

GREEN

Only share inside EESSI community
See Author guidance before sharing

EESSI - RINA

Forms Design Specifications

Contents

Document Revision.....	3
1 Document Information and Audience	5
2 Business Purpose	5
3 RINA Forms Design Specifications.....	5
3.1 Artefacts consumed directly by the RINA portal.....	5
3.1.1 Dynamic Forms Metadata artefacts	6
3.1.2 Translations resources artefacts	25
3.1.3 eUI Dynamic Forms presentation mode	25
3.1.4 Integration of Dynamic Forms inside in the RINA portal.....	26
3.2 Artefacts used indirectly by the RINA portal eUI Dynamic forms	28
3.2.1 SED XSDs	28
3.2.2 SED draft XSDs.....	28
3.2.3 Initial JSONs (empty SED JSONs files)	28

Document Control Information

Document Control	Value
Project Title	Electronic Exchange of Social Security Information (EESSI)
Document Name	EESSI – RINA – Forms Design
Document Category	Architecture & Design Specifications
Revision	-
Component Version	-
Last Publication Date Project Milestone	18/12/2020 EESSI-2020
Document Status	Final
Sensitivity (TLP) Distribution terms	Traffic Light Protocol (TLP) = "GREEN"  The distribution of this document is done strictly in line with the Traffic Light Protocol (TLP) established by the European Commission's note AC 790/15 REV for the EESSI project documentation. In line with the note AC 790/15 REV, this document is labelled as TLP = "Green". Therefore, it can be circulated widely within the EESSI community. However, the document or the information herein may not be published or posted on the Internet, nor released outside of the EESSI community.
Connected/Embedded Files	None
Authors	European Commission, DG EMPL F5, EESSI RINA
Revised by	European Commission, DG EMPL F5, EESSI QA/QC
Approved by	European Commission, DG EMPL F5, EESSI PMO team

Document History

Project Milestone	Changes/Corrections Description
EESSI-2020	Initial Document

1 Document Information and Audience

Purpose:

To detail the requirements that define the business solution. This document is used to gain agreement with stakeholders and to provide a foundation to communicate to a technology service provider what the solution needs to do to satisfy the customer's and business' needs.

Audience:

- All project team members
- Project Sponsor, Business Owner, and IT/Technical Owner
- All other key stakeholders

2 Business Purpose

The purpose of this document is to cover the main planned features as an extension of the current system. The main objective is to highlight:

- What is the business needs?
- How it should be implemented?

List of features to be covered in this document:

Enhancement (s)	Priority	Impact
RINA Forms Design	Must	Low

3 RINA Forms Design Specifications

RINA portal uses eUI Dynamic Forms libraries (developed by DIGIT) to display and interact with the SED (Structured Electronic Document) forms. eUI Dynamic forms libraries are supported by DIGIT (the link to the eUI platform is <https://eui.ecdevops.eu/>).

eUI Dynamic Forms uses metadata and translations files to generate the SED forms.

These artefacts can be categorized into the following 2 types:

- Artefacts that are directly consumed by the RINA portal: these are the eUI dynamic forms metadata and translations used to generate at runtime the forms for capturing data in RINA portal
- Artefacts that are used indirectly by the RINA portal dynamic forms: initial JSON files that represent the data containers for the SEDs, XSD files that will be used to validate the SED XML payload that will be transmitted

3.1 Artefacts consumed directly by the RINA portal

There are 2 main artefacts that are consumed by RINA Portal to generate the eUI Dynamic Forms :

- Dynamic Forms metadata
- Translations related with labels, explanatory notes and vocabularies

The source of the dynamic forms metadata, as mentioned before, is:

- The CDM model – the original model which is maintained by the Commission to represent how the SED documents must be built and populated with data, in order to be later exchanged between the Member States.

3.1.1 Dynamic Forms Metadata artefacts

The dynamic forms use the visual style based on European Commission's User Experience (UX@EC) requirements as implemented by the eUI libraries. Below it is presented an example of the implementation of eUI Dynamic Forms into RINA Portal.

P2000 - Old age pension claim SED API version: 8.18.3 build preview 1
Model version: 4.2.0

1. LOCAL CASE NUMBERS

1.1(1). LOCAL CASE NUMBER

1.1.1. Country *

1.1.2. Case number *

1.1.3. INSTITUTION ⓘ

1.1.3.1. Institution ID ⓘ

1.1.3.2. Institution Name *

1.1(1). Local Case Number: Add ✕

2. INSURED PERSON *

2.1. PERSON IDENTIFICATION *

2.1.1. Family name(s) *

Figure 1 – RINA Dynamic form example

The dynamic forms metamodel describes the structure of the visual form and it is used by eUI libraries to render dynamically at runtime the form. The goal is to obtain at runtime a form to be used by the users to correctly fill in information for SED documents. The eUI Metamodel contains information about fields, the types of fields and what values they are expecting, as well as what restrictions are for filling in those values.

3.1.1.1 Sections and Fields

Sections and Fields are the two key elements that are used to construct a SED. Understanding these concepts is vital to the understanding of how these fields work in the eUI dynamic forms from RINA Portal.

A **Field** is a data element that a clerk will complete. A Field will always have a specific data type which specifies what kind of data should be filled in. For example the data entered into the field must be text (the field has a data type "string") or the data entered into the field must be a date (the field has a data type called "date").

2.1.1. Family name(s) *

Figure 2 – Example of fields in a SED

A section is a group of fields and/or sections.

When a section is used inside a section it is often called a nested section, with the nested section being called a 'child' of the 'parent' section. Every section has a header or a title. Unlike a field, a section does not have a primitive data type.

Here is an example of a section that contains only fields.

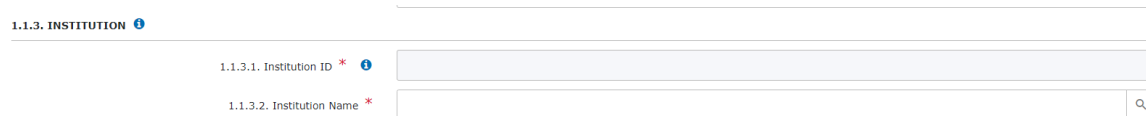


Figure 3 – Example of a Section in a SED

Here is an example of a section that contains fields and another section.

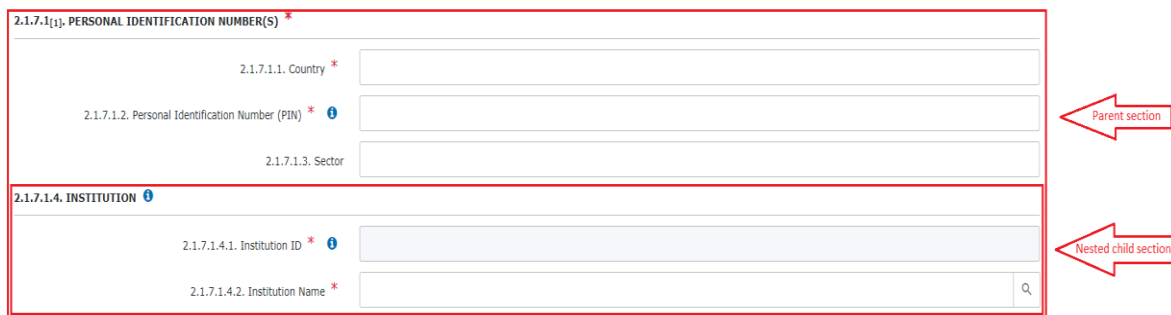


Figure 4 – Example of a nested Section

In this example, the section is called "Personal Identification Number(s)". This section contains three fields ("Country", "PIN" and "Sector") and a nested section ("Institution"). The section "Institution" also contains two fields ("Institution ID" and "Institution Name"). In order to identify which elements are sections and which are only fields you have to look carefully at the numbering of the element. For example in this case you can identify that the "Institution ID" and "Institution Name" are fields within the section Institution because they have a lower level granularity of their numbering under Institution than the other fields above such as "Country", "PIN" or "Sector". In other words, a SED is like a tree.

Nesting of sections can occur infinitely. In this example, we can see multiple nested child sections.

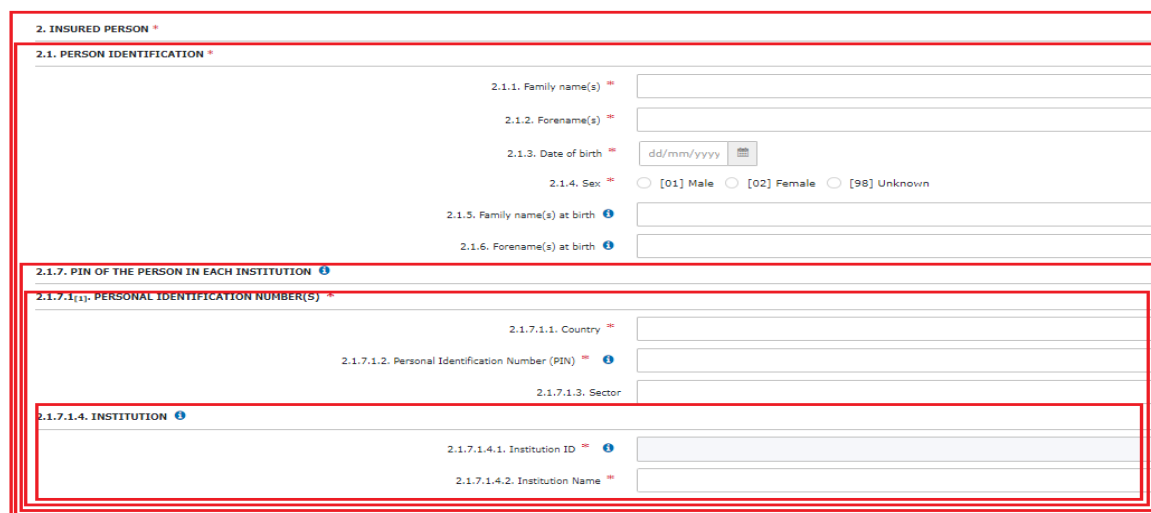


Figure 5 – Example of four nested sections inside the Section 'Insured Person'

The form typically strips away empty fields and empty sections, given they are not mandatory. This is done having in mind the principle that the saved document should not contain redundant information. If the empty fields and sections are mandatory, the appropriate validation errors are triggered (see chapter 3.1.1.10 on Form validation).

3.1.1.2 Section Repetitiveness

The repetitive sections will appear in the form accompanied by a section footer with "Add" and "x" (i.e. remove) buttons for insert and remove respectively. See the next image:



Figure 6 – Section repetition

The repetitive sections have the following behaviour:

- When "Add" button is used, a new empty section is added immediately after the the button. It acts as an "add after" functionality
- The above point means that it is not possible to inject a new instance of the repeatable section before the first instance.
- Empty instances will result in validation offences if and only if the section (the entire group of repeatable instances) is "effectively required". See the **Form validation** section for details.
- Using "x" will remove the section instance placed above the button.
- It is not possible to remove the last repeatable instance in the group.

3.1.1.3 Section and Subsections Alignment

The design of the sections and sub sections of the RINA forms is governed by eUI User Experience guidelines.

- All sections must be highlighted.
- All fields are aligned to the right to be close to the input where their value is expected to be filled in by the user.

These visual enhancements are necessary to ease the work with long forms.

For an example see Figure 7.

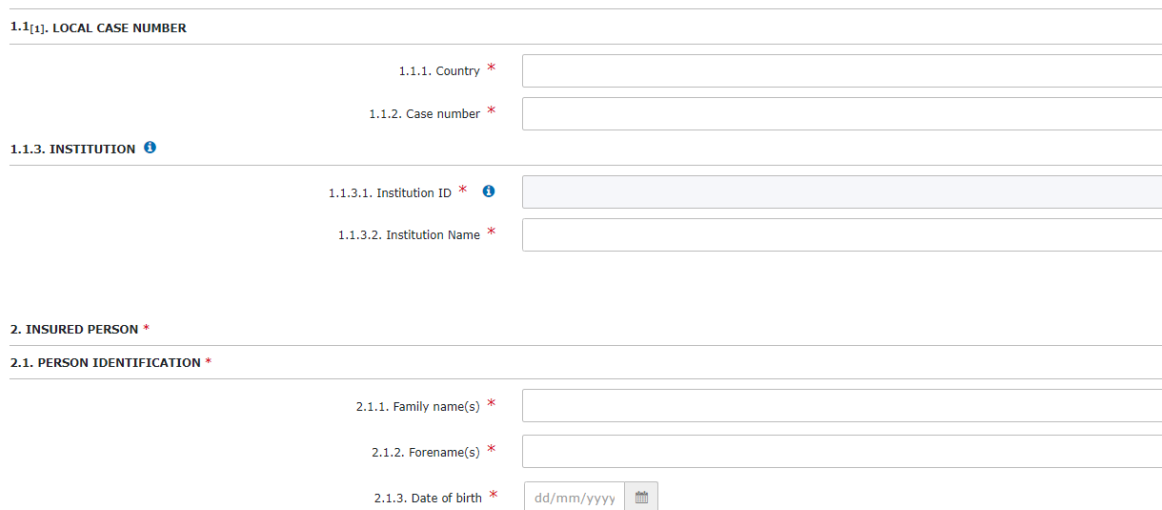


Figure 7 – Section/Sub-section alignment

3.1.1.4 Indexing Repetitive Sections

The indexing method of repetitive sections from RINA forms was implemented according to the following rules:

- The repetitive sections are indexed using a counter within square brackets: [**<counter>**]. The counter will indicate the number of the entry index in the repetitive section. The index is **subscripted** relative to the section to which it belongs to (e.g. 1.1.[1].1 Country).

Note: Each repetitive section must have its own counter, since there might be repetitive sections within repetitive sections, (e.g. 1.3.[1].1.[2].).

The counters for repetitive sections are visible only on the header of the repetitive section, but not in the labels of the items inside the repetitive section.

For an example see Figure 8.

P2000 - Old age pension claim

1. LOCAL CASE NUMBERS

1.1_[1]. LOCAL CASE NUMBER

1.1.1. Country *

1.1.2. Case number *

1.1.3. INSTITUTION ⓘ

1.1.3.1. Institution ID ⓘ

1.1.3.2. Institution Name *

1.1_[2]. LOCAL CASE NUMBER

1.1.1. Country *

1.1.2. Case number *

1.1.3. INSTITUTION ⓘ

1.1.3.1. Institution ID ⓘ

1.1.3.2. Institution Name *

Figure 8 – Indexing of repetitions

3.1.1.5 Choices

The concept of a "choice" represents a group of sibling elements that are mutually exclusive.

The RINA dynamic forms implement this concept using selection radio-buttons for each of the mutually exclusive elements and only one of the elements can be selected.

In this way, only the selected element will be visible/used in the form.

For an example see Figure 9a and Figure 9b.

8.2. BANK INFORMATION ⓘ

8.2.1. Account holder name

* ⓘ ☒ SEPA Account ☐ Non-SEPA Account

8.2.2.1. IBAN * ⓘ

8.2.2.2. BIC-SWIFT ⓘ

Figure 9a – First choice element selected

8.2. BANK INFORMATION ⓘ

* ⓘ ☐ SEPA Account
☒ Non-SEPA Account

8.2.1. Account holder name

8.2.3.1. Account Number *

8.2.3.2. BIC-SWIFT ⓘ

8.2.3.3. Bank Name *

8.2.3.4. BANK ADDRESS ⓘ *

8.2.3.4.1. Street ⓘ

8.2.3.4.2. Building Name ⓘ

8.2.3.4.3. Town *

8.2.3.4.4. Postal Code ⓘ

8.2.3.4.5. Region ⓘ

8.2.3.4.6. Country *

Figure 9b – Second choice element selected

3.1.1.6 Input fields

Primitive or basic fields

The primitive/basic input fields are of type: string, integer, double. They appear as a text-box input fields in the dynamic forms.

The form-validation for these fields covers the following aspects:

- mandatory-ness check
- data format check (integer – the value should be an integer number, double – the value should be a floating-point number with or without decimal places, etc.)
- restrictions check for the following parameters:
 - pattern: a regular expression (regex) pattern can be assigned to a field and the content of the field is verified against it
 - minLength/maxLength: character string length restrictions
 - minInclusive/maxInclusive: numeric value restrictions

For examples see Figures 10a, 10b, 10c and 10d.

2.1. PERSON IDENTIFICATION *

2.1.1. Family name(s) *

Family name

2.1.2. Forename(s) *

Forename

Figure 10a – String input fields

2. INSURED PERSON *

2.1. PERSON IDENTIFICATION *

2.1.1. Family name(s) *

aaa

Maximum length is 155 You have entered 176

2.1.2. Forename(s) *

bbbbbbbbbbbbbbbbbbbbbb

Figure 10b – String input field that violates maxLength restriction

2.2.4.2. EMAIL ADDRESSES

2.2.4.2.1_[1]. EMAIL ADDRESS

2.2.4.2.1.1. Email Address

ssssssssssssssssssssssssssssssss

Pattern should be matched: ^([w\-.]+)@(([w]+\.)+)([a-zA-Z]{2,15})\$

Figure 10c – String input field that violates pattern restriction for email address format

2.3. TOTAL AMOUNT REQUESTED ⓘ *

2.3.1. Amount *

1a

Please use a double value

Figure 10d – Integer input field that does not respect the integer data format

Boolean fields

A Boolean field is modelled as a Yes/No enumeration.

Note: validation of Boolean fields' mandatory-ness requires some considerations and introduces the problem of detection if a boolean field represents a false value or a none selected value; because of this in the current data-model the boolean fields as checkboxes have been replaced with 'Yes/No' single-select enumerations.

5. DECISION ⓘ

- ☒ Decision regarding all children mentioned in this SED
☐ Decision which does not concern all children mentioned in this SED

5.1. Decision regarding all children mentioned in this SED

☒ [1] Yes ☐ [0] No

Figure 11 – Yes/No enumerations

Date fields

Date-fields use the eUI date picker widget. An example can be seen below:

2. RECEIPT OF APPLICATION FOR FAMILY BENEFITS

2.1. Date of receipt of the application  

2.2[1]. REQUEST FOR PERIOD OF INFORMATION

☒ Fixed Period  

☐ Open Period

2.2.1.1. Start date  

2.2.1.2. End date  

17/12/2020  

Su	Mo	Tu	We	Th	Fr	Sa
DEC						
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

Figure 12 – Date field

The date selection can be done as following:

- When the user clicks the mouse on the calendar glyph button, the day-selector will pop-up first.
- After opening the day-selector popup, the user can change to the next month or the previous month by using the arrows in the top-right corner.
- To change the year, the user can click the date displayed in the top-left corner of the day-selector popup; this will trigger the year-selector to appear. From this moment, the date will be selected in this order: year > month > day. That means that when the year-selector is displayed, the user can click the year to select it, and the month-selector will appear automatically; after the user clicks on a month, the day-selector will appear; after the user clicks on a day, the popup closes, and the selected date is filled in the field.
- The currently displayed selector can be dismissed by using the <Esc> key, or by placing the focus in a different form field.
- The date can be cleared from the field by using the 'X' icon displayed on the right of the input, before the calendar icon button

If a date is already set, the selectors reflect the current date accordingly (the year selector will show the year and so on).

Enumerations

The data-model defines enumerations attributes and configures their multiplicity attribute for single-value or multiple-value selection. Depending on the multiplicity and on the number of enum-codes, the enumeration attributes appear differently on the form:

- Single-select value enumerations (multiplicity 0 or 1) with 5 enum-codes or less appear as radio-buttons.
- Single-select (multiplicity 0 or 1) value enumerations with more than 5 enum-codes appear as a single select drop-down with type ahead.
- Multiple-select values enumerations (multiplicity unbounded) with 5 enum-codes or less appear as check-box buttons.

- Multiple-select values enumerations (multiplicity unbounded) with more than 5 enum-codes appear as a multiple-select widget, with image-per-item capabilities.

2.3. Type of claim ☐ [01] New claim ☐ [02] Change in circumstances ← Single-select enumerations
5 enum-codes or less

2.4. Information on application i ☐ [01] We confirm the provided information ← Multiple-select enumerations
5 enum-codes or less
☐ [02] Provide us with information in points

2.5. Points for which more information is needed i

3. INFORMATION ABOUT CHANGE IN CIRCUMSTANCES i

3.1. Change of Circumstance for ☐ [01] Entire Family
☐ [02] Claimant
☐ [03] Spouse/Partner ← Multiple-select enumerations
5 enum-codes or less
☐ [04] Other person
☐ [05] Child(ren)

1. PURPOSE OF THIS SED

1.1. Purpose of this SED i

☒ Select All ☐ Select None ← Multiple-select enumerations
More than 5 enum-codes

6.1.7.1.1. Country * ← Single-select enumerations
More than 5 enum-codes

Figure 13 – The different types of enumerations

Note: A multiple-selection enumeration should result in multiple values also with multiple value fields in the JSON/XML. See the next images where field2 and field4 have multiple values that generate in the JSON/XML multiple value entries for the enumeration.

```
{
  "SED4": {
    "sedPackage": "SEDs",
    "sedGVer": "0",
    "sedVer": "0",
    "field1": {
      "value": [ "S" ]
    },
    "section1": {
      "fieldB": {
        "value": [ "AT" ]
      }
    },
    "field2": {
      "value": [ "R", "G" ]
    },
    "field3": {
      "value": [ "BE" ]
    },
    "field4": {
      "value": [ "AT", "BE" ]
    }
  }
}
```

Figure 14 – JSON file content for the enumerations example SED

```
<?xml version="1.0" encoding="UTF-8"?>
<sedSED4:SED4 sedPackage="SEDs" sedGVer="0" sedVer="0" xmlns:sedSED4="http://ec.europa.eu/eessi/ns/0_0/SED4">
  <field1>
    <value>S</value>
  </field1>
  <section1>
    <fieldB>
      <value>AT</value>
    </fieldB>
  </section1>
  <field2>
    <value>R</value>
    <value>G</value>
  </field2>
  <field3>
    <value>BE</value>
  </field3>
  <field4>
    <value>AT</value>
    <value>BE</value>
  </field4>
</sedSED4:SED4>
```

Figure 15 – XML file content for the enumerations example SED

Read-Only

A field can be marked as read only.

6.1.7.1.4. INSTITUTION ⓘ

6.1.7.1.4.1. Institution ID * ⓘ

6.1.7.1.4.2. Institution Name *

Figure 16 – Read only field in the dynamic form

Fixed Value

A field can be defined as having a fixed value.

5. DECISION ⓘ

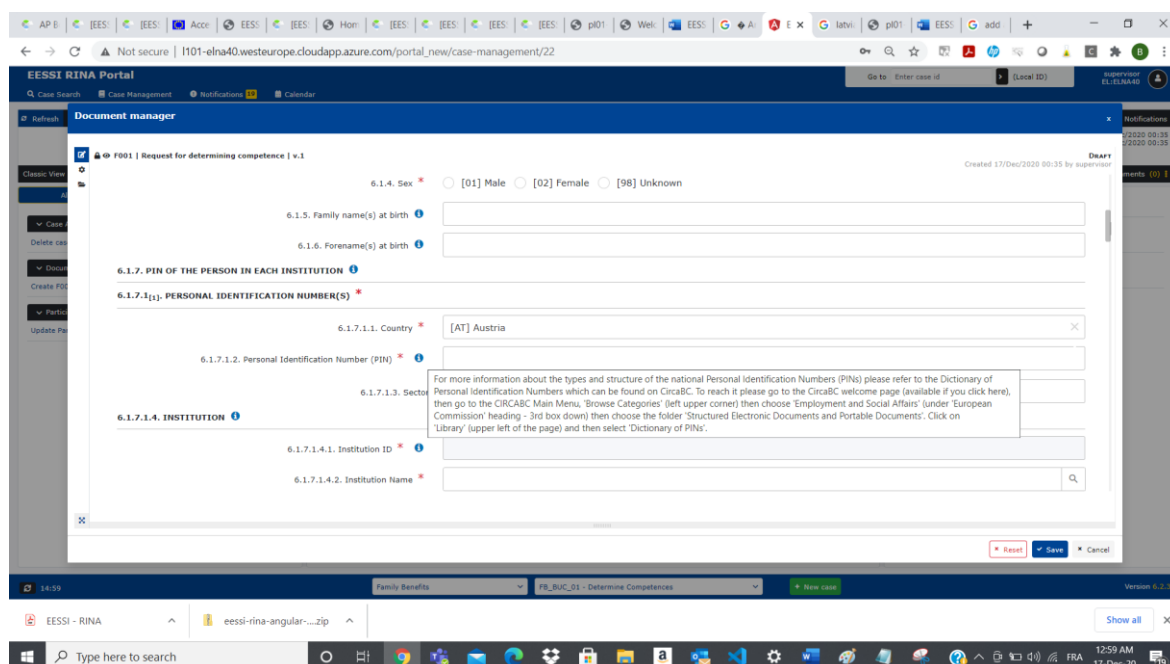
- ☒ Decision regarding all children mentioned in this SED
- ☐ Decision which does not concern all children mentioned in this SED

5.1. Decision regarding all children mentioned in this SED ☒ [1] Yes ☐ [0] No

Figure 17 – Fixed value for a field in the eUI dynamic form in RINA Portal

3.1.1.7 Explanatory Notes

Sections/sub-sections and fields can have attached to them explanatory notes in the data-model by having on that element the tag *xnote* with the corresponding text value. In the eUI dynamic forms, an image icon is placed on the right of the section/sub-section name or the field. Then, when clicked, the icon will display a tooltip with the text of the explanatory note.



The screenshot shows the EESSI RINA Portal interface. The main form is titled 'Request for determining competence' and contains several fields with asterisks indicating they are mandatory. An explanatory note is displayed for the '6.1.7.1.3. Sector' field, stating: 'For more information about the types and structure of the national Personal Identification Numbers (PINs) please refer to the Dictionary of Personal Identification Numbers which can be found on CircaBC. To reach it please go to the CircaBC welcome page (available if you click here), then go to the CIRCA BC Main Menu, 'Browse Categories' (left upper corner) then choose 'Employment and Social Affairs' (under 'European Commission' heading - 3rd box down) then choose the folder 'Structured Electronic Documents and Portable Documents'. Click on 'Library' (upper left of the page) and then select 'Dictionary of PINs'.

Figure 18 – Explanatory note displayed for a field in the eUI dynamic form in RINA Portal

3.1.1.8 Mandatory/Optional check details

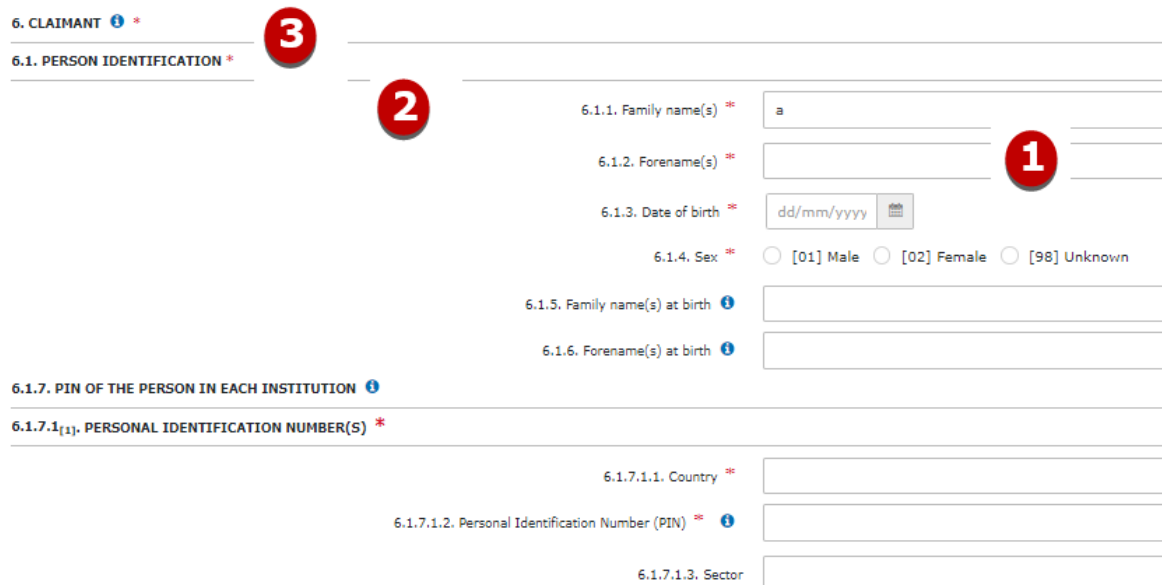
Mandatory or Optional is an attribute that applies to both Fields and Sections. This means that a field can either be Mandatory or Optional and a section can also be Mandatory or Optional. In EESSI, mandatory fields and sections are marked with an asterisk (*) symbol.

The way in which sections and fields interact with each other means that there are two ways in which fields can be mandatory, these are either fixed mandatory or conditional mandatory.

Fields which are always mandatory (Fixed Mandatory fields)

Some fields are always mandatory meaning that they always **MUST** be filled in. This applies to any field that is marked as mandatory **AND** its section is also Mandatory.

If the section where the field is located is nested inside another section or in case of multiple nested sections, then the field is always mandatory only if all of its above parent sections are also mandatory. If any of the parent sections is not marked with the "*" sign, then the field is not always mandatory but could become mandatory under certain conditions as explained in the next section.



6. CLAIMANT ⓘ *

6.1. PERSON IDENTIFICATION *

6.1.1. Family name(s) * a

6.1.2. Forename(s) *

6.1.3. Date of birth * dd/mm/yyyy

6.1.4. Sex * ☐ [01] Male ☐ [02] Female ☐ [98] Unknown

6.1.5. Family name(s) at birth ⓘ

6.1.6. Forename(s) at birth ⓘ

6.1.7. PIN OF THE PERSON IN EACH INSTITUTION ⓘ

6.1.7.1_[1]. PERSONAL IDENTIFICATION NUMBER(S) *

6.1.7.1.1. Country *

6.1.7.1.2. Personal Identification Number (PIN) ⓘ

6.1.7.1.3. Sector

Figure 19 – Example of a fixed mandatory field

1	Forename(s) is a mandatory field denoted by the "*" sign
2	Forename(s) is part of the section 'Person Identification'. This section is mandatory as denoted by the "*" sign
3	'Person Identification' section is nested inside the 'Claimant' Section. This section is mandatory as denoted by the "*" sign. As there are no more sections then we know in this SED there must be: - A 'Claimant' section - The 'Claimant' section must contain a 'Person Identification' section - The 'Person Identification' section must contain the completed field 'Forename(s)'

In the next image we have an example of a field which is marked as mandatory (so marked with the "*" sign) but in fact is optional (and can become mandatory under certain conditions explained in the next section)

6. CLAIMANT ⓘ *

6.1. PERSON IDENTIFICATION *

6.1.1. Family name(s) * a

6.1.2. Forename(s) *

6.1.3. Date of birth * dd/mm/yyyy

6.1.4. Sex * ☐ [01] Male ☐ [02] Female ☐ [98] Unknown

6.1.5. Family name(s) at birth ⓘ

6.1.6. Forename(s) at birth ⓘ

6.1.7. PIN OF THE PERSON IN EACH INSTITUTION ⓘ **3**

6.1.7.1. PERSONAL IDENTIFICATION NUMBER(S) *

6.1.7.1.1. Country * **2**

6.1.7.1.2. Personal Identification Number (PIN) * ⓘ **1**

6.1.7.1.3. Sector

6.1.7.1.4. INSTITUTION ⓘ

6.1.7.1.4.1. Institution ID * ⓘ

6.1.7.1.4.2. Institution Name *

Figure 20 – Example of a field which is marked as mandatory but in fact is optional

1	Person Identification Number is marked with the "*" sign
2	Person Identification Number is part of the section 'Person Identification Number(s)'. This section is marked as mandatory with the "*" sign
3	'Person Identification Number(s)' section is nested inside the 'PIN of the person in each Institution' Section. This section is optional as there is no "*" sign. As a result of this section being optional, we know that if we don't require the 'PIN of the person in each Institution' section, therefore we don't have to have a 'Personal Identification Number(s)' section and therefore we don't have to have to fill in the 'Person Identification Number' field which is thus not mandatory in this particular case.

Fields that are mandatory depending on context (Conditional Mandatory fields)

In certain cases, a field that will be marked with the "*" sign will in fact not always be mandatory and can be in fact optional, depending on the specific circumstances of the SED in question. We call this 'conditionally mandatory', since the field becomes mandatory under certain conditions.

In these cases it may seem like the * sign is pointless; however it does serve an important purpose, explained below.

For example the P2000 SEDs is about an insured person. In this SED, we also need to know about the person's children. However it is clear that not all persons have children,

so the section about children can only be optional. But IF a person has children (in other words the sub-section about a child has data) THEN we would need to know some information about them and fields such as Forename, Last Name and Date of Birth become mandatory!

Once a field from a section that contains conditional mandatory fields is filled in (in other words the section containing that field has data), ALL the other fields from the section which are conditional mandatory become mandatory (in other words all the fields which are marked with the "*" sign under an optional section are mandatory once one of the fields is filled in).

If the example from above is re-examined, if a value has been added in the 'Sector' field of the Person Identification Number(s) section, then all the fields marked with the "*" sign become mandatory.

6. CLAIMANT ⓘ *

6.1. PERSON IDENTIFICATION *

6.1.1. Family name(s) *

6.1.2. Forename(s) *

6.1.3. Date of birth * ⓘ

6.1.4. Sex * ☐ [01] Male ☐ [02] Female ☐ [98] Unknown

6.1.5. Family name(s) at birth ⓘ

6.1.6. Forename(s) at birth ⓘ

6.1.7. PIN OF THE PERSON IN EACH INSTITUTION ⓘ

6.1.7.1_[1]. PERSONAL IDENTIFICATION NUMBER(S) *

6.1.7.1.1. Country * ⓘ

6.1.7.1.2. Personal Identification Number (PIN) * ⓘ

6.1.7.1.3. Sector ⓘ

6.1.7.1.4. INSTITUTION ⓘ

6.1.7.1.4.1. Institution ID * ⓘ

6.1.7.1.4.2. Institution Name *

Figure 21 – Example of Conditional mandatory fields

1	The 'Sector' field is filled in
2	As the 'Sector' Field in filled in we have a 'Personal Identification Number' section
3	The 'PIN of the person in each institution' section is not marked as mandatory but since it has now certain fields of it being filled in (under the 'Personal Identification Number' section), the section becomes relevant in this context
4	Now everything marked as mandatory under section Personal Identification Number becomes mandatory, therefore 'Country' and 'Personal Identification Number' is now mandatory.

When a SED is considered in such as way the following simple decision flow chart can be used to determine if any field is Mandatory or Optional.

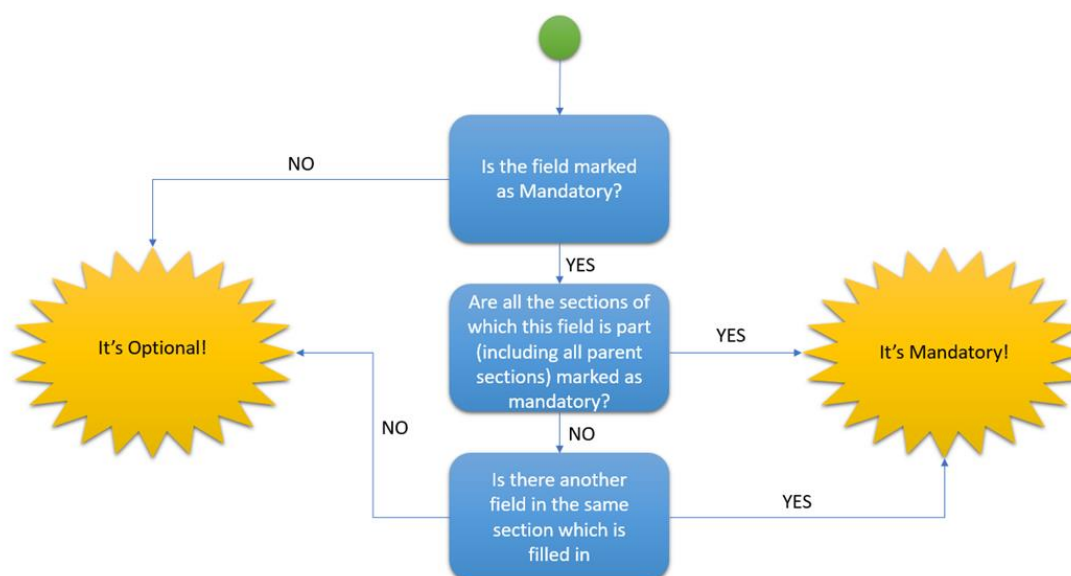


Figure 22 – Decision flow chart to determine mandatory-ness

3.1.1.9 Draft critical

There are SEDs that can be very large and can take a long time to complete by the users. Because of this, in RINA there is the possibility to save a draft of the SED. However, even for a draft version, in order to be able to save it and later retrieve it to continue the completion, certain fields need to be completed. These fields are the 'draft critical' fields.

In the data-model these draft critical fields will be marked with the *draftCritical* tag. As a result, in the eUI metamodel, those fields will be marked accordingly. When the validation occurs for a document in the 'draft' state, the draft critical fields will be marked with critical errors (coloured in red) and the other mandatory fields that are not draft critical will be marked only with warnings (coloured with orange).

This is the first level of draft critical check that is done at the level of the eUI dynamic forms. Another level of check is performed in RINA using the 'save draft' XML payload against the XSD draft files (see the **SED draft XSDs** section below).

6. CLAIMANT ⓘ *

6.1. PERSON IDENTIFICATION *

6.1.1. Family name(s) *

requires data

6.1.2. Forename(s) *

requires data

6.1.3. Date of birth * 

requires data

6.1.4. Sex * ☐ [01] Male ☐ [02] Female ☐ [98] Unknown

requires data

Annotations in the form:

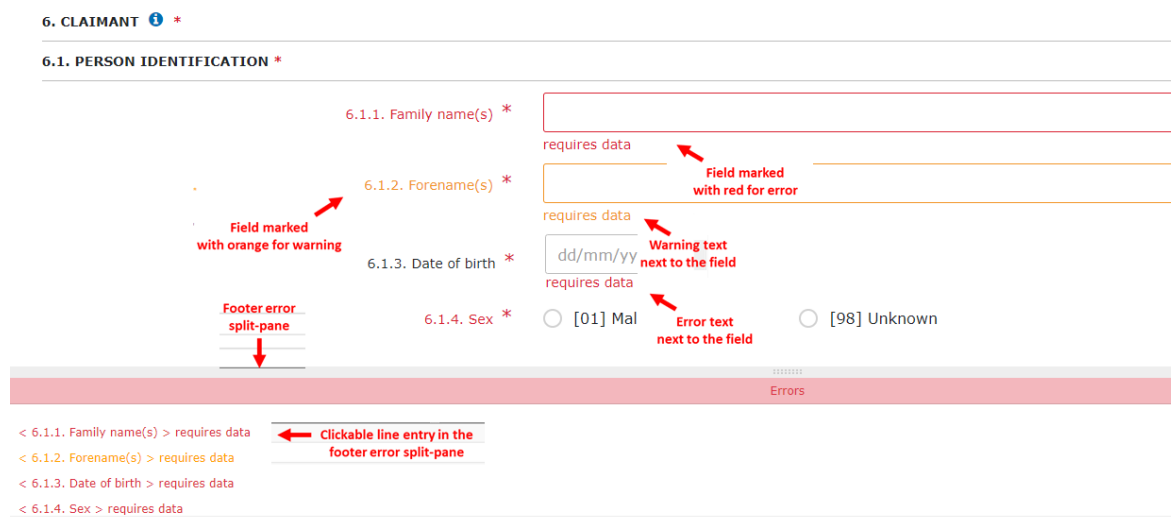
- Red arrow pointing to the Family name(s) field: **Draft Critical field marked with error**
- Orange arrow pointing to the Forename(s) field: **Non Draft Critical field marked with warning**

Figure 23 – Draft critical error display

3.1.1.10 Form validation and error display

The validation approach is inspired from Angular examples: it validates the entire content and is marking/displaying the validation offences in the following parts of the form:

- Colour marking (red for errors, orange for warnings) of the fields and of the corresponding text that explains the error/warning
- Next to the fields that have produced them.
- In a special footer error split-pane that is displayed only when there are errors/warnings; also, each line entry from this split-pane is a clickable link that will make the form to auto-scroll to the corresponding offending field.



6. CLAIMANT ⓘ *

6.1. PERSON IDENTIFICATION *

6.1.1. Family name(s) * requires data

6.1.2. Forename(s) * requires data

6.1.3. Date of birth * dd/mm/yy requires data

6.1.4. Sex * ☐ [01] Mal ☐ [98] Unknown

Field marked with orange for warning

Field marked with red for error

Warning text next to the field

Error text next to the field

Footer error split-pane

Errors

< 6.1.1. Family name(s) > requires data

< 6.1.2. Forename(s) > requires data

< 6.1.3. Date of birth > requires data

< 6.1.4. Sex > requires data

Clickable line entry in the footer error split-pane

Figure 24 – Errors display

The form-validation covers the following aspects:

- mandatory-ness check for sections/fields
- data format check (integer – the value should be an integer number, double – the value should be a floating-point number with or without decimal places, etc.)
- restrictions check for the following parameters:
 - o pattern: a regular expression (regex) pattern can be assigned to a field and the content of the field is verified against it
 - o minLength/maxLength: character string length restrictions
 - o minInclusive/maxInclusive: numeric value restrictions

The behaviour related to validation is the following:

- Validation is performed using Angular reactive forms validators – custom validators as well as built-in validators
- Validation messages appear as (1) inline errors and (2) error messages in Error list, as shown in the previous figure.
- As the user supplies data, potential offences are addressed; if so, the offence messages can disappear if data is valid.
- Mandatory-ness check is sensitive to whether a section is *effectively required*. There is no simpler way to put this but:

- A section is *effectively required* if:
 - it has data OR
 - if it is mandatory and the parent section is effectively-required
- The SED is an *effectively required* section.
- Only fields of effectively-required sections can trigger errors due to mandatory-ness.

Sections may trigger themselves validation offences if they are effectively required but without data. These offences are also displayed in the footer error split-bane at the bottom of the form as well as inline errors.

3.1.1.11 EESSI Institution field

A special section in the dynamic forms is the Institution section that has the data-model type *EESSIIInstitutionType*. This section has two fields, Institution ID and Institution Name. Normally, these EESSI Institutions should be in the EESSI Institution Repository (IR) and - to make sure that the Institution ID is completed correctly in reference to the entries from the IR, a search widget has been implemented in RINA to function in the following way:

- In the dynamic form, the Institution ID field is disabled (read-only)
- A search icon button is now available next to the Institution Name field

There are **two ways** of selecting an Institution using this widget:

- a) Autocomplete-like search in <Institution Name> field
- b) Advanced search using the search icon button next to <Institution Name> field

These ways are detailed below:

a) Autocomplete-like search in <Institution Name> field

- When a user fills in some text in the <Institution Name> field, a API call is done to retrieve institutions which are matching with this text
- A dropdown will appear with matching institutions as suggestions
- The user can select one institution; after selecting one, the ID and the Name fields are filled in automatically with the values corresponding to the institution

P2000 - Old age pension claim SED API version: 0.16.3 build preview 1
Model version: 4.2.0

1. LOCAL CASE NUMBERS

1.1[1]. LOCAL CASE NUMBER

1.1.1. Country *	ELNA40
1.1.2. Case number *	ELNA01 EN
1.1.3. INSTITUTION ⓘ	ELNA02 EN
1.1.3.1. Institution ID * ⓘ	ELNA05 EN
1.1.3.2. Institution Name *	ELNA06 EN

elna

Figure 25 – Autocomplete for Institution selection

b) Advanced search using the search icon button next to <Institution Name> field

- When the search icon is clicked, the widget will open a Search pop-up window that will search for the institution in the IR using various criteria (country, sector, process, etc).
- If an Institution has been found in IR, the <Institution ID> and the <Institution Name> fields from the form will be automatically filled with the ID and the name that corresponds to the Institution found and selected from the pop-up Search window.

P2000 - Old age pension claim SED API version: 0.16.3 build preview 1
Model version: 4.2.0

1. LOCAL CASE NUMBERS

1.1[1]. LOCAL CASE NUMBER

1.1.1. Country *	
1.1.2. Case number *	
1.1.3. INSTITUTION ⓘ	
1.1.3.1. Institution ID * ⓘ	
1.1.3.2. Institution Name *	

Figure 26a – Institution section with IR search widget in RINA

Advanced Search 

Institution Advanced Search

Country *	
Sector *	
Process *	

Figure 26b – IR search widget pop-up window

However, recently the possibility that not all Institutions are in the EESSI IR has been taken into consideration and the necessity to have the Institution ID and Institution Name fields as free text fields has emerged. This means that the Institution ID field should not be read-only and the Institution Name field should not have linked the IR search widget option.

1.4. INSTITUTION WHICH CERTIFIED INSURANCE RECORD *

1.4.1. Institution ID * 

SI:1111

1.4.2. Institution Name *

Institution name

Figure 26c – Institution section with fields as free-text

3.1.1.12 eTranslation field

In the dynamic forms, the large text input fields are displayed as larger text area. A text input field is considered as being large if its string data type has a number of characters equal or greater than 255.

For these, the possibility to have it translated into other languages has been implemented by adding a 'Translate' link at the bottom of the input text area.

8.2. BANK INFORMATION

8.2.1. Account holder name

[Translate](#)

Figure 27a – Text area field with Translate link

Clicking the link will open a pop-up for the RINA eTranslation widget where the languages involved in the translation process can be set and the translation operation can be started. This will make the RINA eTranslation widget to connect to the EC Connecting Europe Facility (CEF) eTranslation service (by using a local RINA eTranslation proxy service), request the translation, receive the translated text and finally to display the result in the pop-up window.

DA001 - Request for Certification of the Right to Benefits in Kind (3.5. Description)

Source Language: EN ENGLISH

Target Language: FR FRENCH (FRANÇAIS)

Translate this text

Print
Copy translation

Translate

Figure 27b – eTranslation widget pop-up window

For more details on how to configure the local RINA eTranslation proxy service and how to connect and obtain the credentials for accessing the CEF eTranslation service please use the RINA installation and user guide manuals.

3.1.1.13 Master / Child SEDs

There are SEDs that can have a section repeated for a very high number of times. For example, a SED that contains a section with global invoicing fields and then several thousand of repetitions for Individual Invoice sections (that can be generated not only by manual input but also by NAs machine-processing scripts).

Such a large SED is almost impossible to be opened and displayed in one block in the RINA portal. To solve this impediment, an approach to split such SEDs in two parts (two other different SEDs) has been implemented: a global part with information that is global and not repeatable (the Master SED) and a part that represents the repeatable data (the Child SED) that can have as many instances as necessary.

This is done at the data-model level and the generated form metadata (Master SED form metadata and Child SED form metadata) are used by eUI Dynamic Forms library to provide the same functionality as the regular forms, the only things that are different are their content and the way the Master and Child SED form metadata are accessed and used in the RINA portal.

3.1.2 Translations resources artefacts

The translation files contain localization resources needed to present all case-related information, not limited only to the form, but related to the cases themselves, in all 24 languages supported in RINA platform.

The **translation files** contain information such as:

- Labels
- Explanatory notes
- Enumeration values for selection controls
- Document names
- Other information needed by the case – case types, case actions

One translation file is created per language, containing localization strings for all SEDs in all CDM versions.

3.1.3 eUI Dynamic Forms presentation mode

The eUI Dynamic Forms library is using the same eUI metadata file to generate both a read-only and an editable version of the form.

3.1.3.1 Edit mode

The eUI Dynamic Forms library is generating, by using the eUI metadata file, an editable version of the form to allow user capture information to build a SED document. The eUI

library is dynamically selecting the controls needed for providing capabilities to the user to fill in data.

3.1.3.2 Read-only mode

For providing the read-only mode, the same eUI metadata file is used to generate dynamically, at runtime, the HTML content of the form, with the difference that all the input fields are disabled (read-only), the possibility to add/remove elements is eliminated and the footer error split-pane is not present anymore in the form. Again, the eUI library is dynamically creating the user controls needed for providing display capabilities to the form.

These read-only forms are used in the RINA portal when opening a SED form that has data from a previous operation (received SED, saved SED, etc) and are used only to visualize the actual data from a SED. In order to use the regular, editable mode, the 'Edit' operation needs to be started on the SED.

2. INSURED PERSON *

2.1. PERSON IDENTIFICATION *

2.1.1. Family name(s) * Test

2.1.2. Forename(s) * First

2.1.3. Date of birth * 12/12/2020

2.1.4. Sex * [98] Unknown

2.1.5. Family name(s) at birth ⓘ

2.1.6. Forename(s) at birth ⓘ

2.1.7. PIN OF THE PERSON IN EACH INSTITUTION ⓘ

2.1.7.1[1]. PERSONAL IDENTIFICATION NUMBER(S) *

Figure 28 – Read only dynamic form

3.1.4 Integration of Dynamic Forms inside in the RINA portal

The RINA Portal is based on Angular technology and it makes use of the eUI Dynamic Forms library to provide dynamic rendering of SED forms used to capture and display data.

The eUI Dynamic Forms library is integrated into the RINA Portal as an external component. eUI uses a metamodel file and translation files to render dynamically at runtime, an Angular reactive form with inputs to be filled by the user. Ultimately, the Angular framework is generating the HTML form displayed in the browser at runtime.

In this way, the need to store statically the HTML forms is removed. Only the metamodel is needed to generate dynamically the HTML form in any language of the EU languages supported. This provides a decoupling between form structure (stored in metamodel file) and actual labels and texts (stored in translation files). This also provides a smaller size because the form needs not to be generated as static file for each language. Instead, only one metadata file per SED is needed for each CDM version (e.g. 4.1, 4.2), as well as

24 translation files, one per each language, having translations for all SEDs and all model versions.

3.1.4.1 eUI metadata file

The eUI metadata file describes the structure of the form. The sections and the fields form a tree-like structure, each one being associated with attributes which describe how the field can be filled with data.

The **metadata file** contains information such as:

- Field/section names
- Types of the fields (single selection dropdown, multiple selection dropdown, radios etc)
- Restrictions on field values (required fields, list of values, patterns, fixed value, etc)
- Structures such as choice sections and repetition sections etc

The eUI Dynamic Forms is also generating both a read-only and an editable version of the form, based on same eUI metadata file. There is no need to store a different model for producing the read-only form.

The actual implementation of the user controls used to display and directing how a control is behaving is done by eUI Dynamic Forms and eUI Core libraries.

3.1.4.2 Translation files

The translation files contain localization resources needed to present the form in all 24 languages supported in RINA platform. The translation files contain information such as labels, explanatory notes, enumeration values for selection controls, etc.

One translation file is created per language, containing all localization strings for all SEDs in all CDM versions.

The eUI dynamic forms needs not only the eUI metadata file, but also the translation files, in order to generate on the fly the SED form with localized text.

3.1.4.1 Creating and editing SED documents

For creating a new SED document, the eUI Dynamic Forms library receive an initial empty SED document, as well as with the metamodel and the translations, and it dynamically generates a form for the user to fill data in.

In a similar fashion, for editing an existing SED document, the eUI Dynamic Forms library receives the existing document and generates a form with fields prepopulated with existing data.

eUI dynamic forms are providing validation on form input, meaning that, when a form is saved, they provide user feedback on invalid values, applying the field restrictions encoded in the eUI metamodel (for example: required fields, fields which must follow a pattern, have a maximum length etc).

As the user finishes updating the eUI dynamic form and resolves all the errors, when the save is requested, eUI produces a JSON-formatted SED document which is sent to the RINA backend. The JSON-formatted document is to be transformed in the final SED XML document which is eventually exchanged by the Member States. SED XML documents must always comply with the XSD files distributed in the CDM model.

3.2 Artefacts used indirectly by the RINA portal eUI Dynamic forms

The eUI Dynamic Forms library use metadata and translation artefacts directly, to display the actual form. The RINA portal however uses a series of other artefacts that are used by processes linked to the dynamic forms.

These artefacts are the following ones:

- SED XSD files
- SED draft XSD files
- SED Initial JSON files

3.2.1 SED XSDs

The SED XSD files represent the actual common data model contract for the SEDs that needs to be respected by all the applications (RINA, NAs) that communicate.

The SED XSD files are present in RINA and also in the CDM repository as Software artefacts.

In RINA, the XSD files are being used to validate the actual XML that has been created/used for a SED (either received or created by the Save operation).

This XSD validation is a different layer of validation than the one embedded at level of the eUI dynamic forms, being more complex and complete. It uses the actual XML data that corresponds to a SED and validates it against the XML Schema defined in the XSD.

3.2.2 SED draft XSDs

The SED draft XSD files are used only in RINA to validate the XML content of a SED for the 'Save draft' operation, which translates to an ordinary Save operation done for a document being in the 'draft' state.

In order to be able to save a draft version for a SED and later retrieve it to continue the completion, certain fields need to be completed (to be mandatory even for the draft). These fields are the 'draft critical' fields and are a subset of the normal mandatory fields.

As stated, these artefacts are present only in RINA and are not part of the CDM repository.

Similar to the normal XSD files, this draft XSD validation is a different layer of validation than the one embedded at level of the eUI dynamic form, as specified in eUI metadata file.

3.2.3 Initial JSONs (empty SED JSONs files)

The initial JSON files are used in RINA to represent the JSON data-container that will be mapped to a SED when it is opened for the first time (the SED is new). These JSON files contain the initial empty SED data-container content and are decorated as needed by the RINA ecosystem. After the SED is opened for the first time and its structure is modified (different choice selected, sections are added/removed) the underlying JSON data-container structure will be updated accordingly.

Below an example for such an initial JSON can be seen.

```
{
  "documentType": "X001_v4.1",
  "content": {
    "X001": {
      "sedPackage": "Sector Components/Administrative/X001",
      "sedGVer": "4",
      "sedVer": "1",
      "CaseContext": {
```

```
        "PersonContext":{
            "familyName":"","
            "forename":"","
            "dateBirth":"","
            "sex":""
        }
    },
    "InformationAboutClose":{
        "closeDate":"","
        "close":"","
        "reasonForClosing":"","
        "pleaseProvideMoreDetailsIf990therSelected":""
    }
}
}
```

Figure 29 – Initial JSON file for the X001 SED