ACC back to SERDS		user content.
9		Receives <erds_receipt_1>_WITH_XML_REL_4 CC</erds_receipt_1>	Generates {NotificationforAcceptance} structure for N entities		
10			Sends {NotificationforAcceptance}		-
			structure to receiving side		
11			Generates XML NOT F ACC ERDS	All the entities in receiving	-
			evidence	side answer positively	
12			Generates and stores XML NOT DEL		
			ERDS evidence for the N entities at		
1			receiving side		
13			Generates XML_CONS_ACC ERDS		
			evidence for all the entities in receiving		
			side		

	Scenario id: 4C_RERDS_NOT_F_ACC#6	Purpose
14	Generates {hndvMet, user content} structure for N entities	
15	Sends {hndvMet, user content} structure to N entities	
16	N-1 {hndvMet, user content} structure consignments succeed. One fails	
17	Generates XML_CONT_CONS ERDS evidence for N-1 entities and XML_CONS_FAIL for one entity	
18	Generates XML_CONT_AC_TRACK ERDS evidence	
19	Generates <erds_receipt_2>_WITH_EV_SET#1</erds_receipt_2>	
20	Sends <erds_receipt_2>_WITH_EV_SET#1 to SERDS</erds_receipt_2>	
21	Receives <erds_receipt_2>_WITH_EV_SET#1, and stores the ERDS evidences</erds_receipt_2>	

5.5 Scenarios for extended model

5.5.1 Introduction

The present clause defines test cases for scenarios that take place when the sender and the entities at the receiving side are subscribed to different ERDSs and there is one intermediate IERDS between the SERDS and the RERDSs.

57

Clause 5.5.2 defines test cases where the RERDS does not use notification for acceptance.

Clause 5.5.3 defines test cases where the RERDS uses notification for acceptance.

As with previous scenarios defined in the present document, new sets of scenarios can be obtained by replacing ERD dispatches with ERD payloads in the scenarios shown below.

5.5.2 Scenarios where RERDS does not use notification for acceptance

Table 7 shows scenarios where RERDS does not use notification for acceptance.

Table 7: Scenarios where SERDS operates Store&Notify a	and RERDS operate Store&Forward (1/6)
--	---------------------------------------

		Scena	rio id: EXT_RERDS_NO_NOT_	F_ACC#1		Purpose
Para	meter: <erd_dispatch_< td=""><td>1>_WITH_XML_SUB_ACC</td><td></td><td></td><td></td><td>•</td></erd_dispatch_<>	1>_WITH_XML_SUB_ACC				•
Para	meter: <erds_receipt_1< td=""><td>The simplest positive scenario:</td></erds_receipt_1<>	The simplest positive scenario:				
Para	meter: <erds_receipt_2< td=""><td></td><td></td></erds_receipt_2<>					
Para	meter: {hndvMet, user co		The SERDS accepts the			
Para	meter: < ERDS receipt 3	3> WITH XML CONT CONS				submission of the original
Para	meter: <erds 4<="" receipt="" td=""><td>4> WITH XML CONT HAND</td><td></td><td></td><td></td><td>message, then generates and</td></erds>	4> WITH XML CONT HAND				message, then generates and
	<u></u>	<u> </u>	Sequence of actions			relies IERDS a dispatch
#	Sender	SFRDS	IERDS	RERDS	Receiving side	message. IERDS accepts the
	Sender sends original					relay message and sends back
1	message					to SERDS a receipt message
2	message	Accepts submission and				containing RelayAcceptance
2		denerates and stores				evidence. Afterwards, IERDS
						generates a dispatch message
		evidence				
3		Generates				
Ŭ		<pre><frd 1="" dispatch=""> WITH</frd></pre>				to RERDS. RERDS accepts the
		XMI SUB ACC				relay message and sends back
4		Relies				to IERDS a receipt message
–		<pre>>ERD dispatch 1> WITH</pre>				
		XMI_SUB_ACC to IERDS				evidence.
5			Receives			Afterwards RERDS generates
Ŭ			<pre><frd 1="" dispatch=""> WITH X</frd></pre>			and consigns a structure
			MI_SUB_ACC			containing user content to N
6			Generates and stores			recipients on receiving side After
Ŭ			XMI REL ACC ERDS			being consigned successfully to
			evidence			N entities on the receiving side
7			Generates			RERDS generates and sends
ľ			<pre><frds 1="" receipt=""> WITH X</frds></pre>			hack the receipt containing
			ML REL ACC			ContentConsignment evidence to
8			Sends			the previous FRDS. This
Ŭ			<pre><frds 1="" receipt=""> WITH X</frds></pre>			Previous ERDS performs similar
			MI REL ACC back to			action till this message is sent
			SERDS			back to the sender
9		Receives				
Ĩ		<pre><frds 1="" receipt=""> WITH</frds></pre>	Relies			The cycle of message events in
		XMI_REL_ACC and stores	<erd_dispatch_1>_WITH_X</erd_dispatch_1>			this scenario is completed with
		ERDS evidence	ML_SUB_ACC to RERDS			the generation of a receipt
10				Receives		message containing
				<erd 1="" dispatch=""> WITH XM</erd>		ContentHandover ERDS
				L SUB ACC to RERDS		evidence.
11				Generates XML_REL_ACC		
· ·				ERDS evidence		

		Scena	rio id: EXT_RERDS_NO_NOT_	F_ACC#1		Purpose
12				Generates <erds_receipt_2>_WITH_XM L_REL_ACC</erds_receipt_2>		
13				Sends <erds_receipt_2>_WITH_XM L_REL_ACC back to IERDS</erds_receipt_2>		
14			Receives <erds_receipt_2>_WITH_X ML_REL_ACC</erds_receipt_2>	Generates {hndvMet, user content} structure		
15				Consigns {hndvMet, user content} structure to receiving side		
16					N consignments of {hndvMet, user content} structure succeed	
17				Generates XML_CONT_CONS ERDS evidence for N entities		
18				Generates <erds_receipt_3>_WITH_XM L_CONT_CONS</erds_receipt_3>	All the entities in receiving side successfully retrieve {hndvMet, user content} structure	
19				Sends <erds_receipt_3>_WITH_XM L_CONT_CONS back to IERDS</erds_receipt_3>		
20			Receives <erds_receipt_3>_WITH_X ML_CONT_CONS</erds_receipt_3>	Generates XML_CONT_HAND for N entities		
21			Sends <erds_receipt_3>_WITH_X ML_CONT_CONS back to SERDS</erds_receipt_3>	Generates <erds_receipt_4>_WITH_XM L_CONT_HAND</erds_receipt_4>		
22		Receives <erds_receipt_3>_WITH_ XML_CONT_CONS and stores ERDS evidences</erds_receipt_3>		Sends <erds_receipt_4>_WITH_XM L_CONT_HAND back to IERDS</erds_receipt_4>		
23		Sends <erds_receipt_3>_WITH_ XML_CONT_CONS back to sender</erds_receipt_3>	Receives <erds_receipt_4>_WITH_X ML_CONT_HAND</erds_receipt_4>			
24	Receives <erds_receipt_3>_W ITH_XML_CONT_CO</erds_receipt_3>		Sends <erds_receipt_4>_WITH_X ML_CONT_HAND back to SERDS</erds_receipt_4>			

ETSI TS 119 524-2 V1.2.1 (2023-12)

	Scenario id: EXT_RERDS_NO_NOT_F_ACC#1					
25	Receives <erds_receipt_4>_WITH_ XML_CONT_HAND and stores ERDS evidences</erds_receipt_4>					

Table 7a: Scenarios where SERDS operates Store&Notify and RERDSs operate Store&Forward (2/6)

	Scenario id: EXT_RERDS_NO_NOT_F_ACC#2						Purpose
Para	meter: <erd_dispatch_1< td=""><td>I>_WITH_XML_SUB_ACC</td><td></td><td>Var SET_EV#</td><td><pre>#1 = {XML_CONT_CONS, XML_C</pre></td><td>ONT_CONS_FAIL}</td><td>· ·</td></erd_dispatch_1<>	I>_WITH_XML_SUB_ACC		Var SET_EV#	<pre>#1 = {XML_CONT_CONS, XML_C</pre>	ONT_CONS_FAIL}	· ·
Para	meter: <erds_receipt_1< td=""><td>>_WITH_XML_REL_ACC</td><td></td><td></td><td>•</td><td></td><td>A negative scenario of</td></erds_receipt_1<>	>_WITH_XML_REL_ACC			•		A negative scenario of
Para	meter: <erds_receipt_2< td=""><td>2>_WITH_XML_REL_ACC</td><td></td><td></td><td></td><td></td><td>EXT_RERDS_NO_NO</td></erds_receipt_2<>	2>_WITH_XML_REL_ACC					EXT_RERDS_NO_NO
Para	meter: {hndvMet, user co	ontent}					<u>T_F_ACC#1</u> :
Para	meter: <erds_receipt_3< td=""><td>3>_WITH_EV_SET#1</td><td></td><td></td><td></td><td></td><td></td></erds_receipt_3<>	3>_WITH_EV_SET#1					
Para	meter: <erds_receipt_4< td=""><td>I>_WITH_XML_CONT_HAND</td><td></td><td></td><td></td><td></td><td>When the user content</td></erds_receipt_4<>	I>_WITH_XML_CONT_HAND					When the user content
			Sequence	of actions			is consigned
#	Sender	SERDS	IERD	S	RERDS	Receiving side	entities but failed for 1
1	Sender sends original message						entity on the receiving
	Ŭ	Accepts submission and					deperates and sends
2		generates and stores					back to IERD a receipt
2		XML_SUB_ACC ERDS					message with the set
		evidence					of evidence (including
~		Generates					XML_CONT_CONS,
3		XML_SUB_ACC					XML_CONT_CONS_F AIL).
		Relies					,
4		<erd_dispatch_1>_WITH_</erd_dispatch_1>					After the user content
		XML_SUB_ACC to IERDS					is retrieved
F			Keceives	4. MUTH V			successfully by N-1
5				_I>_VVIIIX			entities on the
			Generates XMI	REL ACC			
6			ERDS evidence	and stores			events in this scenario
			Generates				is completed with the
7			<erds_receipt< td=""><td>1>_WITH_X</td><td></td><td></td><td>generation of a receipt</td></erds_receipt<>	1>_WITH_X			generation of a receipt
			ML_REL_ACC				message containing
			Sends				ContentHandover
8			<erds_receipt_< td=""><td>1>_WITH_X</td><td></td><td></td><td>ERDS evidence.</td></erds_receipt_<>	1>_WITH_X			ERDS evidence.
•			ML_REL_ACC b	ack to			
		Baaaiyaa	SERDS				
		<pre>CERDS receipt 15 W/ITH</pre>	Relies				
9		XMI_REL_ACC and stores	<erd_dispatch_< td=""><td>1>_WITH_X</td><td></td><td></td><td></td></erd_dispatch_<>	1>_WITH_X			
		ERDS evidence	ML_SUB_ACC to	RERDS			
					Receives		
10					<erd_dispatch_1>_WITH_XM</erd_dispatch_1>		
					L_SUB_ACC		
1					Generates and stores		
11					XML_REL_ACC ERDS		
	1				evidence		

	Scenario id: EXT_RERDS_NO_NOT_F_ACC#2						
12			Generates <erds_receipt_2>_WITH_XM L_REL_ACC</erds_receipt_2>				
13			Sends <erds_receipt_2>_WITH_XM L_REL_ACC back to IERDS</erds_receipt_2>				
14		Receives <erds_receipt_2>_WITH_X ML_REL_AC and stores ERDS evidence</erds_receipt_2>	Generates {hndvMet, user content} structure for N entities				
15			Consigns {hndvMet, user content} to receiving side				
16				N-1 {hndvMet, user content} structure consignments succeed. One fails			
17			Generates and stores XML_CONT_CONS ERDS evidence for N-1 entities and XML_CONT_CONS_FAIL ERDS evidence for one entity				
18			Generates <erds_receipt_3>_WITH_EV_ SET#1</erds_receipt_3>	N-1 entities in receiving side retrieve {hndvMet, user content} structure			
19			Sends <erds_receipt_3>_WITH_EV_ SET#1 back to IERDS</erds_receipt_3>				
20		Receives <erds_receipt_3>_WITH_E V_SET#1</erds_receipt_3>	Generates and stores XML_CONT_HAND for N-1 entities				
21		Sends <erds_receipt_3>_WITH_E V_SET#1 back to SERDS</erds_receipt_3>	Generates <erds_receipt_4>_WITH_XM L_CONT_HAND</erds_receipt_4>				
22	Receives <erds_receipt_3>_WITH_ EV_SET#1 and stores ERDS evidences</erds_receipt_3>		Sends <erds_receipt_4>_WITH_XM L_CONT_HAND back to IERDS</erds_receipt_4>				
23		Receives <erds_receipt_4>_WITH_X ML_CONT_HAND</erds_receipt_4>					
24		Sends <erds_receipt_4>_WITH_X ML_CONT_HAND back to SERDS</erds_receipt_4>					

ETSI TS 119 524-2 V1.2.1 (2023-12)

	Scenario id: EXT_RERDS_NO_NOT_F_ACC#2					
25	Receives <erds_receipt_4>_WITH_ XML_CONT_HAND and stores ERDS evidences</erds_receipt_4>					

Table 7b: Scenarios where SERDS operates Store&Notify and RERDSs operate Store&Forward (3/7)

		Scenar	rio id: EXT_RERD	S_NO_NOT_	F_ACC #3		Purpose
Para	Parameter: <erd_dispatch_1>_WITH_XML_SUB_ACC Var EV_SET#1 = {XML_CONT_HAND, XML_CONT_HAND_FAIL}</erd_dispatch_1>						
Para	meter: <erds_receipt_1< td=""><td>>_WITH_XML_REL_ACC</td><td></td><td></td><td>· · · · · ·</td><td></td><td>A negative scenario of</td></erds_receipt_1<>	>_WITH_XML_REL_ACC			· · · · · ·		A negative scenario of
Para	meter: <erds_receipt_2< td=""><td>2>_WITH_XML_REL_ACC</td><td></td><td></td><td></td><td></td><td>EXT_RERDS_NO_NO</td></erds_receipt_2<>	2>_WITH_XML_REL_ACC					EXT_RERDS_NO_NO
Para	meter: {hndvMet, user co	ontent}					<u>T_F_ACC#1</u> :
Para	meter: <erds_receipt_3< td=""><td>3>_WITH_XML_CONT_CONS</td><td></td><td></td><td></td><td></td><td></td></erds_receipt_3<>	3>_WITH_XML_CONT_CONS					
Para	meter: <erds_receipt_4< td=""><td>I>_WITH_EV_SET#1</td><td></td><td></td><td></td><td></td><td>When N-1 entities</td></erds_receipt_4<>	I>_WITH_EV_SET#1					When N-1 entities
			Sequence	of actions			retrieve the user
#	Sender	SERDS	IERD	S	RERDS	Receiving side	to retrieve the RERDS
1	Sender sends original message						generates and sends a
		Accepts submission and					to IERDS containing a
2		generates and stores					set of evidence
2		XML_SUB_ACC ERDS evidence					(including
		Generates					XML CONT HAND F
3		<erd_dispatch_1>_WITH_ XML_SUB_ACC</erd_dispatch_1>					AIL).
		Relies					
4		<erd_dispatch_1>_WITH_ XML_SUB_ACC to IERDS</erd_dispatch_1>					
			Receives				
5			<erd_dispatch_< td=""><td>1>_WITH_X</td><td></td><td></td><td></td></erd_dispatch_<>	1>_WITH_X			
			ML_SUB_ACC				
			Generates and st	tores			
6			XML_REL_ACC	ERDS			
			Concretes				
7				15 WITH X			
<u></u>			ML_REL_ACC				
			Sends				
8				$1>_VVIII_A$			
			SERDS				
			Relies				
9		<erds_receipt_1>_VVIIH_</erds_receipt_1>	<erd_dispatch_< td=""><td>1>_WITH_X</td><td></td><td></td><td></td></erd_dispatch_<>	1>_WITH_X			
		ERDS evidence	ML_SUB_ACC to	RERDS			
4.0					Receives		
10					<pre> <ekd_dispatch_1>_WITH_XM</ekd_dispatch_1></pre>		
					L_SUB_ACC to RERDS		
11							
''					evidence		

	Scenario id: EXT_RERDS_NO_NOT_F_ACC #3					
12				Generates <erds_receipt_2>_WITH_XM L_REL_ACC</erds_receipt_2>		
13				Sends <erds_receipt_2>_WITH_XM L_REL_ACC back to IERDS</erds_receipt_2>		
14			Receives <erds_receipt_2>_WITH_X ML_REL_ACC and stores ERDS evidence</erds_receipt_2>	Generates {hndvMet, user content} structure		
15				Consigns {hndvMet, user content} to receiving side		
16					N {hndvMet, user content} structure consignments succeed	
17				Generates and stores XML_CONT_CONS ERDS evidence for N entities		
18				Generates <erds_receipt_3>_WITH_XM L_CONT_CONS</erds_receipt_3>	N-1 entities in receiving side successfully retrieve {hndvMet, user content} structure. One entity fails	
19				Sends <erds_receipt_3>_WITH_XM L_CONT_CONS back to IERDS</erds_receipt_3>		
20			Receives <erds_receipt_3>_WITH_X ML_CONT_CONS</erds_receipt_3>	Generates XML_CONT_HAND ERDS evidence for N-1 entities and XML_CONT_HAND_FAIL for one entity		
21			Sends <erds_receipt_3>_WITH_X ML_CONT_CONS back to SERDS</erds_receipt_3>	Generates <erds_receipt_4>_WITH_EV_ SET#1</erds_receipt_4>		
22		Receives <erds_receipt_3>_WITH_ XML_CONT_CONS and stores ERDS evidences</erds_receipt_3>		Sends <erds_receipt_4>_WITH_EV_ SET#1 back to IERDS</erds_receipt_4>		
23			Receives <erds_receipt_4>_WITH_E V_SET#1</erds_receipt_4>			
24			Sends <erds_receipt_4>_WITH_E V_SET#1 back to SERDS</erds_receipt_4>			

ETSI TS 119 524-2 V1.2.1 (2023-12)

	Scenario id: EXT_RERDS_NO_NOT_F_ACC #3				
25	Receives <erds_receipt_4>_WITH_ EV_SET#1 and stores ERDS evidences</erds_receipt_4>				

Table 7c: Scenarios where SERDS operates Store&Notify and RERDSs operate Store&Forward (4/7)

		Scena	rio id: EXT_RERDS_NO_NOT_	F_ACC #4		Purpose
Para	ameter: <erd_dispatch_< td=""><td>1>_WITH_XML_SUB_ACC</td><td></td><td></td><td></td><td></td></erd_dispatch_<>	1>_WITH_XML_SUB_ACC				
Para	ameter: <erds_receipt_< td=""><td>1>_WITH_XML_REL_REJ</td><td></td><td></td><td></td><td>A negative scenario of</td></erds_receipt_<>	1>_WITH_XML_REL_REJ				A negative scenario of
Para	ameter: {hndvMet,XML_R	EXT_RERDS_NO_NOT_F_A				
			Sequence of actions			<u>CC#1</u> :
#	Sender	SERDS	IERDS	RERDS	Receiving side	When the SERDS accepts
1	Sender sends original					the submission, it generates
-	message					and relies IERDS on an ERD
		Accepts submission and				dispatch message.
2		XML SUB ACC ERDS				Afterwards, IERDS rejects
		evidence				the ERD dispatch message
		Generates				a receipt message containing
3		<erd_dispatch_1>_WITH_</erd_dispatch_1>				RelayRejection evidence.
		XML_SUB_ACC				The cycle of message events
4		Relies				in this scenario is completed
4		XMI_SUB_ACC to IFRDS				with SERDS generates and
5			Rejects relying			sends the structure
			Generates and stores			metadata along with
6			XML_REL_REJ ERDS			RelayRejection evidence
			evidence			back to the sender.
7			Generates			
ľ						
			Sends			
~			<erds 1="" receipt=""> WITH X</erds>			
8			ML_REL_REJ back to			
			SERDS			
		Receives				
9		<pre><erds_receipt_1>_WITH_ XML_BEL_BEL and stores</erds_receipt_1></pre>				
		FRDS evidence				
		Generates				
10		{hndvMet,XML_REL_REJ}				
		structure				
		Sends				
11		{hndvMet,XML_REL_REJ}				
	Pacaivas	structure back to sender				
12	{hndvMet.XML_RFI					
·-	REJ} structure					

Table 7d: Scenarios where SERDS operates Store&Notify and RERDSs operate Store&Forward (5/7)

		Scena	rio id: EXT_RERDS_NO_NOT_	F_ACC #5		Purpose
Para	ameter: <erd_dispatch_< td=""><td>1>_WITH_XML_SUB_ACC</td><td></td><td></td><td></td><td></td></erd_dispatch_<>	1>_WITH_XML_SUB_ACC				
Para	ameter: <erds_receipt_< td=""><td>I>_WITH_XML_REL_ACC</td><td></td><td></td><td></td><td>A negative scenario of</td></erds_receipt_<>	I>_WITH_XML_REL_ACC				A negative scenario of
Para	ameter: <erds_receipt_2< td=""><td>2>_WITH_XML_REL_REJ</td><td></td><td></td><td></td><td>EXT_RERDS_NO_NOT_F</td></erds_receipt_2<>	2>_WITH_XML_REL_REJ				EXT_RERDS_NO_NOT_F
		<u>_ACC#1</u> :				
#	Sender	SERDS	IERDS	RERDS	Receiving side	The SERDS accepts the
1	Sender sends original					submission of the original
	message					message, It generates and
2		Accepts submission and				relies IERDS a dispatch
2		ERDS evidence				message. IERDS accepts
		Generates				the relay message and
3		<erd_dispatch_1>_WITH_</erd_dispatch_1>				sends back to SERDS a
		XML_SUB_ACC				RelavAcceptance evidence
		Relies				Afterwards, IERDS
4		<pre><erd_dispatch_1>_WITH_</erd_dispatch_1></pre>				generates a dispatch
		XML_SUB_ACC to IERDS	Deseives			message along with
5			Receives			SubmissionAcceptance
5			MI_SUB_ACC			evidence to RERDS.
-			Generates XML REL ACC			RERDS rejects the relay
6			ERDS evidence			to IERDS a receipt
			Generates			message containing
7			<erds_receipt_1>_WITH_X</erds_receipt_1>			RelayRejection evidence.
			ML_REL_ACC			
			Sends			
8			<erds_receipt_1>_VITH_X</erds_receipt_1>			
			SERDS			
		Receives	Relies			
9		<erds_receipt_1>_WITH_</erds_receipt_1>	<erd_dispatch_1>_WITH_X</erd_dispatch_1>			
		XML_REL_ACC	ML_SUB_ACC to RERDS			
				Receives		
10				<erd_dispatch_1>_WITH_XM</erd_dispatch_1>		
				L_SUB_ACC but rejects it		
11				Generates XML_REL_REJ		
		1		Generates		
12				<erds 2="" receipt=""> WITH XM</erds>		
				L_REL_REJ		
				Sends		
13				<erds_receipt_2>_WITH_XM</erds_receipt_2>		
1				L_REL_REJ back to IERDS		

Scenario id: EXT_RERDS_NO_NOT_F_ACC #5							
14			Receives <erds_receipt_2>_WITH_X ML_REL_REJ</erds_receipt_2>				
15			Sends <erds_receipt_2>_WITH_X ML_REL_REJ back to SERDS</erds_receipt_2>				
16		Receives <erds_receipt_2>_WITH_ XML_REL_REJ</erds_receipt_2>					
17		Sends <erds_receipt_2>_WITH_ XML_REL_REJ back to sender</erds_receipt_2>					
18	Receives <erds_receipt_2>_W ITH_XML_REL_REJ</erds_receipt_2>						

Table 7e: Scenarios where SERDS operates Store&Notify and RERDS operate Store&Forward (6/6)

		Scena	rio id: EXT_RERDS_NO_NOT_	F_ACC#6		Purpose
Para	meter: <erd_dispatch_< td=""><td>1>_WITH_XML_SUB_ACC</td><td></td><td></td><td></td><td></td></erd_dispatch_<>	1>_WITH_XML_SUB_ACC				
Para	meter: <erds_receipt_< td=""><td>1>_WITH_XML_REL_ACC</td><td></td><td></td><td></td><td>The simplest positive scenario:</td></erds_receipt_<>	1>_WITH_XML_REL_ACC				The simplest positive scenario:
Para	ameter: <erds_receipt_2< td=""><td>2>_WITH_XML_REL_ACC</td><td></td><td></td><td></td><td></td></erds_receipt_2<>	2>_WITH_XML_REL_ACC				
Para	ameter: {hndvMet, user co		The SERDS accepts the			
Para	ameter: <erds_receipt_3< td=""><td></td><td>submission of the original</td></erds_receipt_3<>		submission of the original			
Para	ameter: <erds_receipt_4< td=""><td>4>_WITH_XML_CONT_AC_TF</td><td>RACK</td><td></td><td></td><td>message, then generates and</td></erds_receipt_4<>	4>_WITH_XML_CONT_AC_TF	RACK			message, then generates and
			Sequence of actions			relies IERDS a dispatch
#	Sender	SERDS	IERDS	RERDS	Receiving side	message.
1	Sender sends original					IERDS accepts the relay
1	message					message and sends back to
2		Accepts submission and				SERDS a receipt message
		generates and stores				containing RelayAcceptance
		XML_SUB_ACC ERDS				evidence.
		evidence				
3		Generates				Afterwards, IERDS generates a
		<pre><erd_dispatch_1>_WITH_</erd_dispatch_1></pre>				dispatch message along with
		XML_SUB_ACC				SubmissionAcceptance evidence
4		Relies				to RERDS.
		<pre><erd_dispatch_1>_WITH_</erd_dispatch_1></pre>				
_		XML_SUB_ACC to IERDS	.			RERDS accepts the relay
5			Receives			message and sends back to
			<erd_dispatch_1>_WITH_X</erd_dispatch_1>			IERDS a receipt message
<u> </u>			ML_SUB_ACC			
0						evidence.
			AWL_REL_ACC ERDS			Attenuerde DEDDC renerates
7			Concretes			Alterwards, RERDS generates
'			<pre>CERDS receipt 1> WITH Y</pre>			containing user content to N
						recipients on receiving side After
8			Sends	l		being consigned successfully to
Ŭ			<pre><frds 1="" receipt=""> WITH X</frds></pre>			N entities on the receiving side
			MI REL ACC back to			RERDS generates and sends
			SERDS			back the receipt containing
9		Receives				ContentConsignment evidence to
ľ		<pre><erds 1="" receipt=""> WITH</erds></pre>	Relies			the previous ERDS.
		XML REL ACC and stores	<pre><erd_dispatch_1>_WITH_X</erd_dispatch_1></pre>			
		ERDS evidence	ML_SUB_ACC to RERDS			This Previous ERDS performs
10				Receives		similar action till this message is
				<erd_dispatch 1=""> WITH XM</erd_dispatch>		sent back to the sender.
				L_SUB_ACC to RERDS		
11				Generates XML_REL_ACC		The cycle of message events in
				ERDS evidence		this scenario is completed with

		Scena	rio id: EXT_RERDS_NO_NOT_	F_ACC#6		Purpose
12				Generates <erds_receipt_2>_WITH_XM L_REL_ACC</erds_receipt_2>		the generation of a receipt message containing ContentAccessTracking ERDS
13				Sends <erds_receipt_2>_WITH_XM L_REL_ACC back to IERDS</erds_receipt_2>		evidence.
14			Receives <erds_receipt_2>_WITH_X ML_REL_ACC</erds_receipt_2>	Generates {hndvMet, user content} structure		
15				Consigns {hndvMet, user content} structure to receiving side		
16					N consignments of {hndvMet, user content} structure succeed	
17				Generates XML_CONT_CONS ERDS evidence for N entities		
18				Generates <erds_receipt_3>_WITH_XM L_CONT_AC_TRACK</erds_receipt_3>	All the entities in receiving side successfully retrieve {hndvMet, user content} structure	
19				Sends <erds_receipt_3>_WITH_XM L_CONT_CONS back to IERDS</erds_receipt_3>		
20			Receives <erds_receipt_3>_WITH_X ML_CONT_CONS</erds_receipt_3>	Generates XML_CONT_AC_TRACK for N entities		
21			Sends <erds_receipt_3>_WITH_X ML_CONT_CONS back to SERDS</erds_receipt_3>	Generates <erds_receipt_4>_WITH_XM L_CONT_AC_TRACK</erds_receipt_4>		
22		Receives <erds_receipt_3>_WITH_ XML_CONT_CONS and stores ERDS evidences</erds_receipt_3>		Sends <erds_receipt_4>_WITH_XM L_CONT_AC_TRACK back to IERDS</erds_receipt_4>		
23		Sends <erds_receipt_3>_WITH_ XML_CONT_CONS back to sender</erds_receipt_3>	Receives <erds_receipt_4>_WITH_X ML_CONT_AC_TRACK</erds_receipt_4>			
24	Receives <erds_receipt_3>_W ITH_XML_CONT_CO NS</erds_receipt_3>		Sends <erds_receipt_4>_WITH_X ML_CONT_AC_TRACK back to SERDS</erds_receipt_4>			

ETSI TS 119 524-2 V1.2.1 (2023-12)

	Scenario id: EXT_RERDS_NO_NOT_F_ACC#6					
25	Receives <erds_receipt_4>_WITH_ XML_CONT_AC_TRACK</erds_receipt_4>					
	and stores ERDS evidences					
5.5.3 Scenarios where RERDS uses notification for acceptance

Table 8 shows scenarios where RERDS submits notification for acceptance.

Table 8: Scenarios where SERDS operates Store&Notify and RERDSs operate Store&Forward (1/3)

Parameter: <erd_inspatch_1>_WITH_XML_SUB_ACC Var EV_SET#1 = (XML_NOT_F_ACC, XML_NOT_DEL) The simplest positive scenario: Parameter: <erds_receipt_1>_WITH_XML_REL_ACC Where the SERDS accepts the submission of the original message. Where the SERDS accepts the submission and generates and stores and stores</erds_receipt_1></erd_inspatch_1>			Scenario id: SER	DS_SN_IERDS_SF	_RERDS_B	B_NO_NOT_F_ACC#1			Purpose
Parameter: The simplest positive scenario: Parameter: (NotificationforAcceptance) Parameter: CRDS_receipt_2>_WITH_XML_REL_ACC Where the SERDS accepts the submission of the original message Parameter: CRDS_receipt_2>_WITH_XML_CONS_ACC it generates and relies IERDS a dispatch message. Parameter: CRDS_receipt_2>_WITH_XML_CONS_ACC it generates and relies IERDS accepts the submission of the original message. Parameter: (Ind/Met, user content) dispatch message. # Sender SERDS IERDS RERDS receipt as dispatch message and sends back to SERDS a dispatch message and sends back to SERDS. 1 Sender sends original message generates and stores dispatch message also dispatch message also dispatch message also dispatch message and sends back to SERDS. RERDS receipt message 2 Accepts submission and generates and stores XML_SUB_ACC ERDS accepts the relay message and sends back to SERDS. with SubmissionAcceptance evidence to RERDS. RERDS acceptance evidence to RERDS. RERDS accepts the relay message and sends back the relay message and sends back the relay message and sends back the relay message als dispatch me	Para	ameter: <erd_dispatch_< td=""><td>1>_WITH_XML_SUB_ACC</td><td></td><td>Var EV_SET</td><td>#1 = {XML_NOT_F_ACC</td><td>C, XML_N</td><td>OT_DEL}</td><td></td></erd_dispatch_<>	1>_WITH_XML_SUB_ACC		Var EV_SET	#1 = {XML_NOT_F_ACC	C, XML_N	OT_DEL}	
Parameter: {NotificationforAcceptance} Where the SERDS accepts the submission of the original message. Parameter: <erds_receipt_3>_WITH_EV_SET#1 submission of the original message. Parameter: <erds_receipt_4>_WITH_EV_SET#1 dispatch message. IERDS a dispatch message. IERDS a dispatch message. Parameter: <indvmet, content)<="" td="" user=""> dispatch message. # Sender SERDS I Sender sends original message containing RelayAcceptance evidence. Afterwards, IERDS generates and stores with Submission and generates and stores with Submission and generates and stores with CERDS message. with Submission Acceptance evidence to RERDS in the relay message and stores sends back RelayAcceptance to RERDS in the receipt with CERDS in the receipt message. 3 Generates (ERD Signatch_1>_WITH_SUB_ACC to IERDS) The RERDS generates and sends stores sends back RelayAcceptance to RERDS in the receipt message. 4 Relies <erd_dispatch_1>_WITH_ XML_SUB_ACC to IERDS Receives <erd_dispatch_1>_WITH_X the relay message.</erd_dispatch_1></erd_dispatch_1></indvmet,></erds_receipt_4></erds_receipt_3>	Para	ameter: <erds_receipt_< td=""><td>1>_WITH_XML_REL_ACC</td><td></td><td></td><td></td><td></td><td></td><td>The simplest positive scenario:</td></erds_receipt_<>	1>_WITH_XML_REL_ACC						The simplest positive scenario:
Parameter: ZERDS_receipt_2>_WITH_XML_REL_ACC Where the SERDS accepts the submission of the original message submission of the original message and relies terms Parameter: Kereipt_3>_WITH_XML_CONS_ACC it generates and relies terms Parameter: Kereipt_4>_WITH_XML_CONS_ACC it generates and relies terms Parameter: Kereipt_4>_WITH_XML_CONS_ACC it generates and relies terms dispatch message. IERDS accept terms # Sender Sender Sender sectors to SERDS receipt terms 1 Sender sends original message IERDS RERDS Receiving side containing RelayAcceptance 2 Accepts submission and generates and stores XML_SUB_ACC ERDS evidence evidence to RERDS. RERDS acceptance evidence to RERDS. RERDS acceptance 3 Generates (ERD_dispatch_1>_WITH_X KML_SUB_ACC to IERDS evidence to IERDS in the receipt message. 4 Relies < ERD_dispatch_1>_WITH_X Receives < ERD_dispatch_1>_WITH_X the original is dealog with the generates and sends the second with submison for the cept and sends back the notification For Acceptance evidence to Networks.	Para	ameter: {NotificationforA	cceptance}						
Parameter: <erds_receipt_3>_WITH_EV_SET#1 submission of the original message it generates and relies IERDS a dispatch message. IERDS accepts the relay message and sends back the relay message and sends back the relay message and with SubmissionAcceptance evidence to RERDS. RERDS accepts the relay message and sends back RelayAcceptance evidence to IERDS in the receipt message. 2 Accepts submission and generates and stores XML_SUB_ACC ERDS evidence Sender sends original generates accepts the relay message and sends back RelayAcceptance evidence to IERDS. It he receipt message. 3 Generates < ERD_dispatch_1>_WITH_ XML_SUB_ACC to IERDS The RERDS generates and sends the NotificationForAcceptance to N entities at receiving side along with the generation of XML_SUB_ACC to IERDS</erds_receipt_3>	Para	ameter: <erds_receipt_< td=""><td>2>_WITH_XML_REL_ACC</td><td></td><td></td><td></td><td></td><td></td><td>Where the SERDS accepts the</td></erds_receipt_<>	2>_WITH_XML_REL_ACC						Where the SERDS accepts the
Parameter: <erds_receipt_4>_WITH_XML_CONS_ACC it generates and relies IERDS accepts Parameter: {hndvMet, user content} dispatch message. IERDS accepts the relay message and sends back # Sender SERDS IERDS RERDS receiving side containing RelayAcceptance 1 Sender sends original message Accepts submission and generates and stores generates and stores evidence evidence evidence to IERDS in the receipt message and sends back 2 Accepts submission and generates and stores generates and stores evidence evidence to IERDS in the receipt message and sends back RelayAcceptance 3 Generates Generates evidence evidence to IERDS in the receipt message. 4 Relies The RERDS generates and stores XML_SUB_ACC the NotificationForAcceptance evidence to IERDS in the receipt message. 5 XML_SUB_ACC Receives evidence to IERDS in the receipt message. 5 Receives Receives entities at receiving side along with the NotificationForAcceptance evidence of RED_dispatch_1>_WITH_X NotificationForAcceptance evidence with the generation of NotificationForAcceptance evidence with the generation of the second evidence of RED_dispatch_1>_WITH_X</erds_receipt_4>	Para	Parameter: <erds_receipt_3>_WITH_EV_SET#1</erds_receipt_3>							submission of the original message,
Parameter: {hndvMet, user content} dispatch message. IERDS accepts # Sender SERDS IERDS RERDS Receiving side 1 Sender sends original message IERDS RERDS Receiving side containing RelayAcceptance 2 Accepts submission and generates and stores generates and stores generates and stores generates and stores 3 Generates Generates Generates sends participatch_1>_WITH_ XML_SUB_ACC KML_SUB_ACC Receives sends participatch_1>_WITH_ XML_SUB_ACC to IERDS Receives The RERDS generates and sends participatch_1>_WITH_X XML_SUB_ACC to IERDS Receives the notificationForAcceptance evidence to Networks in the receipt message.	Para	Parameter: <erds_receipt_4>_WITH_XML_CONS_ACC</erds_receipt_4>							it generates and relies IERDS a
Sequence of actions the relay message and sends back to SERDS a receipt message # Sender SERDS IERDS RERDS Receiving side containing RelayAcceptance 1 Sender sends original message	Para	ameter: {hndvMet, user o	content}						dispatch message. IERDS accepts
# Sender SERDS IERDS RERDS Receiving side to SERDS a receipt message 1 Sender sends original message		•	t.	Sequence	of actions				the relay message and sends back
1 Sender sends original message Containing RelayAcceptance 2 Accepts submission and generates and stores XML_SUB_ACC ERDS evidence generates a dispatch message alo 3 Generates evidence to RERDS. RERDS accepts the relay message and evidence to RERDS. network sends back RelayAcceptance 4 Relies <erd_dispatch_1>_WITH_ XML_SUB_ACC to IERDS The RERDS generates and sends the NotificationForAcceptance to N entities at receiving side along with the generation of XML_SUB_ACC to IERDS</erd_dispatch_1>	#	Sender	SERDS	IERDS		RERDS		Receiving side	to SERDS a receipt message
1 School of denote on generates and stores generates and stores and stores XML_SUB_ACC ERDS generates and stores XML_SUB_ACC ERDS generates and stores accepts the relay message and sevidence to RERDS. RERDS accepts the relay message and sends back RelayAcceptance evidence to IERDS in the receipt message. 3 Generates Generates Generates Generates Generates Generates and stores XML_SUB_ACC ERDS generates and stores XML_SUB_ACC MITH_ 4 Relies CERD_dispatch_1>_WITH_ 5 Receives Receives Receives CERD_dispatch_1>_WITH_X	1	Sender sends original							
2 Accepts submission and generates and stores XML_SUB_ACC ERDS evidence generates and stores XML_SUB_ACC ERDS evidence to RERDS. RERDS accepts the relay message and sends back RelayAcceptance evidence to IERDS in the receipt message. 3 Generates <erd_dispatch_1>_WITH_ XML_SUB_ACC The RERDS generates and sends the NotificationForAcceptance to N entities at receiving side along with the generation of XML_SUB_ACC to IERDS 5 Receives <erd_dispatch_1>_WITH_X Receives <erd_dispatch_1>_WITH_X</erd_dispatch_1></erd_dispatch_1></erd_dispatch_1>		message							evidence. Alterwards, IERDS
a generates and stores XML_SUB_ACC ERDS evidence with SubhissionAcceptance evidence to RERDS. accepts the relay message and sends back RelayAcceptance evidence to IERDS in the receipt message. 3 Generates <erd_dispatch_1>_WITH_ XML_SUB_ACC evidence to IERDS in the receipt message. 4 Relies <erd_dispatch_1>_WITH_ XML_SUB_ACC to IERDS The RERDS generates and sends the NotificationForAcceptance to N entities at receiving side along with the generation of NotificationForAcceptance evidence</erd_dispatch_1></erd_dispatch_1>	2		Accepts submission and						generates a dispatch message along
SML_SUB_ACC ERDS evidence accepts the relay message and sends back RelayAcceptance 3 Generates evidence to IERDS in the receipt message. 4 Relies The RERDS generates and sends <erd_dispatch_1>_WITH_ XML_SUB_ACC The RERDS generates and sends 5 Receives evidence to IERDS 5 Receives the generation of NotificationForAcceptance evidence</erd_dispatch_1>	_		generates and stores						with SubmissionAcceptance
accepts the feasy message and sends accepts the feasy message and sends sends back RelayAcceptance accepts the feasy message and sends sends back RelayAcceptance evidence to IERDS in the receipt message. 4 Relies <erd_dispatch_1>_WITH_ XML_SUB_ACC to IERDS 5 Receives <erd_dispatch_1>_WITH_X 6 Receives <erd_dispatch_1>_WITH_X</erd_dispatch_1></erd_dispatch_1></erd_dispatch_1>			XML SUB ACC ERDS						accepts the relay message and
3 Generates <erd_dispatch_1>_WITH_ XML_SUB_ACC evidence to IERDS in the receipt message. 4 Relies <erd_dispatch_1>_WITH_ XML_SUB_ACC to IERDS The RERDS generates and sends the NotificationForAcceptance to N entities at receiving side along with the generation of Network and the generation of Network and the sender and the sender</erd_dispatch_1></erd_dispatch_1>			evidence						sends back RelavAccentance
<erd_dispatch_1>_WITH_ XML_SUB_ACC message. 4 Relies <erd_dispatch_1>_WITH_ XML_SUB_ACC to IERDS The RERDS generates and sends the NotificationForAcceptance to N entities at receiving side along with the generation of Network of the receiver of the sender NotificationForAcceptance evidence 5 Receives <erd_dispatch_1>_WITH_X NotificationForAcceptance evidence</erd_dispatch_1></erd_dispatch_1></erd_dispatch_1>	3		Generates						evidence to IERDS in the receipt
XML_SUB_ACC Incodego 4 Relies <erd_dispatch_1>_WITH_ XML_SUB_ACC to IERDS The RERDS generates and sends the NotificationForAcceptance to N entities at receiving side along with the generation of NotificationForAcceptance evidence 5 Receives <erd_dispatch_1>_WITH_X Nut_SUB_ACC NotificationForAcceptance evidence</erd_dispatch_1></erd_dispatch_1>			<erd_dispatch_1>_WITH_</erd_dispatch_1>						message.
4 Relies <erd_dispatch_1>_WITH_ XML_SUB_ACC to IERDS The RERDS generates and sends the NotificationForAcceptance to N entities at receiving side along with the generation of VerticationForAcceptance evidence 5 Receives <erd_dispatch_1>_WITH_X NotificationForAcceptance evidence</erd_dispatch_1></erd_dispatch_1>			XML_SUB_ACC						
<erd_dispatch_1>_WITH_ XML_SUB_ACC to IERDS the NotificationForAcceptance to N entities at receiving side along with the generation of <erd_dispatch_1>_WITH_X 5 Receives <erd_dispatch_1>_WITH_X NotificationForAcceptance evidence</erd_dispatch_1></erd_dispatch_1></erd_dispatch_1>	4		Relies						The RERDS generates and sends
XML_SUB_ACC to IERDS entities at receiving side along with 5 Receives the generation of <			<erd_dispatch_1>_WITH_</erd_dispatch_1>						the NotificationForAcceptance to N
5 Receives <erd_dispatch_1>_WITH_X ML_SUB_ACC</erd_dispatch_1>			XML_SUB_ACC to IERDS						entities at receiving side along with
<erd_dispatch_1>_WITH_X NotificationForAcceptance evidence</erd_dispatch_1>	5			Receives					the generation of
				<erd_dispatch_1< td=""><td>>_WITH_X</td><td></td><td></td><td></td><td>NotificationForAcceptance evidence.</td></erd_dispatch_1<>	>_WITH_X				NotificationForAcceptance evidence.
When all the entities on receiving				ML_SUB_ACC					When all the entities on receiving
6 Generates and stores side responded positively, then	6			Generates and sto	ores				side responded positively, then
XML_REL_ACC ERDS RERDS generates and sends back				XML_REL_ACC E	RDS				RERDS generates and sends back
evidence to IERDS a				evidence					to IERDS a
7 Generates ConsignmentAcceptance evidence	7			Generates					ConsignmentAcceptance evidence.
<pre></pre>				<pre><erds_receipt_1< pre=""></erds_receipt_1<></pre>	>_WITH_X				
ML_SUB_ACC The RERDS generates and				ML_SUB_ACC					The RERDS generates and
8 Sends consigns a structure containing use	8			Sends					consigns a structure containing user
<pre><pre><pre><pre>Content to N recipients on receiving</pre></pre></pre></pre>				<erds_receipt_1< td=""><td>>_vviiH_X</td><td></td><td></td><td></td><td>content to N recipients on receiving</td></erds_receipt_1<>	>_vviiH_X				content to N recipients on receiving
ML_SUB_ACC back to				ML_SUB_ACC ba	CK to				side. Upon successful consignment
TO N entities on the receiving side,			Baaaiyaa	SERDS					BERDS generates and condo back
Receives RERDS generates and senus back	Э		FDDS receipt 1. MUTU	Relies					the receipt containing
XML_SUB_ACC and stores <= <erd_dispatch_1>_WITH_X ContentConsignment evidence to</erd_dispatch_1>			XMI SUB ACC and stores	<erd_dispatch_1< td=""><td>>_WITH_X</td><td></td><td></td><td></td><td>ContentConsignment evidence to</td></erd_dispatch_1<>	>_WITH_X				ContentConsignment evidence to
ERDS evidence ML_SUB_ACC} to RERDS				ML_SUB_ACC} to	RERDS				the previous ERDS

	Scenario id: SERDS_SN_IERDS_SF_RERDS_BB_NO_NOT_F_ACC#1	Purpose
10	Receives <erd_dispatch_1>_WITH_XM L_SUB_ACC</erd_dispatch_1>	
11	Generates and stores XML_REL_ACC ERDS evidence	
12	Generates <erds_receipt_2>_WITH_XM L_REL_ACC</erds_receipt_2>	
13	Sends <erds_receipt_2>_WITH_XM L_REL_ACC back to IERDS</erds_receipt_2>	
14	Receives Sends <erds_receipt_2>_WITH_X {NotificationforAcceptance} ML_REL_ACC Kends</erds_receipt_2>	
15	Generates and stores All the entites at receiving XML_NOT_F_ACC ERDS side receive {NotificationforAcceptance}	
16	Generates and stores XML_NOT_DEL ERDS evidence	
17	Generates All the entites at receiving <erds_receipt_3>_WITH_EV_side access IERDS and SET#1 accept consignment</erds_receipt_3>	
18	Sends <erds_receipt_3>_WITH_EV_ SET#1 back to IERDS</erds_receipt_3>	
19	Receives <erds_receipt_3>_WITH_E V_SET#1 Generates {hndvMet, user content} structure</erds_receipt_3>	
20	Sends Consigns {hndvMet, user <erds_receipt_3>_WITH_E content} structure to N entities V_SET#1 back to SERDS at receiving side</erds_receipt_3>	
21	Receives Generates and stores N {hndvMet, user content} <erds_receipt_3>_WITH_ Generates and stores structure consignments EV_SET#1 and stores XML_CONS_ACC N entities structure consignments ERDS evidences succeed</erds_receipt_3>	
22	Generates <erds_receipt_4>_WITH_XM L_CONS_ACC</erds_receipt_4>	
23	Sends <erds_receipt_4>_WITH_XM L_CONS_ACC back to IERDS</erds_receipt_4>	
24	Receives <erds_receipt_4>_WITH_X ML_CONS_ACC</erds_receipt_4>	

	Scenario id: SE	Purpose		
25		Sends <erds_receipt_4>_WITH_X ML_CONS_ACC back to SERDS</erds_receipt_4>		
26	Receives <erds_receipt_4>_WITH_ XML_CONS_ACC and stores ERDS evidence</erds_receipt_4>			

Table 8a: Scenarios where SERDS operates Store&Notify and RERDSs operate Store&Forward (2/3)

		Scenario id: SERI	DS_SN_IERDS_SF_RE	RDS_BB_I	NO_NOT_F_ACC#2		Purpose
#	Sender	SERDS	IERDS		RERDS	Receiving side	
Param	neter: <erd_dispatch_< td=""><td>1>_WITH_XML_SUB_ACC</td><td>Var E</td><td>EV_SET#1</td><td>= {XML_NOT_F_ACC, XM</td><td>L_NOT_DEL}</td><td></td></erd_dispatch_<>	1>_WITH_XML_SUB_ACC	Var E	EV_SET#1	= {XML_NOT_F_ACC, XM	L_NOT_DEL}	
Param	neter: <erds_receipt_< td=""><td>1>_WITH_XML_REL_ACC</td><td>Var E</td><td>EV_SET#2</td><td>= {XML_CONS_ACC, XML</td><td>_CONS_REJ}</td><td>A negative scenario of</td></erds_receipt_<>	1>_WITH_XML_REL_ACC	Var E	EV_SET#2	= {XML_CONS_ACC, XML	_CONS_REJ}	A negative scenario of
Param	neter: {NotificationforAc	cceptance}					EXT_RERDS_NOT_F
Param	neter: <erds_receipt_< td=""><td>_2>_WITH_XML_REL_ACC</td><td></td><td></td><td></td><td></td><td><u>_ACC #1</u>:</td></erds_receipt_<>	_2>_WITH_XML_REL_ACC					<u>_ACC #1</u> :
Param	neter: <erds_receipt_< td=""><td>_3>_WITH_EV_SET#1</td><td></td><td></td><td></td><td></td><td></td></erds_receipt_<>	_3>_WITH_EV_SET#1					
Param	neter: <erds_receipt_< td=""><td>_4>_WITH_EV_SET#2</td><td></td><td></td><td></td><td></td><td>When one of the</td></erds_receipt_<>	_4>_WITH_EV_SET#2					When one of the
Param	neter: <erds_receipt_< td=""><td>_5>_WITH_XML_REL_ACC</td><td></td><td></td><td></td><td></td><td></td></erds_receipt_<>	_5>_WITH_XML_REL_ACC					
Param	neter: {hndvMet, user c	content}					
Param	neter: <erds_receipt_< td=""><td>_6>_WITH_XML_CONT_CONS</td><td></td><td></td><td></td><td></td><td>deperates and sends</td></erds_receipt_<>	_6>_WITH_XML_CONT_CONS					deperates and sends
			Sequence of act	tions			back to IERDS a
#	Sender	SERDS	IERDS		RERDS	Receiving side	
1	Sender sends						containing a set of
	original message						evidence (including
2		Accepts submission and					both N-1
							ConsignmentAcceptan
		AWL_SOB_ACC ERDS					ce and 1
3		Generates					
Ŭ		<pre><frd 1="" dispatch=""> WITH</frd></pre>					deperates and
		XML SUB ACC					consigns a structure
4		Relies					containing user content
		<erd_dispatch_1>_WITH_</erd_dispatch_1>			1		to N-1 recipients on
		XML_SUB_ACC to IERDS					receiving side. Upon
5			Receives				successful
			<erd_dispatch_1>_W</erd_dispatch_1>	/ITH_X			consignment to N -1
			ML_SUB_ACC				entities on the
6			Generates and stores				receiving side, RERDS
			XML_REL_ACC ERDS	S			generates and sends
7			evidence				Dack to IERDS a
1			Generates				
				/118_X			evidence
0			Sonda				
0			ERDS receipt 1 W				
			MI REL ACC back to))			
			SERDS	,			
9		Receives					—
		<erds_receipt_1>_WITH</erds_receipt_1>	Generates				
		XML_REL_ACC and stores	{NotificationforAccepta	ance}			
		ERDS evidence	[-			

		Scenario id: SERI	DS_SN_IERDS_SF_RERDS_BE	3_NO_NOT_F_ACC#2		Purpose
#	Sender	SERDS	IERDS	RERDS	Receiving side	
10			Relies <erds_notification_1>_forAc ceptance to RERDS</erds_notification_1>			
11				Receives <erds_notification_1>_forAcc eptance</erds_notification_1>		
12				Generates XML_REL_ACC ERDS evidence		
13				Generates <erds_receipt_2>_WITH_XM L_REL_ACC</erds_receipt_2>		
14				Sends <erds_receipt_2>_WITH_XM L_REL_ACC back to IERDS</erds_receipt_2>		
15			Receives <erds_receipt_2>_WITH_X ML_REL_ACC</erds_receipt_2>	Sends {NotificationforAcceptance}		
16				Generates and stores XML_NOT_F_ACC ERDS evidence	All the entities at receiving side receive {NotificationforAcceptance}	
17				Generates and stores XML_NOT_DEL ERDS evidence		
18				Generates <erds_receipt_3>_WITH_EV_ SET#1</erds_receipt_3>		
19				Sends <erds_receipt_3>_EV_SET#1 back to IERDS</erds_receipt_3>		
20			Receives <erds_receipt_3>_WITH_E V_SET#1</erds_receipt_3>	Generates and stores XML_CONS_ACC ERDS evidence for N-1 entities at receiving side and one XML_CONS_REJ ERDS evidence for one entity at receiving side	N-1 entities at receiving side accept consignment. One party rejects it	
21			Sends <erds_receipt_3>_WITH_E V_SEYT#1 back to SERDS</erds_receipt_3>	Generates <erds_receipt_4>_WITH_EV_ SET#2</erds_receipt_4>		
22		Receives <erds_receipt_3>_WITH_ EV_SET#1 and stores ERDS evidence</erds_receipt_3>		Sends <erds_receipt_4>_WITH_EV_ SET#2 back to IERDS</erds_receipt_4>		

		Scenario id: SERI	DS_SN_IERDS_SF_RERDS_BE	3_NO_NOT_F_ACC#2		Purpose
#	Sender	SERDS	IERDS	RERDS	Receiving side	
23			Receives <erds_receipt_4>_WITH_E V_SET#2</erds_receipt_4>	Generates hndvMet, user content} structure		
24			Sends <erds_receipt_4>_WITH_E V_SET#2 back to SERDS</erds_receipt_4>	Consigns hndvMet, user content} structure to N-1 entities at receiving side		
25		Receives <erds_receipt_4>_WITH_ EV_SET#2 and stores ERDS evidences</erds_receipt_4>	Relies <erd_dispatch_1>_WITH_X ML_SUB_ACC to RERDS to RERDS</erd_dispatch_1>	Generates and stores XML_CONT_CONS ERDS evidence for N-1 entities	N-1 {hndvMet, user content} structure consignments succeed	
26		Sends <erds_receipt_4>_WITH_ EV_SET#2 back to sender</erds_receipt_4>		Generates <erds_receipt_5>_WITH_XM L_CONT_CONS</erds_receipt_5>		
27	Receives <erds_receipt_4>_ WITH_EV_SET#2</erds_receipt_4>			Relies <erds_receipt_5>_WITH_XM L_CONT_CONS to IERDS</erds_receipt_5>		
28			Receives <erds_receipt_5>_WITH_X ML_CONT_CONS</erds_receipt_5>			
29			Relies <erds_receipt_5>_WITH_X ML_CONT_CONS to SERDS</erds_receipt_5>			
30		Receives <erds_receipt_5>_WITH_ XML_CONT_CONS and stores ERDS evidence</erds_receipt_5>				

Table 8b: Scenarios where SERDS operates Store&Notify and RERDSs operate Store&Forward (3/3)

		Scenario id: SER	S_SN_IERDS_SF	_RERDS_BB	NO_NOT_F_ACC#3		Purpose
Para	meter: <erd_dispatch_1< td=""><td>>_WITH_XML_SUB_ACC</td><td></td><td>√ar EV_SET#</td><td>1 = {XML_NOT_F_ACC, XML_N</td><td>DT_DEL}</td><td></td></erd_dispatch_1<>	>_WITH_XML_SUB_ACC		√ar EV_SET#	1 = {XML_NOT_F_ACC, XML_N	DT_DEL}	
Para	meter: <erds_receipt_1< td=""><td>>_WITH_XML_REL_ACC</td><td>N N</td><td>√ar EV_SET#</td><td>2 = {XML_CONS_ACC, XML_CC</td><td>NS_REJ}</td><td>A negative scenario of</td></erds_receipt_1<>	>_WITH_XML_REL_ACC	N N	√ar EV_SET#	2 = {XML_CONS_ACC, XML_CC	NS_REJ}	A negative scenario of
Parameter: <erds_notification_1>_forAcceptance</erds_notification_1>				√ar EV_SET#	#3 = {XML_CONT_HAND, XML_C	ONT_HAND_FAIL }	EXT_RERDS_NOT_F_ACC #1:
Para	meter: <erds_receipt_2< td=""><td>>_WITH_XML_REL_ACC</td><td></td><td></td><td></td><td>· · · · · · · · · · · · · · · · · · ·</td><td></td></erds_receipt_2<>	>_WITH_XML_REL_ACC				· · · · · · · · · · · · · · · · · · ·	
Para	meter: <erds_receipt_3< td=""><td>3>_WITH_EV_SET#1</td><td></td><td></td><td></td><td></td><td>Where RERDS generates and</td></erds_receipt_3<>	3>_WITH_EV_SET#1					Where RERDS generates and
Para	meter: <erds_receipt_4< td=""><td>>_WITH_EV_SET#2</td><td></td><td></td><td></td><td></td><td>sends the</td></erds_receipt_4<>	>_WITH_EV_SET#2					sends the
Para	meter: <erds_receipt_5< td=""><td>5>_WITH_XML_REL_ACC</td><td></td><td></td><td></td><td></td><td>NotificationForAcceptance to N</td></erds_receipt_5<>	5>_WITH_XML_REL_ACC					NotificationForAcceptance to N
Para	meter: <erds_receipt_6< td=""><td>>_WITH_XML_CONT_CONS</td><td></td><td></td><td></td><td></td><td>entities at receiving side. Then it</td></erds_receipt_6<>	>_WITH_XML_CONT_CONS					entities at receiving side. Then it
Para	meter: <erds_receipt_7< td=""><td>/>_WITH_EV_SET#3</td><td></td><td></td><td></td><td></td><td>generates and sends a receipt</td></erds_receipt_7<>	/>_WITH_EV_SET#3					generates and sends a receipt
			Sequence o	of actions			NotificationEcrAccentance
#	Sender	SERDS	IERDS	6	RERDS	Receiving side	evidence back to IERDS When N-
	Sender sends original						1 the entities on receiving side
1	message						respond positively while 1
2		Accepts submission and					negatively, the IERDS generates
		generates XML_SUB_ACC					and sends back to SERDS a
		ERDS evidence					receipt message containing a set
3		Generates					of evidence (including N-1
		<erd_dispatch_1>_WITH_</erd_dispatch_1>					ConsignmentAcceptance and 1
		XML_SUB_ACC					ConsignmentRejection evidence).
4		Relies					
		<erd_dispatch_1>_WITH_</erd_dispatch_1>					Afterwards, RERDS generates and
_		XML_SUB_ACC to IERDS	<u> </u>				consigns a structure containing
5			Receives				user content to N-1 recipients on
			<pre><erd_dispatch_1< pre=""></erd_dispatch_1<></pre>	>_WIIH_X			receiving side. Upon successful
_			ML_SUB_ACC an	d stores it			consignment to N-1 entities on the
6			Generates XML_F	REL_ACC			receiving side, RERDS generates
7			ERDS evidence				and sends back the receipt
1			Generates				evidence to the previous ERDS
				>_\\\\\			evidence to the previous ENDS.
8			Sende				The cycle of message events in
0			<pre>>ERDS receipt 1</pre>	< WITH X			this scenario is completed with the
			MI SUB ACC ba	ck to			generation of a receipt message
			SERDS				containing a set of evidence
9		Receives	Generates				(including ContentHandover and
Ŭ		<pre><frds 1="" receipt=""> WITH</frds></pre>	<erds notificatio<="" td=""><td>on> forAcce</td><td></td><td></td><td>ContentHandoverFail evidence)</td></erds>	on> forAcce			ContentHandoverFail evidence)
		XML SUB ACC	ptance				and sends it back to IERDS.
10			Relies				
			<erds notification<="" td=""><td>n 1> forAc</td><td></td><td></td><td></td></erds>	n 1> forAc			
			ceptance to RERE	DS			
11			•		Receives		
					<erds_notification_1>_forAcc</erds_notification_1>		
1					eptance		

	Scenario id: SERDS	6_SN_IERDS_SF_RERDS_BE	3_NO_NOT_F_ACC#3		Purpose
12			Generates XML_REL_ACC ERDS evidence		
3			Generates <erds_receipt_2>_WITH_XM L_REL_ACC</erds_receipt_2>		
4			Sends <erds_receipt_2>_WITH_XM L_REL_ACC back to IERDS</erds_receipt_2>		
5		Receives <erds_receipt_2>_WITH_X ML_REL_ACC</erds_receipt_2>	Sends <erds_notification_1>_forAcc eptance</erds_notification_1>		
6			Generates XML_NOT_F_ACC ERDS evidence	All the entities at receiving side receive <erds_notification_1>_for Acceptance</erds_notification_1>	
17			Generates XML_NOT_DEL ERDS evidence		
18			Generates <erds_receipt_3>_WITH_EV_ SET#1</erds_receipt_3>		
9			Sends <erds_receipt_3>_WITH_EV_ SET#1 back to IERDS</erds_receipt_3>		
20	F	Receives <erds_receipt_3>_WITH_E V_SET#1</erds_receipt_3>			
21		Sends <erds_receipt_3>_WITH_E V_SET#1 back to SERDS</erds_receipt_3>		N-1 entities at receiving side access IERDS and accept consignment. One party rejects it	
22	Receives <erds_receipt_3>_WITH_ EV_SET#1</erds_receipt_3>	Generates XML_CONS_ACC ERDS evidence for N-1 entities at receiving side and one XML_CONS_REJ ERDS evidence for one entity at receiving side			
23		Generates <erds_receipt_4>_WITH_E V_SET#2</erds_receipt_4>			
24		Sends <erds_receipt_4>_WITH_E V_SET#2 back to SERDS</erds_receipt_4>			
25	Receives <erds_receipt_4>_WITH_ EV_SET#2</erds_receipt_4>	Relies <erd_dispatch_1>_WITH_X ML_SUB_ACC to RERDS to RERDS</erd_dispatch_1>			

		Scenario id: SERD	DS_SN_IERDS_SF_RERDS_BE	B_NO_NOT_F_ACC#3		Purpose
26		Sends		Accepts		
		EV_SET#2 back to sender		L_SUB_ACC		
27	Receives <erds_receipt_4>_W ITH_EV_SET#2</erds_receipt_4>			Generates XML_REL_ACC ERDS evidence		
28				Generates <erds_receipt_5>_WITH_XM L_REL_ACC</erds_receipt_5>		
29				Sends <erds_receipt_5>_WITH_XM L_REL_ACC back to IERDS</erds_receipt_5>		
30				Generates <erd_dispatch_2>_WITH_XM L_SUB_ACC</erd_dispatch_2>		
31			Receives <erds_receipt_5>_WITH_X ML_REL_ACC</erds_receipt_5>	Consigns <erd_dispatch_2>_WITH_XM L_SUB_ACC to N-1 entities at receiving side</erd_dispatch_2>		
32					<erd_dispatch_2>_WITH_ XML_REL_ACC is consigned to N-1 entities at receiving side</erd_dispatch_2>	
33				Generates XML_CONT_CONS		
34				Generates <erds_receipt_6>_WITH_XM L_CONT_CONS</erds_receipt_6>		
35					N-2 entities in receiving side retrieve user content, one entity fails	
36			Receives <erds_receipt_6>_WITH_X ML_CONT_CONS</erds_receipt_6>	Generates XML_CONT _HAND evidence for N-2 entities and XML_CONT_HAND_FAIL for one entity		
37			sends <erds_receipt_6>_WITH_X ML_CONT_CONS back to SERDS</erds_receipt_6>	Generates <erds_receipt_7>_WITH_EV_ SET#3</erds_receipt_7>		
38		Receives <erds_receipt_7>_WITH_ XML_CONT_CONS</erds_receipt_7>		Sends <erds_receipt_7>_WITH_EV_ SET#3 back to IERDS</erds_receipt_7>		
39		Sends <erds_receipt_7>_WITH_ XML_CONT_CONS back to sender</erds_receipt_7>	Receives <erds_receipt_7>_WITH_E V_SET#3</erds_receipt_7>			

		Scenario id: SERD	S_SN_IERDS_SF_RERDS_BB	_NO_NOT_F_ACC#3	Purpose
40	Receives <erds_receipt_7>_W ITH_XML_CONT_CO NS</erds_receipt_7>		sends <erds_receipt_7>_WITH_E V_SET#3 back to SERDS</erds_receipt_7>		
41		Receives <erds_receipt_7>_WITH_ EV_SET#3</erds_receipt_7>			
42		Sends i <erds_receipt_7>_WITH_ EV_SET#2 back to sender</erds_receipt_7>			
43	Receives <erds_receipt_7> WITH EV SET#2</erds_receipt_7>				

6 ERD Messages instances

6.1 Introduction and technical approach

The present clause defines a number of instances of the different types of ERD Messages, namely: ERDS notification, ERD payload, ERDS receipt, and ERD dispatch, as defined in ETSI EN 319 522-1 [1]. These instances are used in clause 8 for defining different test cases.

83

The set of ERD messages instances is built following the technical approach shown below:

- The set includes instances of each type of ERD message.
- For each type of ERD message there will be at least one instance including all the optional components of relay metadata components defined in ETSI EN 319 522-3 [3].
- The present document first defines different combinations for the metadata components profiled in ETSI EN 319 522-4-1 [4] and in ETSI EN 319 522-4-2 [5], and for the relay metadata components defined in ETSI EN 319 522-2 [2].
- Each instance of a certain type of ERD message is defined as a composition of different metadata components combinations already mentioned.

The rest of the present clause is organized as follows:

- Clause 6.2 presents a number of combinations of metadata components for the different headers that can be present in the different ERD message types. These combinations are specified separately as they are used in the definition of instances of different ERD message types.
- Clause 6.3 defines instances of ERD payloads.
- Clause 6.4 defines instances of ERDS receipts.
- Clause 6.5 defines instances of ERD dispatches.

As mentioned in clause 4.2 new combinations of metadata components may be defined for each header, and new instances of ERD messages may be added to the current set, for defining new test cases.

6.2 Combinations of fields for headers in ERD envelopes

6.2.1 Introduction

Clause 6.2.2 defines combinations of values for the metadata components defined in OASIS Standards [i.2] and [8] and further profiled in ETSI EN 319 522-4-1 [4].

Clause 6.2.3 defines combinations of values for the relay metadata components defined in ETSI EN 319 522-3 [3].

Clause 6.2.4 defines aggregations of combinations defined in clauses 6.2.2 and 6.2.3.

6.2.2 Combinations of AS4 metadata profiled in ETSI EN 319 522-4

The present clause defines different combinations of values for the metadata components defined in OASIS Standards [i.2] and [8] and further profiled in ETSI EN 319 522-4-1 [4].

These combinations are shown in Table 9.

Cells in column "Component name" contain the name of the metadata components.

Cells in column "Value" shows the value to be assigned to the metadata component. These cells may contain the following values:

- As specified in ETSI EN 319 522-4-1 [4]. This value is reserved for cases where the ETSI EN 319 522-4-1 [4] specifies a mandatory value for the metadata component.
- As recommended in ETSI EN 319 522-4-1 [4]. This value is reserved for cases where the ETSI EN 319 522 4-1 [4] recommends one certain value for the metadata component (usually using the modal verb should). In these cases, the metadata component has this recommended value in the combination.
- TEST_DEPENDANT means that the value of the metadata component will be defined in test cases for testing ERD message formats.

Cells in "Notes/Additional requirements" contain one or more letters or/and one or more integer numbers. The letters correspond to additional requirements that are given after the table. The numbers correspond to numbers of notes that appear after the aforementioned additional requirements.

Combination identifier	Component / Feature name	Value	Notes/Additional requirements					
	PMode.Initiator	As specified in ETSI EN 319 522-4-1 [4]						
	PMode.Responder	As specified in ETSI EN 319 522-4-1 [4]						
	PMode.Initiator.Role	As specified in ETSI EN 319 522-4-1 [4]						
	PMode.Responder.Role	As specified in ETSI EN 319 522-4-1 [4]						
	PMode[1].BusinesssInfo.Service	As specified in ETSI EN 319 522-4-1 [4]						
AS4Met#1	PMode[1].Security.SendReceipt	As specified in ETSI EN 319 522-4-1 [4]	1					
	PMode[1].Security.SendReceipt.NonRepudiation	As specified in ETSI EN 319 522-4-1 [4]	1					
	PMode[1].Security.SendReceipt.ReplyPattern	As specified in ETSI EN 319 522-4-1 [4]	2					
	PMode[1].ErrorHandling.Report.AsResponse	As specified in ETSI EN 319 522-4-1 [4]	2					
	AS4 Reception Awareness Feature	As recommended in ETSI EN 319 522-4-1 [4]						
	Duplication elimination function	As recommended in ETSI EN 319 522-4-1 [4]						
NOTE 1: ETSI EN NOTE 2: ETSI EN	NOTE 1: ETSI EN 319 522-4-1 [4] profiles this component for signed receipts. NOTE 2: ETSI EN 319 522-4-1 [4] profiles this component for signed receipts and error signal messages.							

Table 9: Combinations of metadata values as defined in OASIS specifications

6.2.3 Combinations of components of relay metadata

Table 10 shows combinations of values of the relay metadata defined in ETSI EN 319 522-3 [3].

For the purpose of defining the test cases, the field "ApplicablePolicy" shall always consist in a single URI. New combinations may be added where this field consists in a sequence of URIs.

Cells in "Component name" column contain the name of relay metadata component in the ERD message. The names used are the ones defined in ETSI EN 319 522-3 [3].

Cells in "Component value" column contain either:

- 1) the value of the relay metadata component whose name is the one indicated in the previous column; or
- 2) AS_PER_TESTCASE, meaning that for this test case the ERDS is free to give to the aforementioned field the value it considers worth.

Cells in "Notes/Additional requirements" contain one or more letters or/and one or more integer numbers. The letters correspond to additional requirements that are given after the table. The numbers correspond to numbers of notes that appear after the aforementioned additional requirements.

85

Cells in "Purpose" contain a description of the purpose of the combination defined in the row.

Table 10 defines parameterized combinations for new headers defined in ETSI EN 319 532-3 [7]. The combinations have as parameters the assurance levels and the consignment mode.

Taple TV. Falance ized complianons for relay menada components	Table 10: Parameterized	combinations for re	elav metadata con	nponents
--	-------------------------	---------------------	-------------------	----------

Combination identifier	Header field name	Header field value	Notes/Additional	Purpose	
			requirements		
RelayMet#1	RelayTime	AS_PER_TESTCASE	a,	Use in tests where: the contents of the components are	
	ExpirationTime	AS_PER_TESTCASE	b	correct; there is no indication neither of assurance	
	ScheduledDeliveryTime	AS_PER_TESTCASE	с	levels nor of mode of consignment.	
	ApplicablePolicy	AS_PER_TESTCASE			
RelayMet#2	RelayTime	AS_PER_TESTCASE	а	Use in tests where: the contents of the components are	
	ExpirationTime	AS_PER_TESTCASE	b	correct; the assurance levels required is one	
	RequiredAssuranceLevel	Parameter		parameter, and there is no indication of mode of consignment.	
	ScheduledDeliveryTime	AS_PER_TESTCASE	С		
	ApplicablePolicy	AS_PER_TESTCASE			
RelayMet#3	RelayTime	AS_PER_TESTCASE	а	Use in tests where: the contents of the components are	
	ExpirationTime	AS_PER_TESTCASE	b	correct; relay metadata include all the optional	
	RequiredAssuranceLevel	Parameter		components present; and the assurance levels and the	
	RequestedConsignmentMode	Parameter		mode of consignment are parameters.	
	ScheduledDeliveryTime	AS_PER_TESTCASE	с		
	ApplicablePolicy	AS_PER_TESTCASE			
RelayMet#4	RelayTime	AS_PER_TESTCASE	a,	Use in negative tests where: the scheduled delivery is after the expiration date; assurance levels indication is	
	ExpirationTime	AS_PER_TESTCASE	b		
	REM-RecipientAssuranceLeve	Parameter		one parameter, and there is no indication of mode of	
	ScheduledDeliveryTime	AS_PER_TESTCASE	d	consignment.	
	ApplicablePolicy	AS_PER_TESTCASE			
RelayMet#5	RelayTime	AS_PER_TESTCASE	а	Use in negative tests where: the scheduled delivery is after the expiration date; and the assurance levels and	
	ExpirationTime	AS_PER_TESTCASE	b		
	RequiredAssuranceLevel	Parameter		the mode of consignment are parameters.	
	RequestedConsignmentMode	Parameter			
	ScheduledDeliveryTime	AS_PER_TESTCASE	d		
	ApplicablePolicy	AS_PER_TESTCASE			
RelayMet#6	RelayTime	AS_PER_TESTCASE	а	As RelayMet#1 but without ScheduledDeliveryTime.	
	ExpirationTime	AS_PER_TESTCASE			
	ApplicablePolicy	AS_PER_TESTCASE			
RelayMet#7	RelayTime	AS_PER_TESTCASE	а	As RelayMet#2 but without ScheduledDeliveryTime.	
	ExpirationTime	AS_PER_TESTCASE			
	RequiredAssuranceLevel	Parameter			
	ApplicablePolicy	AS_PER_TESTCASE			
RelayMet#8	RelayTime	AS_PER_TESTCASE	а	As RelayMet#3 but without ScheduledDeliveryTime.	
	ExpirationTime	AS_PER_TESTCASE			
	RequiredAssuranceLevel	Parameter		7	
	RequestedConsignmentMode	Parameter		7	
	ApplicablePolicy	AS_PER_TESTCASE			

Additional requirements:

- a) The date and time indicated in this component shall be earlier than the date and time indicated in "ERD-ExpirationDate" and "ERD-ScheduledDelivery" (if present).
- b) The date and time indicated in this component shall be later than the date and time indicated in "ERD-RelayDate" and "ERD-ScheduledDelivery" (if present).
- c) The date and time indicated in this component shall be earlier than the date and time indicated in "ERD-ExpirationDate" and "ERD-ScheduledDelivery" and later than the date and time indicated in "ERD-RelayDate.
- d) The date and time indicated in this component shall be later than the date and time indicated in "ERD-RelayDate" and later than "ERD-ExpirationDate" (if present).

6.2.4 Combinations of AS4 metadata profiled and relay metadata

The present clause defines aggregations of AS4 metadata components defined in OASIS Standards [i.2] and [8] and further profiled in ETSI EN 319 522-4-1 [4], and relay metadata. These combinations are shown in Table 11.

Each metadata combination instance is defined as the aggregation of the combination defined in Table 9 and one of the combinations defined in Table 10.

Each combination defined in Table 10 has 4 parameters, namely:

- 1) An integer identifying the AS4 metadata combination in table of clause 6.2.2.
- 2) An integer identifying the relay metadata combination in table of clause 6.2.3.
- 3) The value of the required assurance levels (nil if this information is not present in the combination).
- 4) The value for consignment mode (nil if this information is not present).

All the metadata combinations instances can be obtained from the following expression:

Metadata(AS4-metadata#I, Relay-Metadata#J,AssuranceLevelComb, ConsignmentMode) where:

- I, for the combinations specified in the present document is always 1.
- J is one of {1 to 8}.
- AssuranceLevelCombs is one of {nil, low/low, subs/subs, high/high}.
- ConsignmentMode is one of {nil, basic, consented, consentedSigned}.

Table 11 defines a number of possible combinations of these parameters when the required assurance levels are the same for sender and entities at receiving side. It also defines combinations where the consignment mode field is not present. Finally, it also defines two illegal combinations, that can be used in negative test cases.

Fable 11: ERD) messagesmetada	combinations
---------------	------------------	--------------

For test cases without ShceduledDelivery			
OutermostHeade(1,6,nil,nil)	AS4Met#1 + RelayMet#6	No indications neither of	
		assurance levels nor	
		consignment mode	
Metadata(1,7,low/low,nil)	AS4Met#1 + RelayMet#7(low/low)	No consignment mode	
Metadata(1,7,sub/subs,nil)	AS4Met#1 + RelayMet#7(substantial/substantial)	indicated and assurance	
Metadata(1,7,high/high,nil)	AS4Met#1 + RelayMet#7(high/high)	level	
Metadata (1,8,low/low,basic)	AS4Met#1 + RelayMet#8(low/low, basic)	Basic consignment mode	
Metadata(1,8,subs/subs,basic)	AS4Met#1 + RelayMet#8(substantial/substantial,	and assurance level	
	basic)		
Metadata (1,8,high/high,basic)	AS4Met#1 + RelayMet#8(high/high, basic)		
Metadata (1,8,low/low,consented)	AS4Met#1 + RelayMet#8(low/low, consented)	Consented consignment	
Metadata(1,8,subs/subs,consented)	AS4Met#1 + RelayMet#8(substantial/substantial,	mode and assurance level	
	consented)		
Metadata (1,8,high/high,consented)	AS4Met#1 + RelayMet#8(high/high, consented)		
Metadata (1,8,low/low,consentedSigned)	AS4Met#1 + RelayMet#8(low/low,	Consented and signed	
	consentedSigned)	consignment mode and	
Metadata(1,8,subs/subs,consentedSigned)	AS4Met#1 + RelayMet#8(substantial/substantial,	assurance level	
	consentedSigned)		
Metadata (1,8,high/high,consentedSigned)	AS4Met#1 + RelayMet#8(high/high,		
	consentedSigned)		
For test cases with ShceduledDelivery			
OutermostHeade(1,1,nil,nil)	AS4Met#1 + RelayMet#1	No indications neither of	
		assurance levels nor	
		consignment mode	
Metadata(1,2,low/low,nil)	AS4Met#1 + RelayMet#2(low/low)	No consignment mode	
Metadata(1,2,sub/subs,nil)	AS4Met#1 + RelayMet#7(substantial/substantial)	indicated and assurance	
Metadata(1,2,high/high,nil)	AS4Met#1 + RelayMet#2(high/high)	level	
Metadata (1,3,low/low,basic)	AS4Met#1 + RelayMet#3(low/low, basic)		

For test cases without ShceduledDelivery			
Metadata(1,3,subs/subs,basic)	AS4Met#1 + RelayMet#8(substantial/substantial, basic)	Basic consignment mode and assurance level	
Metadata (1,3,high/high,basic)	AS4Met#1 + RelayMet#3(high/high, basic)		
Metadata (1,3,low/low,consented)	AS4Met#1 + RelayMet#3(low/low, consented)	Consented consignment	
Metadata(1,3,subs/subs,consented)	AS4Met#1 + RelayMet#3(substantial/substantial, consented)	mode and assurance level	
Metadata (1,3,high/high,consented)	AS4Met#1 + RelayMet#3(high/high, consented)		
Metadata (1,3,low/low,consentedSigned)	AS4Met#1 + RelayMet#3(low/low, consentedSigned)	Consented and signed consignment mode and	
Metadata(1,3,subs/subs,consentedSigned)	AS4Met#1 + RelayMet#3(substantial/substantial, consentedSigned)	assurance level	
Metadata (1,3,high/high,consentedSigned)	AS4Met#1 + RelayMet#3(high/high, consentedSigned)		
Metadata (1,4,low/low,nil)	AS4Met#1 + RelayMet#4(low/low)	Combinations for negative	
Metadata (1,5,low/low,basic)	AS4Met#1 + RelayMet#5(low/low, basic)	test cases (cause of submission rejection for instance)	

89

6.3 Instances of ERD payload

As an ERD payload is the aggregation of metadata and user content, the present document defines as many ERD payload instances as metadata instances have been defined in clause 6.2.4.

6.4 Instances of ERDS receipts

An ERD receipt is the aggregation of one metadata payload and one or more ERDS evidence payloads. The number and contents of these payloads will depend on the specific test case.

The present document uses the following convention for identifying sets of ERDS receipts:

ERD_ReceiptInst (Metadata(AS4Met#1,RelayMet#J,nil,nil), userContent,<EVID#M>+)

Where:

• J is a positive integer between 1 and 9 (inclusive), and the + symbol in <EVID#M> indicates that in each case, the ERDS receipt instance shall contain as many payloads for ERDS evidence as ERDS evidences indicated in the test case where the ERDS receipt is used.

This unique content identifies all the possible ERDS receipts that will be needed in the definitions of the test cases.

6.5 Instances of ERD dispatch

The present document uses the following convention for identifying sets of ERD dispatches:

ERDS_DispatchInst(Metadata(AS4Met#1,RelayMet#J, AssLevelComb, ConsignmentModeId), <EVID#M>+)

Where:

• J is a positive integer between 1 and 9 (inclusive), AssLevelComb is one of {low/low, substantial/substantial, high/high}, ConsignmentModeId is one of {basic,consented,consentedSigned}, and the + symbol in <EVID#M> indicates that in each case, the ERD dispatch instance shall contain as many payloads for ERDS evidence as ERDS evidences indicated in the test case where the ERDS receipt is used.

This unique content identifies all the possible ERD dispatches that will be needed in the definitions of the test cases.

7 Other named sets

7.1 Named sets of participating ERDSs

The present clause defines named sets of participating REMSs that are used for naming test cases in clause 5.

The details on the participating ERDSs are given using the following pattern: ERDS(I,J), where:

- I stands for the number of Intermediate ERDSs (IERDSs); and
- J stands for the number of the Recipient's ERDSs (RERDSs).

7.2 Named sets of additional requirements

The present clause defines named sets of additional requirements that are used for building different test cases based on the same scenarios of REM messages.

Table 12 shows the named sets of additional requirements.

Table 12: Named sets of additional requirements

Name of the set	Additional requirements in the set	
Additional Page#1	Original message: not signed, no attachment	
AdditionalReqs#1	Sender sends original message	
	Original message: not signed, no attachment	
AdditionalReqs#2	Sender's delegate	
	Sender's delegate sends original message	

7.3 Named sets of entities in receiving side

The present clause defines named sets of entities that are present at receiving side. This allows using one scenario in defining different test cases by changing the entities in the receiving side.

EXAMPLE: Scenarios defined for one recipient could be used in test cases where the scenarios involve only one delegate of one recipient.

In order to define the names of the sets, the pattern RecSide (I, J, K) is used where:

- I stands for the number of recipients.
- J stands for the number of recipient's delegates.
- K stands for the number of recipients each delegate is delegate of.

K shall always be less or equal than I. If I is not 0 then K shall also be different from 0.

8 Test cases definition

8.1 Introduction

8.1.1 General

The notations shown in clauses 4.1, 6 and 7, allow building a compact notation for defining tests cases.

The present document defines sets of test cases. Each set of test cases is expressed as a function of a number of parameters (some of them are integers, other are tuples of several values, other -mainly reasons for failures- are enumerated values specified in another ETSI deliverable).

Under these conditions one specific test case is obtained when the set is particularized by assigning a single value to each parameter.

8.1.2 Notation for black box model scenarios

No ERD messages are generated in the scenarios defined for the black box model. Consequently, the only parameters the test cases depend on are the additional requirements specified in clause 7.2 and the entities present at receiving side, identified in clause 7.3.

Below follows the notation used for identifying the set of test cases for scenario ERDS_BB_NOT_F_ACC#1:

```
ERDS_BB_NOT_F_ACC#1 (
    RecSide(F,G,H),
    AssLevelComb,ConsignmentModelId,
    AdditionalReqs#P
)
```

This part shows the components required for defining the test cases for this scenario. For this scenario each test case in the set will require providing details of:

- The entities in the receiving side (RecSide). The notation for identifying one of the different alternatives is as specified in clause 7.3.
- The additional requirements, whose notation has been specified in clause 7.2.

Scenarios where there is some failure also include the FailureReason parameter.

Below follows the second part of the definition of the set of test cases:

Where:

- For RecSide: (F,G,H) is one of {(1,0,0), (3,0,0), (3,1,2),(3,1,3)}: define test cases for one recipient, several recipients, and several recipient's delegate.
- For AdditionalReqs: P is one of {1,2}.

8.1.3 Notation for 4 corner and extended models scenarios

For helping in understanding the notation, below follows the definition of the set of test cases for the scenario 4C_RERDS_NO_NOT_F_ACC #2. The definition of a set of test cases has two parts.

Below follows the first one:

```
4C_RERDS_NO_NOT_F_ACC #2 (
    RecSide(F,G,H),
    ERD_dispatchInst_WITH_XML_SUB_ACC
        (Metadata(AS4Met#1, RelayMet#I, AssLevelComb, ConsignmentModeId), UserContent,XML_SUB_ACC),
    ERDS_receipt_WITH_XML_REL_ACC
        (Metadata(AS4Met#1, RelayMet#I, AssLevelComb, ConsignmentModeId),XML_REL_ACC),
AdditionalRegs#P
```

This part shows the components required for defining the test cases for this scenario. For this scenario each test case in the set will require providing details of:

- The entities in the receiving side (RecSide). The notation for identifying one of the different alternatives is as specified in clause 7.3.
- The ERDS Dispatch instance, which also carries an ERDS Evidence (XML_SUB_ACC). The notation for completely defining one specific instance among all the possibilities offered by the different parameters, is as specified in clause 6.5.

- The ERDS Receipt, carrying a XML_REL_ACC ERDS evidence. The notation for completely defining one specific instance among all the possibilities offered by the different parameters, is as specified in clause 6.4.
- The additional requirements, whose notation has been specified in clause 7.2.

Scenarios where there is some failure also include the FailureReasonId parameter.

Some scenarios include REM payloads instead of REM dispatches. The details of the components of a REM payload are provided as the details of components of a REM dispatch.

Below follows the second part of the definition of the set of test cases:

Where:

- For RecSide: (F,G,H) is one of {(1,0,0), (3,0,0), (3,1,2),(3,1,3)}: define test cases for one recipient, several recipients, and several recipient's delegate.
- For ERDS_dispatchInst_WITH_XML_SUB_ACC and REMS_receipt_WITH_XML_REL_ACC:
 - I is one of {1..8}, AssLevelComb is one of {nil, low/low, substantial/substantial, high/high}, and ConsignmentModeId is one of {nil, basic, consented, consentedSigned.
- For AdditionalReqs: P is one of {1,2}.

This part shows the different values that the parameters present in the first part, can have.

Each legal combination of all the parameters will collapse the set in ONE test case. For instance:

```
RecSide(1,0,0),
ERDS_dispatchInst_WITH_XML_SUB_ACC
      (OuterMostHeader(AS4#1, RelayMet#8, high/high, consentedSigned), UserContent,XML_SUB_ACC)),
ERDS_receipt_WITH_XML_REL_ACC
      (OuterMostHeader(RFCFields#1, NewFields#6, nil, nil), XML_REL_ACC),
AdditionalReqs#1
```

Defines ONE test case in the set, where:

- The REM-RelayDate and REM-ScheduledDelivery header fields are absent in the outermost header of the REM Dispatch and the REMS Receipt (NewFields#11 combination of new header fields).
- The assurance level combination indication is present and its value is high/high.
- The consignment mode indication is present and its value is consentedSigned.
- There is only one recipient at receiving side.

In addition, the following set of rules govern the selection of coherent triplets (RelayMet#I, AssLevelComb, ConsignmentModeId):

- 1) Absence of Assurance levels indication is indicated by a nil value.
- 2) Absence of Consignment mode indication is indicated by a nil value.
- 3) For I one of {1, 6} both AssLevelComb and ConsignmentModeId are nil.
- 4) For I one of {2, 4, 7} AssLevelComb is not nil, and ConsignmentModeId is nil.
- 5) For I one of {3, 5, 8} neither AssLevelComb nor ConsignmentModeId are nil.

Any combination (RelayMet#I, AssLevelComb, ConsignmentModeId)in a specific test case has to meet the rules 3 to 5.

8.1.4 Reasons for Failures

The present clause shows the different values that may have the reason for failure parameters in those scenarios where failures occur:

- For XML_SUB_REJ, the FailureReasonId is one of {RA02, RA03, RA04, RA05, RA06} as specified in clause 8.3.3.1 of ETSI EN 319 522-2 [2].
- For XML_REL_REJ, the FailureReasonId is one of {RB02, RB03, RB04, RB05, RB06, RB07, RB08, RB09, RB010, RB21, RB22} as specified in clause 8.3.3.2 of ETSI EN 319 522-2 [2].
- For XML_CONS_FAIL, and XML_CONS_NOT_FAIL, the FailureReasonId is one of {RD03, RD04, RD05, RD06, RD21, RD32, RD34} as specified in clause 8.3.3.4 of ETSI EN 319 522-2 [2].
- For XML_CONT_HAND_FAIL, the FailureReasonId is one of {RE03, RE04} as specified in clause 8.3.3.5 of ETSI EN 319 522-2 [2].
- For XML_CONS_REJ, the FailureReasonId is RC08 as specified in clause 8.3.3.3 of ETSI EN 319 522-2 [2].
- For XML_ACC_REJ_EXP, the FailureReasonId is RC09 as specified in clause 8.3.3.3 of ETSI EN 319 522-2 [2].

8.2 Test cases for black box model

For each scenario the present document defines the set of test cases defined by the following expression:

```
SCENARIOID (
    RecSide(F,G,H),
    AssLevelComb,ConsignmentModelId,
    AdditionalRegs#P,
    FailureReasonId
)
```

Where:

- For RecSide: (F,G,H) is one of {(1,0,0), (3,0,0), (3,1,2),(3,1,3)}: define test cases for one recipient, several recipients, and several recipient's delegate.
- For AdditionalReqs: P is one of {1,2}.
- For FailureReasonId: see clause 8.1.4.

8.3 Test cases for 4-cornel and extended models

8.3.1 General

The present clause, instead of showing one test case set expression for each scenario, which would make the present document very long, defines rules for building these expressions for each scenario in function of: the ERD dispatches, ERDS receipts, ERD payloads present in them, the entities at the receiving side, the failure reasons, and the additional requirements.

Some of the rules are defined for parametrizing the different ERD messages appearing in each scenario, other for parametrizing the receiving side, others for parametrizing the failure reasons, and others for parametrizing the additional requirements. The application of all these rules for one scenario results in generating test cases set expressions as the one shown in clause 8.1.3 which define a set of test cases for such an scenario.

8.3.2 Rules for parametrizing ERD dispatches

The ERD dispatches for these scenarios will be built on the following components:

(Metadata(AS4Met#1, RelayMet#1, AssLevelComb, ConsignmentModeId), UserContent,<EVID>+)

Where:

• i is one of {1,2,3,4,5,6,7,8}, AssLevelComb is one of {nil, low/low, substantial/substantial, high/high}, ConsignmentModeId is one of {nil, basic, consented, consentedSigned},and <EVID>+ stands for the ERDS evidences present in the ERD dispatch as per the scenario.

94

8.3.3 Rules for parametrizing ERD payloads

The ERD payloads for these scenarios will be built on the following components:

(Metadata(AS4Met#1, RelayMet#I, AssLevelComb, ConsignmentModeId), UserContent

Where:

• i is one of {1,2,3,4,5,6,7,8}, AssLevelComb is one of {nil, low/low, substantial/substantial, high/high}, and ConsignmentModeId is one of {nil, basic, consented, consentedSigned}.

8.3.4 Rules for parametrizing ERDS receipts

The ERDS receipts for these scenarios will be built on the following components:

(Metadata(AS4Met#1, RelayMet#1, AssLevelComb, nil), <EVID>+)

Where:

• i is one of {1,2,3,4,5,6,7,8}, AssLevelComb is one of {nil, low/low, substantial/substantial, high/high}, and <EVID>+ stands for the ERDS evidences present in the ERDS receipt as per the scenario.

8.3.5 Rules for parametrizing entities at receiving side

The number and type of entities present at the receiving side will be parametrized as indicated below:

RecSide(F,G,H),

Where, as specified in clause 7.3:

- F stands for the number of recipients.
- G stands for the number of recipient's delegates.
- H stands for the number of recipients each delegate is delegate of.

H shall always be less or equal than F. If F is not 0 then H shall also be different from 0.

8.3.6 Rules for parametrizing failure reasons

The FailureReasonId shall follow the requirements specified in clause 8.1.4.

8.3.7 Rules for parametrizing additional requirements

The additional requirements will be parametrized as indicated below:

AdditionalReqs#P

Where P is one of $\{1,2\}$ with the meaning specified in clause 7.2.

- ETSI EN 319 532-1: "Electronic Signatures and Infrastructures (ESI); Registered Electronic Mail (REM) Services; Part 1: Framework and Architecture".
- ETSI EN 319 532-2: "Electronic Signatures and Infrastructures (ESI); Registered Electronic Mail (REM) Services; Part 2: Semantic Contents".
- ETSI EN 319 532-4: "Electronic Signatures and Infrastructures (ESI); Registered Electronic Mail (REM) Services; Part 4: Interoperability profiles".

Annex B (informative): Change History

Date	Version	Information about changes
February 2019	1.1.1	Publication as ETSI TS 119 524-1.
April 2023	1.1.2	Early draft - update version 1.1.1 with resolutions to part of the comments received from stakeholders on version 1.1.1
October 2023	1.1.3	Final draft for TS Approval. This version includes the implementation of resolutions to all the comments received to versions v 1.1.1 and v1.1.2. Of important relevance are test cases that include the generation and management of the three new events and their corresponding ERDS evidence added in 319 522-1 clause 6, namely: NotificationDelivered (C.6), NotificationAccessTracking (D.5), and ContentAccessTracking (D.6)
October 2023	1.1.4	Final draft for TS Approval. This version includes the right WI; defines the acronym NOT_F_ACC_FAIL; merges empty cells in the last columns of some scenario tables; and makes that each scenario table starts at the beginning of one page.

History

Document history			
V1.1.1	February 2019	Publication	
V1.2.1	December 2023	Publication	

97