D5.11

Design of information extractor and natural language translator - v1

ABSTRACT

This deliverable provides motivations, background, and specifications for multilingual natural language processing functionalities for the purposes of healthcare data interoperability, according to the scope of the InteropEHRate project. It covers two main areas of use of language technologies: automated translation and information extraction. The role of the former is to present health record content in a language different from the original, the target being both patients and healthcare professionals practising in a different country. The role of the latter is to convert unstructured health record data into formal data structures, allowing their automated integration and querying.

Delivery Date	February 3rd, 2021
Work Package	WP5
Task	T5.4
Dissemination Level	Public
Type of Deliverable	Report
Lead partner	UNITN

InteropEHRate project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826106.

This document has been produced in the context of the InteropEHRate Project which has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826106. All information provided in this document is provided "as is" and no guarantee or warranty is given that the information is fit for any particular purpose.

CONTRIBUTORS

	Name	Partner
Contributors	Gábor Bella	UniTN
Contributors	Simone Bocca	UniTN
Contributors	Francesco Torelli	ENG
Contributors	Stefano Dalmiani	FTGM
Contributors	Patrick Duflot	CHU Liège
Reviewers	Stella Dimopoulou	BYTE
Reviewers	Sofianna Menesidou	UBITECH

LOG TABLE

Version	Date	Change	Author	Partner
0.1	2020-09-17	First version created	Simone Bocca	UniTN
0.2	2020-09-21	TOC finalised for review	Gábor Bella	UniTN
0.3	2020-12-18	Added material on information extraction	Gábor Bella	UniTN
0.4	2021-01-19	Finished section on translation	Simone Bocca	UniTN
0.5	2021-01-20	Added more material on information extraction	Gábor Bella	UniTN
0.6	2021-01-27	Wrapping up for internal review	Gábor Bella	UniTN
0.7	2021-01-28	Sent for internal review	Gábor Bella	UniTN
0.8	2021-02-01	Internal review applied	Stella Dimopoulou, Sofianna Menesidou	BYTE, UBITECH
0.9	2021-02-02	Applied suggestions from internal review	Gábor Bella	UniTN
Vfinal	2021-02-03	Quality check and final version for submission	Laura Pucci	ENG

ACRONYMS

Acronym	Term and definition
MT	Machine Translation
NLP	Natural Language Processing
HCP	Healthcare Professional
IE	Information Extraction
IHS	InteropEHRate Health Services: the information system deployed at a healthcare organisation that offers high-level services and performs operations related to health data conversion and interoperability, including translation and information extraction.
IHSI	InteropEHRate Health Service Interface
KE	Knowledge Extraction
TranslationI	Translation Interface

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1. Scope of the document

This deliverable provides a specification of the services, software components, and processes responsible for (1) the translation of electronic health record content into different languages, thus helping the cross-border transferability of health records; and (2) the extraction of symbolic, language-independent---and thus cross-border interoperable---knowledge from natural language text within health records through natural language processing methods.

1.2. Intended audience

This document is intended for (1) engineers who wish to use the services described here for processing health records within their own healthcare information systems, within or without the InteropEHRate project; (2) engineers or researchers wishing to design and implement similar services; and (3) healthcare professionals wishing to understand the potential uses and the challenges related to the linguistic processing and translation of text within structured and unstructured health data.

1.3. Structure of the document

Section 2 motivates the use of translation and information extraction in the scope of the InteropEHRate scenarios. Section 3 specifies the InteropEHRate services for health data translation. Section 4 specifies the InteropEHRate services for information extraction. Section 5 presents related work and how our design goes beyond the state of the art. Section 6 gives conclusions and addresses future work.

1.4. Updates with respect to previous version (if any)

Not applicable.

2. RELATION TO INTEROPEHRATE SCENARIOS

Translation of natural-language text contained in (structured and unstructured) health records is a requirement raised by cross-border operation. Scenarios 1-3 of InteropEHRate all represent cross-border use cases: in Scenario 1, a Belgian citizen moves to Greece and presents his/her smart health record at a Greek hospital. In Scenario 2, an Italian patient suffers an accident in Greece and his/her health record is subsequently retrieved from the S-EHR Cloud to be consulted by Greek HCPs. Finally, in Scenario 3, health records from multiple countries are retrieved automatically in order to carry out cross-border medical research. Note, however, that the translation of health records may be useful even in a single-country scenario, if the patient does not speak the local language but would like to understand his/her health data nevertheless.

Information extraction serves the purpose of automatically extracting structured information from pieces of unstructured natural-language text within health records. While such a functionality is generally useful for converting unstructured data into structured form that is both easier to process by computers and easier to understand by humans, within the InteropEHRate project it is specifically used in Scenario 3 that implements medical research based on health data automatically retrieved from the mobile devices of patients. Since quantitative research requires fully structured data, relevant pieces of natural-language text contained in health records need to be converted to a structured form. Within the research scenario, information extraction can be executed:

- *upstream*, that is upon the retrieval of smart health data from a hospital, in order to allow the automated querying and sharing of previously extracted data from citizens' mobile devices;
- and/or *downstream*, that is within a research centre, after having received the health data retrieved from a citizen's mobile device.

While the main use of information extraction within InteropEHRate is limited to the research scenario, there is no limitation to its use in the other scenarios, as its implementation does not change from a system integration point of view. For this reason, the InteropEHRate demonstrators corresponding to the other scenarios can also integrate information extraction functionalities.

Both translation and information extraction are *optional* components of the InteropEHRate Health Services. This means that (1) their implementation is not necessary for successful cross-border interoperability in a strict technological sense; (2) the additional information provided by these two services are for informative purposes only, and their quality is not guaranteed due to current technological limitations; and (3) the results provided of these services can be ignored by HCPs and even discarded at any moment from the health records.

3. DATA TRANSLATION

This section presents the motivations, service specifications, and service architecture for the translation of natural language text contained within health data.

3.1. Motivations

One of the most important objectives of InteropEHRate is to allow citizens to carry their health data across European borders. In this general scenario, a patient may not be a speaker of the language in which his/her health data was produced, yet he/she may want to be able to read and understand it. Likewise, a healthcare practitioner accessing the health data of a patient in a foreign country may not understand neither the patient nor the original language of the health data. Thus, there is a major need that the IT framework in charge of data interoperability should support the translation of health data content. For this reason, the InteropEHRate project includes a set of services developed to translate the contents of the healthcare resources exchanged between human actors. As will be better detailed in the following sections of the document, these services work on the different kinds of data (codified or free-text) contained in the health data managed in the project.

3.2. Translation methods

In order to provide the needed translations, as described in the above section, InteropEHRate considers, and offers as services (described in the following sections) within its framework, two different methods to translate the health data. The two methods defined as *knowledge Driven Translation* and *Machine Translation*, adopt different approaches to translate the data and they are applied on different kinds of data. In the sections below, a description of the two translation methods is provided.

- *Knowledge Driven Translation*: As the name suggests this kind of translation is based on a set of multilingual knowledge resources. This set of resources aims to collect information about each code contained in specific medical standards, used to define medical concepts within the health data. Being multilingual, this set of knowledge resources contains, for each standard code, a textual description of the concepts expressed through the code itself, defined in different natural languages (such as Italian, English, French, and so on, depending on the availability of the language for the specific medical standard considered). The resources that compose this set are obtained directly from the organization which defines the medical standards. The knowledge resources are stored in a dedicated database system, called Knowledge Base, part of the Data Integration Platform [D5.8] which supports the InteropEHRate framework. The knowledge driven translation aims to exploit this set of knowledge resources providing the description of a specific concept, identified by its code in the medical standard, translated in the language needed for the citizen or for the healthcare practitioner. Thanks to this translation approach, each time a medical standard code is identified within health data, it is possible to provide the code's description in the desired language (if the set of knowledge resources include the descriptions of the specific medical standard in the desired language). This kind of translation can be applied only on 'structured' data, or in other words, to data representing a specific concept included in the knowledge base. In this specific case, the knowledge driven translation is applied on medical codes defined by specific medical standards (such as SNOMED-CT, ICD10, LOINC). The quality of this kind of translation is

based on the definition of medical standards collected in the knowledge base, for this reason the translation can be considered reliable.

- *Machine Translation*: Respect the previous kind of translation, this one is applied to portions of 'free-text', or 'unstructured' text, in other words, natural language part of text inserted within the health data. The machine translation aims to translate portions of text using an external translation service, such as Google Translate. This kind of translation is applied to the same health data, as the previous one, but it works on different portions of those data, in this case, the data identified to be translated are text fields such as medical reports or drugs descriptions. This translation, due to the fact that it is based on external text translation service, cannot be considered reliable as much as the previous one, but it can cover the 'unstructured' portion of health data, that cannot be translated using the knowledge driven approach.

The two translation methods described above are implemented as separate services within the InteropEHRate framework, which can be used through the endpoints collected in the Translation Interface. The services and their usage are defined in the following sections.

3.3. Translation Services

As described in the motivation section, the health data translated have to be provided to the healthcare professional (HCP), through the HCP application. The HCP, having the translated data can consult and/or update them, then the data can also be forwarded to the citizen through the S-EHR application. Due to these necessary operations, the HCP application has to interact with the translation services in order to retrieve the translated data, and provide them to the HCP.

The access to the translation services is provided through a specific component of the InteropEHRate framework, called IHS Controller [D5.8] that acts as a middleware between calls from external applications (such as the HCP App) and health data conversion and translation functionalities. The IHS Controller, accessed through the IHS Interface (IHSI), provides the usage of the Translation Interface (TranslationI) that includes the endpoints for two different translation services, the *Machine Translation Service* and the *Concept Translation Service*. The two translation services, respectively based on the machine translation and knowledge driven translation, work on health data structures, defined following the interoperability profiles based on FHIR data standard.

The table below describes the endpoint provided by the Translation Interface. It includes a method, called *extendWithTranslation*, invoked by the IHS Controller in order to translate the content of an health record. This method involves, as internal calls, the usage of the other two specific translation methods, however the TranslationI can expose the calls of all the internal methods singularly.

TranslationI Endpoint	Input	Output	Example call URI	Description
POST /extendWithTranslation	JSON-FHIR resource, String targetLang, LEVEL	JSON-FHIR translatedResource	/extendWithTranslation?targetLang=it&level=SEM	This call allows the translation of an entire FHIR resource (i.e. Bundle), calling internally both the machineTranslation and

	localInteropLevel			conceptTranslation methods. It takes as input a FHIR resource and provides as output the same resource with the attributes translated in the language indicated in input (targetLang).

Table 1 - High-level TranslationI endpoint for translating an entire FHIR resource

In order to work correctly the translation services refer to the initial configuration parameters provided to the IHS, such as the main language used by the systems (and users) that have to interact with the IHS Controller in order to exploit the translation services.

In the sections below the two translation services are described in detail.

3.4. Machine translation service

The Machine Translation service is one of the two services, in charge of translating the content of health data, provided by the Translation interface. The service aims to translate small portions of free-text (unstructured text not involving codes or specific medical terms) within the health records. It acts on specific attributes of the smart health data, identified following the FHIR structure defined by the interoperability profiles. Thanks to the profiles definition, the service knows, for each medical resource, which are the attributes that have values containing free-text, and so proceeds to identify them within the smart health data, performing the translation needed. Once the target attributes have been identified, the translation of the respective values is performed calling an external text translation service. In the first version of this component the external service adopted was the IBM Watson Language Translator, which provides the translation of one million characters per month for free. In order to improve the translation quality, the final version of the machine translation service will adopt the Google Translator as external service.

As already mentioned, the machine translation method is invoked internally by the *extendWithTranslation* method, moreover it can be called, as a single service, through the endpoint provided by the TranslationI.

Machine Translation Service	Input	Output	Example call URI	Description
POST /machineTranslation	JSON-FHIR resource, String targetLang	JSON-FHIR translatedResource	/machineTranslation?targetLang=it	This call allows the translation of specific free-text attributes only, within a FHIR resource. It takes as input a FHIR resource and provides as output the same resource with the attributes translated in the language indicated in input (targetLang). This service is also used by the more

				general endpoints retrieveFHIRHealthRecord and extendWithTranslation

Table 2 - TranslationI endpoint for machine translation

3.5. Concept translation service

The Concept translation is the second service, offered by the Translation interface, for the translation of smart health data. This service aims to translate structured information identified within the smart health data structure. Thanks to the definition of the interoperability profiles, the service knows which are the FHIR attributes that can include structured data.

The data managed by the concept translation are represented by medical codes and standards terms following specific coding systems (such as SNOMED-CT, ICD-9, ICD-10, LOINC, and others listed in [D5.8]), that identify medical concepts (i.e. Loinc:8716-3 represents "Vital signs"). All the medical concepts used in the health records (used both at local and international level), together with their meaning descriptions, form the *Knowledge resources* used to support the procedure of the Concept Translation Service. Due to that, this kind of translation is called "knowledge driven" (more details regarding the knowledge driven approach are reported in the section 3.2 of the current document, and in [D5.10], section 4. The knowledge resources are previously collected and stored in the Data Integration Platform [D5.8], moreover the description of the concepts is collected in several languages, composing in this way a set of multilingual knowledge resources that is exploited for the translations.

The concept translation procedure works by identifying, within the smart health data structure, the attributes which can contain structured values (concepts values). Once the concept value has been identified, the service checks if the value, representing the medical concept, is present within the knowledge resources set (stored in a dedicated database system called *Knowledge Base*) in the Data Integration platform, and if the meaning of that concept is available in the language specified in input as target for the translation. In case the concept and meaning (in the correct language) are available, the service adds the description of the specific concept in the language desired, to the smart health resource that has to be translated.

If the concept value is not present within the knowledge resources available, and/or the meaning of a specific code/term is not available in the target language specified in input, the Concept Translation Service cannot provide the translated value requested. Due to this possibility, a log (more details about the log file(s) produced are available in D5.8) is produced for any code or other information that cannot be translated, in order to be consulted by the Data Scientist to complete or improve the corresponding knowledge.

As already mentioned, the concept translation method is invoked internally by the *extendWithTranslation* method, moreover it can be called, as a single service, through the endpoint provided by the TranslationI.

Concept Translation Service	Input	Output	Example call URI	Description
POST /conceptTranslation	JSON-FHIR resource, String targetLang	JSON-FHIR translatedResource	/conceptTranslation?targetLang=it	This call allows the translation at concept level (the translation of all the codes and terminology terms identified as concepts on the base of the domain knowledge collected). It takes as input a FHIR resource and provides as output the same resource with the concepts translated in the language indicated in input (targetLang). This call is used within the more general endpoints retrieveFHIRHealthRecord and extendWithTranslation

Table 3 - TranslationI endpoint for concept translation

3.6. Translation Sequence Diagram

The diagram below shows how the interfaces above are called to implement the translation of smart health data downloaded from the S-EHR App, and eventually updated, by the HCP, with new medical (FHIR) resources. After the figure a description of the procedures invoked is provided.

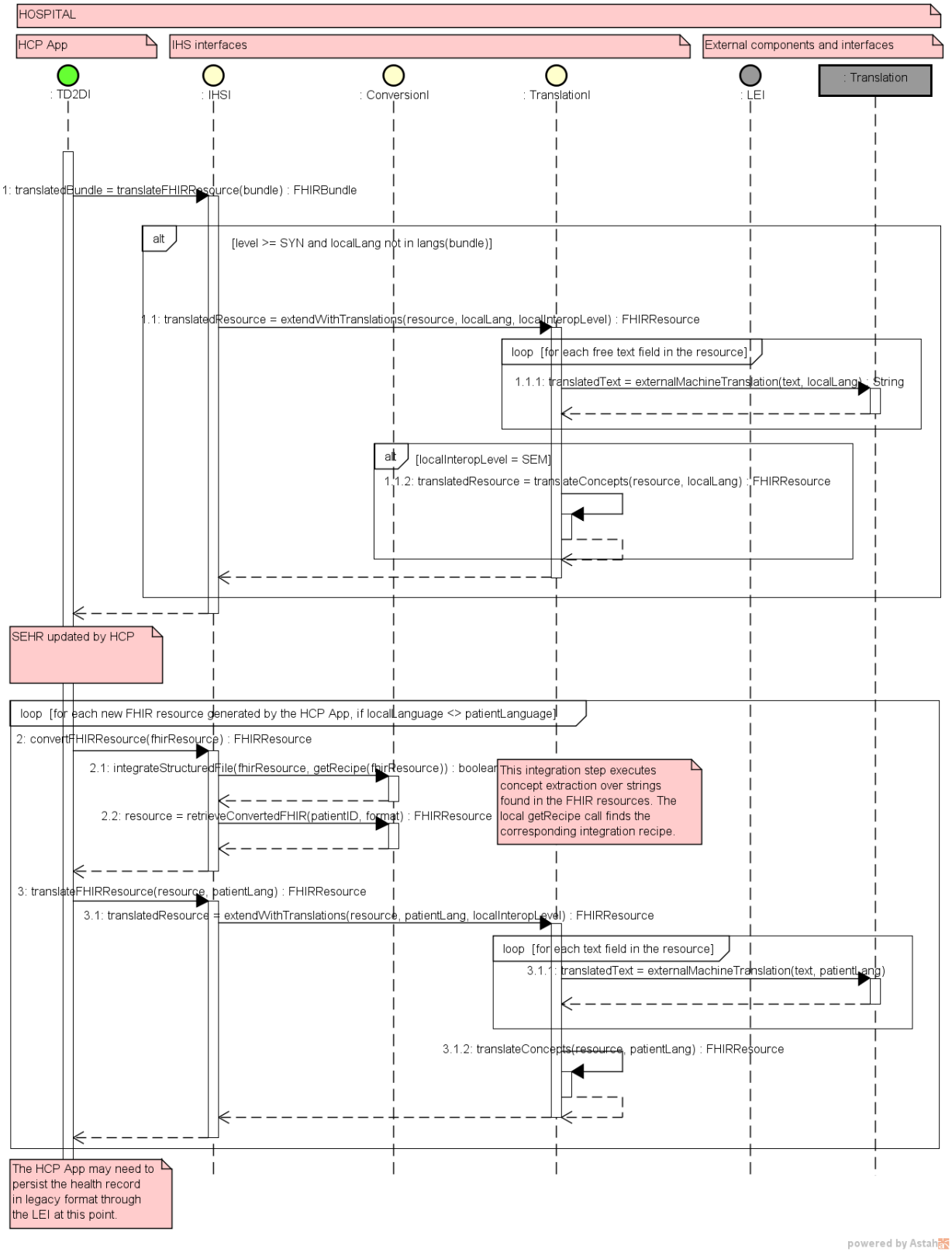

Figure 1: Sequence diagram of a request, coming from the HCP App, to translate an existing health record downloaded by the S-EHR App.

The first method, as shown in the diagram above, is translateFHIRResource (see [D5.8] for more details), offered by the IHS Interface and called from the HCP application. This first method aims to translate smart

health data, represented by a set of FHIR resources, passing in input a FHIR bundle containing such resources. If the HCP local language (specified in a configuration file of the IHS) is different respect the language used in the FHIR bundle in input, and the local language is not included in the same bundle, the IHS invokes the method extendWithTranslation, offered by the Translation Interface. The Translation Service, now, checks if the 'localInteropLevel' parameter, passed in input in the last method invoked, is equal to 'SYN' or 'SEM'. In the first case, the Translation Service calls only the Machine Translation Service's method (machineTranslation), that internally calls the external translation service (externalMachineTranslation). While in the latter case, it always calls the Machine Translation Service's method and after that also the Concept Translation Service's method (conceptTranslation) is invoked. Then the translated FHIR resource is returned from the Translation Service to the IHS and then forwarded to the HCP application.

In the second part of the diagram, shown in figure 2, is defined the activity process invoked when the smart health data has to be updated by the HCP (i.e. adding some health data), through the HCP application. In this case, the first method called is relative to the conversion of the new data in order to be added to the original smart health data (FHIR based), this activity process is described in [D5.10]. The second method invoked on the new data added by the HCP, is the already described translateFHIRResource that aims to eventually translate the new information added in order to be consulted by the citizen when the smart health data will be returned on the S-EHR application.

3.7. Translation service architecture

In the diagram below is shown the architecture of the Translation service and its interactions between the other InteropEHRate systems as well as external systems involved.

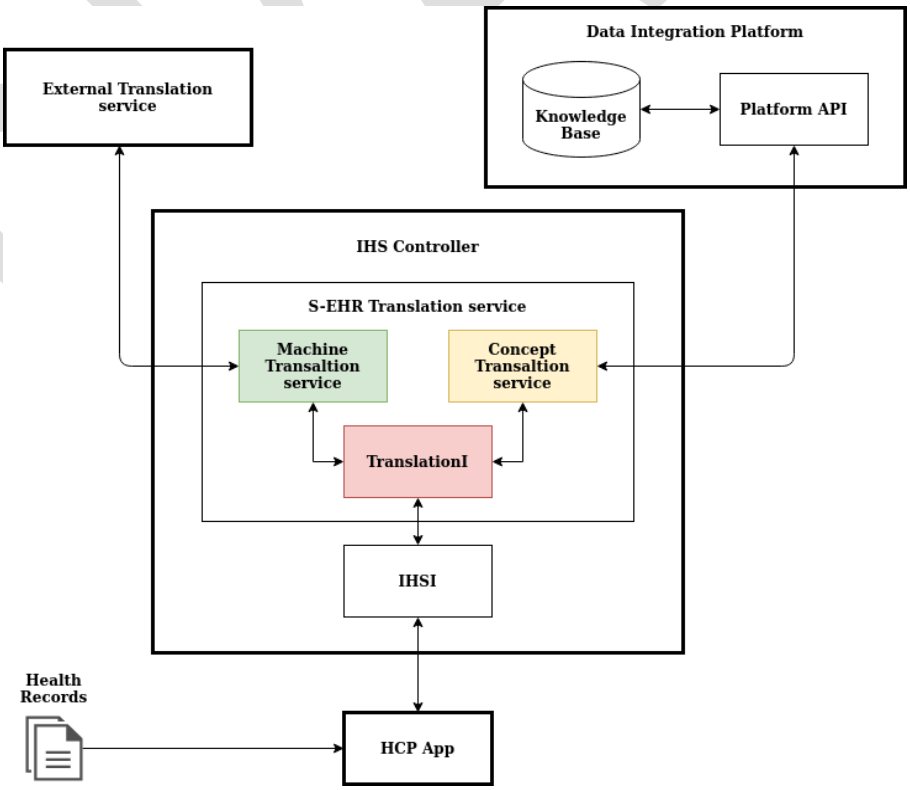

Figure 2 - Translation service architecture and external interactions

The efficiency of the services that perform automatic translations cannot be considered completely reliable, a margin of error is always considered in the translation procedures. Due to this, the interoperability profiles, used to define the health records, support the usage of FHIR *extensions*. The extensions are specific FHIR attributes that are associated with the target attribute that has to be translated. The extension includes, for each translation applied to the target attribute, the translated value of the attribute to which it refers, the target language used for the translation, and a reference to a FHIR *Provenance* resource, that is added to the S-EHR by the translation services, which specifies the agent who performed the translation (the software application) as well as when the translation has been produced, using a date type attribute. Thanks to the usage of FHIR extensions, the original data are not altered by the translation process, every time that a new translation is needed a new extension is associated with the target attribute. The implementation of both the Machine and Concept translation services support, and manage the translations using the FHIR extension.

In order to provide more detailed information regarding the translations management, a concrete example of translated FHIR file is here described. In this specific case the FHIR bundle resource, reported in the file, includes one *Observation* resource type, that has been managed by the Translation service. The FHIR Observation resource includes one Observation code defined by the *code* attribute which is composed by three sub attributes:

- **system**: the medical standard to which it belongs. In this case the value of this attribute indicates the LOINC medical standard.
- **code**: the standard code value. In this example the value is [10333-3].
- **display**: a textual value describing the code meaning. In this specific case the text value is "Appearance of Cerebral spinal fluid".

```
"code": {
  "coding": [ {
    "system": "http://loinc.org",
    "code": "10333-3",
    "display": "Appearance of Cerebral spinal fluid",
```

Figure 3 - "code" FHIR attribute example in English language

The *display* attribute's value is, in this example, the content that has to be translated in the desired language by the translation service. More in detail, this example, perform the translation in three different languages, German, Italian, and Dutch. Due to that, three different extensions are added for the target attribute *display*. The FHIR extension structure, as already mentioned above, involves the addition (immediately after the target attribute) of a new attribute named [*_display*] (more in general the new attribute is composed by the target attribute name preceded by the '_') which contains the *extension* sub-attribute. *extension* itself is a list of elements (one for each translation performed) composed by two sub-attributes, respectively, *url* (an identifier of the translation) and a list attribute called again *extension*. The content of the latter are three elements:

- {"url": "lang", "valueCode": <language ISO code>} : this element indicates the target language of the translation defined by the specific extension.
- {"url": "content", "valueString": <translation string value>} : this element contains the target attribute's value translated.

- {“url”: “http://interopehrate.eu/fhir/StructureDefinition/ExtendedTranslation-IEHR”, “valueReference”: <FHIR Provenance reference>} : this element specifies the reference to the FHIR Provenance resource (in the same FHIR Bundle) that includes the translation provenance information.


```
"_display": {
  "extension": [ {
    "url": "http://hl7.org/fhir/StructureDefinition/translation",
    "extension": [ {
      "url": "lang",
      "valueCode": "de-DE"
    }, {
      "url": "content",
      "valueString": "Erscheinungsbild:Ersch:Pkt:CSF:Nom:"
    }, {
      "url": "http://interopehrate.eu/fhir/StructureDefinition/ExtendedTranslation-IEHR",
      "valueReference": {
        "reference": "Provenance/1"
      }
    } ]
```


Figure 4 - FHIR extension example for a translation to German language

An element is generated, and added to the external *extension* list, for each different translation performed. If, for example, also the Italian translation is needed, the following element will be added:


```
"url": "http://hl7.org/fhir/StructureDefinition/translation",
"extension": [ {
  "url": "lang",
  "valueCode": "it-IT"
}, {
  "url": "content",
  "valueString": "Aspetto:Asp:Pt:LCS:Nom:"
}, {
  "url": "http://interopehrate.eu/fhir/StructureDefinition/ExtendedTranslation-IEHR",
  "valueReference": {
    "reference": "Provenance/1"
  }
} ]
```


Figure 5 - FHIR extension example for a translation to Italian language

During the translation process, as already mentioned, the FHIR Provenance resource is added, in the end of the FHIR Bundle of the file in input. The FHIR Provenance resource structure is composed by the following attributes:

- fullUrl : global identifier of the specific FHIR Provenance resource.
- resourceType : always defined as “Provenance”, indicates the type of FHIR resource described.
- id : internal identifier of the specific FHIR Provenance resource.
- target : a reference to the FHIR resource considered in the translation, which generates the provenance resource itself. In the example here described, this is the reference to the FHIR Observation resource.
- occurredDateTime : the moment (date time value) in which the translation has been performed.
- recorded : the moment (date time value) in which the provenance information has been recorded.
- agent : a reference to a FHIR Device resource which indicates the software application that performs the translation.

In the figures below are reported, the FHIR Provenance resource and the FHIR Device resource, generated in the translation example considered.


```
"fullUrl": "http://213.249.46.205:8080/NCP/fhir/Provenance/1",
"resource": {
  "resourceType": "Provenance",
  "id": "1",
  "target": [ {
    "reference": "Observation/987654"
  } ],
  "occurredDateTime": "2020-11-26T09:51:03+01:00",
  "recorded": "2020-11-26T09:51:03.350+01:00",
  "agent": [ {
    "who": {
      "reference": "Device/4321"
    }
  } ]
}
```


Figure 6 - FHIR Provenance example


```
"fullUrl": "http://213.249.46.205:8080/NCP/fhir/Device/4321",
"resource": {
  "resourceType": "Device",
  "id": "4321",
  "status": "active",
  "deviceName": [ {
    "name": "TestDevice",
    "type": "user-friendly-name"
  } ]
}
```


Figure 7 - FHIR Device example

4. INFORMATION EXTRACTION

Information extraction serves the purpose of automatically extracting structured information from pieces of unstructured natural-language text within health records. Such information can be numerical values (e.g. “37.5 °C”), domain terms (e.g. “*myocardial infarction*”), or names (e.g. “Aspirine”). Besides the extraction of such *data values*, it is also necessary to specify the structured attributes to which the values belong (such as *“body temperature”*, *“primary diagnosis”*, or *“prescribed item”*), whether or not they are explicitly specified within the text.

4.1. Motivations

Health data is often required or desirable to be available in structured form:

- it is a necessity for automated querying or analytics of data;
- data elements may need to be displayed to humans in a structured representation (e.g. inside a form or a table);
- the automated integration, conversion, or translation of data may be easier when the data is structured.

In the InteropEHRate project, the Medical Research Scenario requires data to be presented to researchers in an integrated and structured form, both for human inspection and automated analytics. Inside health records, however, shorter or longer pieces of natural-language text abound, as shown by the examples below.

Within unstructured or semi-structured documents such as visit or discharge reports:

Histoire de l'affection et plaintes :
Patiente agée de 64ans qui à été opérée en France au mois de mai 2012 d'une fracture de radius distal.
Les suites opératoires se sont compliquées d'un syndrome d'algoneurodystrophie.
A deux ans de l'intervention, indication d'ablation de la plaque et de neurolyse du nerf médian.

Détails de l'intervention :
AMOS de plaque radial gauche sous anésthésie plexuelle le 08/xx/2014.

Suites opératoires :
Les suites opératoires ont été simples.
La patiente a présenté des douleurs bien contrôlées par les antalgiques classiques.
La plaie est saine, on ne note aucune rougeur ni aucun suintement.
Pas de trouble neuro-vasculaire.

Traitement de sortie :
Zaldiar 1 à 2 cp 3x/j
Automobilisation directe.

Figure 8 - Example of data that can be extracted from unstructured text

Within short phrases contained within structured data, such as drug product names or prescription instructions:

```
<medicinalproduct>
  <intendedcd S="CD-DRUG-CNK" SV="20100701">1539238</intendedcd>
  <intendedname>Metformine Mylan filmomh. tabl. 60x 500mg</intendedname>
</medicinalproduct>
```

Figure 9 - Example of data that can be extracted from a phrase inside structured data

Individual terms expressed as text inside otherwise structured data:

```
<content>
  <text L="fr">Insuffisance rénale chronique</text>
</content>
<content>
  <cd S="CD-CLINICAL" SV="1.0">10119104</cd>
  <cd S="ICD" SV="10">N18.9</cd>
  <cd S="ICPC" SV="2">U99</cd>
</content>
```

Figure 10 - Example of individual concepts that can be extracted from terms inside structured data

4.2. Problem

Information extraction or *knowledge extraction* (especially in the context of building a knowledge graph from the data extracted, as is our case in the InteropEHRate project) is a much-researched problem from the general area of natural language processing. It is usually formalised as performing together the specific tasks of:

- **concept extraction:** finding relevant terms in the text, such as “thrombosis”, and linking them to their respective concept meanings;
- **entity extraction:** finding names in the text, such as “Aspirine”, and linking them to their respective entity meanings;
- **relation extraction:** finding the attributes corresponding to the concepts and names extracted, such as “diagnosis = thrombosis” or “drug name = Aspirine”.

These subproblems are generally considered to be hard to automate, operating within a range of accuracy (typically 60-90% of precision and recall) and capable of reaching a very high level of precision (>90%) only in controlled circumstances. The specific problem of information extraction over health records, solved within the InteropEHRate project, is made even more complex by the following additional requirements.

Multilinguality. The operations above need to function over multiple languages (potentially all languages of the European Union). This means that, to some extent, solutions that are specific to each language need to be developed.

Domain language. The language to be processed is specific to the healthcare domain and its subdomains. This involves the extensive use of domain terminology and domain grammar. As conventional natural language processing (NLP) resources and tools are not efficient for the processing of these kinds of text, specialised resources need to be developed and/or adopted.

Semi-structured text. Beyond longer pieces of unstructured text (such as a discharge report), short phrases contained within otherwise structured data also need to be parsed. Such short phrases contain a smaller amount of contextual information and often follow a grammar (orthography, syntax) that is different from regular text, called “the language of data” [LanguageOfData].

High precision. The health domain requires information extraction to be very precise: even at the price of lower recall (a lower amount of information extracted), the information extracted must not be erroneous, lest downstream data processing and use should be compromised. Again, this means that NLP methods

that increase precision should be favoured over conventional approaches that tend to optimise for recall or F-measure (i.e. the balance of precision and recall).

Integration into the InteropEHRate Health Services. Our goal is not merely to specify and design a stand-alone information extractor tool or prototype, but to integrate a functional component into the InteropEHRate Health Services. This means that the output of the information extractor must be reused by the Health Data Integration Platform to integrate the information extracted as structured FHIR data, allowing subsequent conversion and translation services to be run over them. It also means that the *data scientist* who oversees the entire integration process should be given control over the extraction method.

4.3. High-Level Solution Design

The solution proposed within the InteropEHRate project is based on the following macro-components:

- a fully automated, extensible *Information Extraction Tool*;
- the semi-automated piloting of NLP operations provided by the tool above by a data scientist using the *Data Mapper Tool* defined in [D5.8];
- the integration of the information extracted into the FHIR-based data representation using the *Health Data Integration Platform*.

The components within the InteropEHRate Health services that constitute our solution are highlighted in red in the figure below.

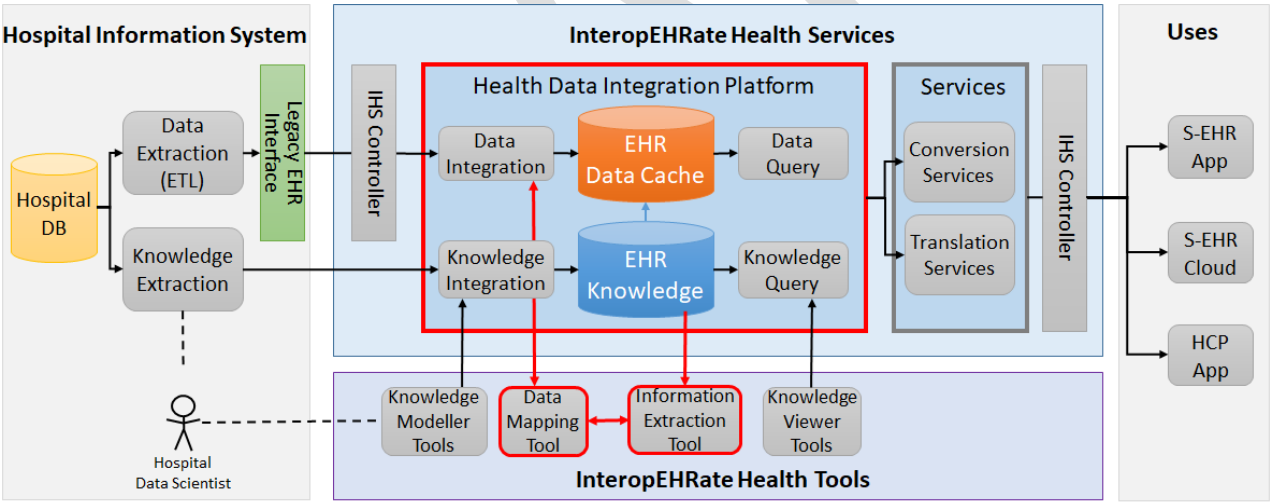

Figure 11: Components related to the Information Extraction task within the InteropEHRate Health Services

The Information Extraction Tool is set up to be able to process local health records in the local language. It is configured to carry out a set of "elementary" information extraction operations automatically, such as "extract prescribed drugs" or "extract body temperature". A local data scientist specifies which elementary operation to be applied to which input health record file (or portion of the file: paragraph, data value, etc.), and to which target FHIR health record attribute to associate the extracted value. These manually defined specifications are then executed automatically by the Health Data Integration Platform during the conversion of health records.

The Information Extraction Tool not only identifies portions of the input natural-language text to be extracted, but also annotates these portions by formal health knowledge previously uploaded to the knowledge base ("EHR Knowledge") of the Data Integration Platform. For example, as shown in the figure below, after parsing the text "Paracetamol 500mg", the substrings "Paracetamol" and "mg" will be linked by the Information Extraction Tool to the related concepts of the substance and the unit of measure, respectively, using the international standards ATC and UCUM. Subsequently, the Data Mapper tool is used to convert the annotated string into a fully structured and integrated form.

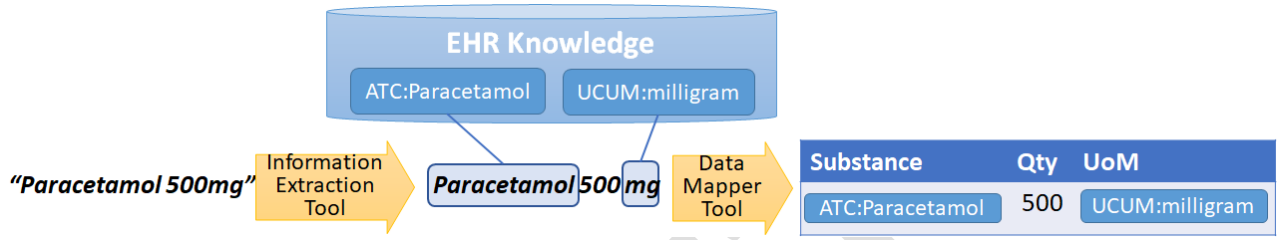

Figure 12: Information extraction processing steps over a text string

4.4. Automated Information Extraction

Information extraction from health records---or more precisely *knowledge extraction*, the difference residing in the degree of formality of the data extracted---is realised by a domain-specific and language-specific NLP pipeline that implements the three main NLP tasks described above: concept, entity, and relation extraction. Domain specificity refers to the NLP pipeline being adapted to processing text from the health domain, and from electronic health records in particular. Language specificity refers to the pipeline being adapted to the language of the text.

Due to the general context of data integration and analytics, an important feature of the knowledge extraction solution is that it should be *knowledge-driven*: this means that concept, entity, and relation extraction are all executed with respect to a knowledge base. In other words, the concepts, entities, and attributes to which the extraction process links pieces of natural-language text can all be found within a reference knowledge base, depicted in Figure 7 as "EHR Knowledge". The goal of the entire process is precisely to find the links between pieces of the input texts and elements of the background knowledge. This allows unstructured text eventually to become structured knowledge that can be automatically processed downstream.

4.4.1. Input Requirements

Input text. The NLP pipeline takes as its principal input a piece of plain natural-language text. The text may be as short as a single word or as long as several paragraphs of text. For texts longer than a single sentence, it is assumed that sentences and phrases are clearly delimited (e.g. by sentence-ending periods). The quality of the NLP output will, of course, depend on the quality (noisiness, grammaticality, etc.) of the input text.

Input language. The caller can specify the language of the input text. This is optional as the language can also be detected automatically. Manual specification can, however, avoid the occasional language

detection mistake. The language of the input text must be among those supported by the NLP pipeline deployed.

Pipeline selection. The Information Extractor Tool may (and typically will) implement multiple NLP pipelines corresponding to different text processing tasks, usable in different contexts. For example, different pipelines may interpret prescriptions and discharge reports. Therefore, the caller needs to specify which pipeline to execute.

Input constraints. The pipeline allows constraints to be specified with respect to the concepts, entities, and relations to be extracted. For example, the caller can specify that the NLP process should only find drug names, or only diseases, or even only certain forms of disease. Such constraints are formulated in a knowledge-based manner: for example, if only cardiovascular diseases are to be found, then the caller should specify as a constraint the concept of *cardiovascular disease* (e.g. expressed as an ICD-10 code). The pipeline will then attempt to extract terms that refer to concepts that are equivalent to or more specific than *cardiovascular disease.* With respect to unconstrained operation, such constraints increase the precision of the results significantly.

API call. The Information Extractor Tool provides the following interface, described below as a RESTful web API.

Information Extractor	Input	Output	Description
GET /nlp/pipelines	none	JSON	Get the list of available NLP pipelines.
POST /nlp/runPipeline	String[] text, NLPParameters parameters (in JSON)	SemText[] annotatedText (in JSON)	Run a specific pipeline.

Table 4 - The API of the Information Extractor Tool

The NLPParameters object contains all input parameters of the NLP pipeline, including the text language and the input constraints. Its precise structure will be defined in the next version of this deliverable.

4.4.2. Output Requirements

Conventionally, NLP pipelines provide their output as annotations over the (itself intact) input text. The precise annotation format is important as the output of the NLP is supposed to be read automatically by the Data Mapper Tool (see Figure 7) in order to convert the annotations into data structures. While solution providers can agree on multiple possible representation formats for NLP annotations. Currently there is no single prevailing standard for representing NLP annotations, only more or less popular NLP frameworks (such as [UIMA]) each providing its own markup method. The implementation provided as part of the InteropEHRate Project uses the open, JSON-based SemText ("semantic text") format described here [SemText].

The actual annotations output by the Information Extractor Tool depend, of course, on the contents of the texts being processed. In a typical scenario of processing local EHR data inside a hospital, the text will

contain locally used codes, local terminology, and names of locally used entities (drugs, medics, etc.). Some of the terms, of course, may be internationally used, such as units of measure or international codes. There is no consciously in-built limitation inside the Information Extraction Tool with respect to the kinds of terms or names that it can recognise: rather, it is the contents of the underlying background knowledge that limit its capabilities.

4.4.3. Internal Design Guidelines

As multilingual and domain-specific NLP is considered to be a very hard and actively researched problem, it is not possible to give a single right solution to solving the three extraction tasks at hand. We therefore provide only general guidelines with respect to the major design decisions.

Knowledge-based operation. There is today a major division between, on the one hand, cutting-edge language technology presented as research (conferences, journals) and, on the other hand, actual language processing systems implemented and used in production. Research-centric solutions have embraced machine learning and, more in particular, black-box end-to-end deep learning approaches. These approaches are usually based on training over large textual corpora (mostly English as less data is available for other languages) and their success is measured in terms of incremental improvements over state-of-the-art precision and recall figures. In research, a precision improvement of 5%, from, say, 30% to 35%, is considered a success. In a production system, of course, a tool with an overall precision of 35% would not be usable, especially not in the healthcare domain. The typical black-box operation of deep learning is also a disadvantage for healthcare applications, as health experts do not understand and do not have direct control over how the system operates and why it provides certain results. For these reasons, systems actually used in healthcare typically employ a combination of conventional machine learning methods, processing rules, and knowledge-based (e.g. ontology-based) reasoning. This is what one of the leading medical NLP systems, [Apache cTAKES], is doing: it is integrating machine-learning preprocessing components (such as tokenizers, part-of-speech taggers, named entity recognisers), rule-based term extractors, and connectivity to [UMLS] from where health concepts are retrieved. At least at the time of writing this deliverable, the partially rule-based and knowledge-based approach is unavoidable, and is also adopted for project-level implementations.

Multilinguality. The multilingual requirement puts a serious design constraint on the implementation. It is not likely that a single solution provider will ever be able to implement an all-in-one NLP solution that serves all languages of the European Union on the same level of accuracy. Rather, it is more likely that each country will need to develop its own NLP solutions for its own language(s). The system requirement is therefore that the general NLP framework (1) should be customisable so that language-specific solutions can be plugged in in a straightforward and modular manner; and (2) should implement as much language-independent (in other words, multilingual or cross-lingual) logic as possible, in order to reduce the workload of local NLP solution providers. The solution provided as part of the InteropEHRate project divides NLP pipelines into a language-specific *upstream preprocessing* part, and a language-independent *downstream disambiguation* part. Only the upstream part needs to be adapted to each language, as the downstream part works in a knowledge-based and language-independent manner. More precisely, the language-specificity of the downstream part is implemented as part of the formal knowledge, and not as part of the NLP solution.

Pipeline components. Again, the actual NLP pipeline components used greatly depend on the overall design of the NLP solution, a myriad of approaches existing today. In a general, multilingual context, where no major assumptions can be made about the structure (simplicity or complexity) of the language to be processed, the following NLP pipeline components appear as necessary:

1. **language detector:** automatically detects the language in which the input text is written;
2. **sentence detector:** detects sentence boundaries;
3. **tokeniser:** detects word boundaries;
4. **lemmatiser:** converts inflected words to their dictionary form;
5. **multiword detector:** detects multiword expressions;
6. **named entity recogniser:** detects names inside the text;
7. **concept disambiguator:** links words or expressions to their meanings, represented as language-independent concepts in the background knowledge;
8. **named entity disambiguator:** links names to the data records or objects they denote, represented as entities in the background knowledge;
9. **relation disambiguator:** provides the interpretation of the two kinds of links above, in terms of an attribute-value relationship, where the attribute is taken from the background knowledge.

Precision over recall. Given that information extraction is fundamentally an optional component within the data integration process, and given the nature of healthcare data processing (where data corruption or the introduction of false data can have serious consequences), an important design feature of NLP pipelines should be to favour precision over recall. This means that it is generally and relatively better to miss (fail to extract) certain pieces of information than to extract them in an incorrect way. When designing NLP pipeline components, this design principle should be taken into account.

4.5. Integration with the InteropEHRate Health Services

The deployment of a complete Information Extractor Tool (as depicted in Figure 7) and its integration with the rest of the InteropEHRate Health Services depends on a set of prerequisites.

Text extraction from health record documents. Inside electronic health records, natural language text may be embedded either inside data structures---e.g. spreadsheet cells---or inside unstructured documents, e.g. DOC or PDF format. As the NLP solution described above takes plain text as input, the pieces of text need to be extracted from such documents and fed to the Information Extractor Tool. In the solution implemented by the InteropEHRate project, for text in structured data, text extraction is done by the Data Mapper Tool. For unstructured documents such as PDF, however, a separate solution is needed. This task is not considered as part of Information Extraction and needs to be solved by a stand-alone tool as a preliminary processing step. Multiple tools (both free and for-pay) exist for text extraction from PDF and other popular document formats.

Knowledge setup. In order for the Information Extractor Tool to be able to disambiguate terms, names, and relationships, it needs to have a reference repository for such knowledge units. This repository, depicted as "EHR Knowledge" in Figure 7, needs to be populated before the Information Extractor can be used. The relevant knowledge (terminology, codes, names, data attributes) needs to be pre-loaded into the knowledge base, including both the possible words and expressions that can express the term in the local

language and their underlying language-independent concept. For example, in an English context, for a concept representing high blood pressure, the following "words" may need to be associated as formal knowledge: "*hypertension*", "*high blood pressure*", as well as "*I10*" (as the corresponding ICD-10 code).

NLP pipeline and component adaptation. The NLP pipelines and pipeline components enumerated above need to be implemented for each language and, to a lesser extent, adapted to each hospital context. Generally, the better this adaptation, the more accurate the information extraction results will be. Supposing that a generic medical NLP pipeline implementation already exists for the local language, adaptation may consist of extending the supporting dictionaries and corpora (e.g. lemmatisation dictionary, gazetteers/name lists) and also of writing custom rules (e.g. regular expressions) for the extraction of specific kinds of information.

4.6. Processes: Bootstrapping and Automated Execution

The Information Extractor Tool, similarly to the Data Mapper Tool (described in [D5.8]), is used in two different ways in two distinct phases of execution:

1. **bootstrapping phase:** the NLP pipelines are adapted, the knowledge updated, and the execution of information extraction incorporated into the *data integration recipe* (described in [D5.8]) using the Data Mapper Tool;

2. **execution phase:** information extraction is executed on legacy health records in a fully automated manner, either orchestrated by the Data Mapper Tool (itself running in fully automated mode) or called independently by the InteropEHRate Health Services.

4.6.1. Bootstrapping Phase

Bootstrapping of the Information Extraction Tool consists of the following operations, ideally in the following order:

1. knowledge update;
2. NLP pipeline adaptation and unit testing;
3. incorporation of information extraction into the data integration process.

The operations are executed by the Data Scientist, in the same way as in [D5.10]. The only exception is NLP pipeline adaptation and testing: this operation ideally requires an NLP engineer trained in natural language engineering. Without NLP adaptation, an "out-of-the-box" NLP pipeline may still work, but may not be perfectly adapted to the requirements of local health records.

4.6.1.1. Knowledge Update

Knowledge update consists of verifying that the concepts, entities, and attributes to be extracted from health records exist as formal knowledge within the EHR Knowledge Base. In the case of missing concepts, lexicalizations (words) in the local language, or entities, the Knowledge Base needs to be updated accordingly, as described in deliverables [D5.8] and [D5.10]. The risk of not having a properly updated knowledge base is simply that the incriminated concepts (entities, etc.) will not be identified from the text. Such missing knowledge issues can, of course, be rectified at any moment during the execution of the system, as part of *knowledge evolution* [D5.10].

4.6.1.2. NLP Pipeline Adaptation

NLP pipeline adaptation consists of one or more of the following:

- the extension of existing NLP pipeline components by new functionalities (such as the recognition of a new type of name, or the handling of a new grammatical feature);
- the insertion of a new pipeline component in order to execute a new, yet unsupported information extraction subtask (such as the recognition of dates and reasoning with temporal data extracted);
- the definition of a new pipeline that executes a new information extraction task (such as a pipeline dedicated to parsing prescriptions).

The adaptation of the NLP pipelines is a software development task that should follow the regular software development life cycle, including the unit testing of the new NLP functionalities.

4.6.1.3. Incorporation into the Data Integration Process

The Data Mapper Tool has an inbuilt functionality of being able to call the Information Extraction Tool on pieces of text extracted from structured data. (Note that "structured data" may also be a simple table that contains paragraphs of text previously extracted from unstructured documents.) This is because the Data Mapper Tool is compiled to use the Information Extractor Tool through its Java APIs. When used in this modality, there is no need for any extra step of "plugging in" the Information Extractor Tool into the hospital system, other than setting up its configuration parameters and executing the adaptation steps above.

While using the Data Mapper Tool when defining the data integration recipe [D5.10], the data scientist needs to define precisely what kind of NLP operation to execute on which text(s):

- the NLP pipeline to execute: for example, execute a dedicated *PrescriptionExtractor* pipeline;
- the data value(s) the pipeline should be executed: for example, over the values of an attribute containing textual prescriptions;
- the input constraints to the pipeline, such as the extraction of drugs of only a specific kind.

These input settings are defined through a dedicated user interface implemented within the Data Mapper Tool.

The Information Extractor Tool, however, can also be used as a stand-alone library, called independently of the Data Mapper Tool. This modality can be used to implement programmatic data integration that does not foresee the intervention of a data scientist through an interactive tool, or to integrate information extraction with other tools. In this case, the library can be called through its Java or RESTful web API.

4.6.2. Automated Execution Phase

Once the data integration pipeline is fully set up and is running in "production" mode, i.e. converting health records or research data on the fly and in a fully automated manner, the execution of information extraction is also carried out without any human intervention. The caller of the Information Extraction Tool

can be the Data Mapper Tool running in fully automated mode, or any other tool or library performing data integration within the IHS.

The performance, in terms of precision and recall, of information extraction is typically much below 100%. It is in the bootstrapping phase that the NLP developer can evaluate the typical performance of the pipelines and decide whether they are performant enough for production use.

The Information Extractor Tool is not foreseen to send notifications to the data scientist (beyond the standard logging operation of software systems). It is up to the caller, such as the Data Mapper Tool, to interpret the results of the Information Extractor Tool as either success or failure and signal to the data scientist any result that seems anomalous and could require human intervention.

5. RELATED WORK AND OUR PROGRESS BEYOND IT

Translation. Contemporary IT systems that offer some form of automation with respect to the translation of text implement either *computer-aided (manual) translation* or a form of *machine translation*. The former approach is usually based on *translation memories* and a human translation expert that applies translated sentences stored in the translation memory in case of near-perfect matches with sentences of the input text. Within machine translation (MT in the following), many approaches exist; in increasing order of complexity: dictionary-based, rule-based, statistical, and neural. At the moment of writing this deliverable, state-of-the-art research is entirely dominated by neural approaches, based on deep learning architectures trained on so-called *parallel* or *comparable corpora* (i.e. the same or similar texts in two languages) [Wu et al., 2016]. When trained on vast amounts (several gigabytes) of text, neural MT systems have proven to be by far the most effective; however, they struggle when not trained on adequately-sized corpora. So far, in real-world multi-sentence translation tasks, only neural systems have been capable of providing a translation quality judged acceptable by humans, and only on specific language pairs (those with the largest training corpora) and on text that was not too complex.

While efforts do also exist on domain-specific (including healthcare-specific) MT, they have not yet reached a level of precision that would be sufficient in "mission-critical" circumstances, such as the translation of clinical documents [Laï et al., 2020, Vieira et al., 2020]. However, progress in the field of MT has been so fast in the last 10 years that the prospect of applying such a system in a healthcare context is not to be entirely excluded in the midterm (i.e. within the next decade). This explains our decision to consider MT as a possible solution within an IT system supporting cross-border data interoperability, provided that it is made very clear to users that translation results are merely informative and should not be trusted as clinically precise. That said, due to the high barrier of entry with respect to the development of such systems---state-of-the-art MT has become the game of players such as Google and Microsoft---we did not deem it realistic to tackle it as a minor, task-level research topic within InteropEHRate. Hence our choice of relying on third-party MT systems for the translation of sentence-level and longer pieces of free text within health records.

Structured health data, however, frequently contains short pieces of natural-language text, in the form of single words or multi-word domain terms (such as "sample haemolysed" or "cerebral infarction"). Such terms can be efficiently translated by dictionary-based and rule-based systems, in some cases even more efficiently than by neural MT systems (due to the lack of textual context). As a special case of the dictionary-based approach, in such cases we apply *knowledge-based translation*, where the dictionary is a formal knowledge base where multilingual terminology is provided by international standards such as ICD, LOINC, or SNOMED. Our approach is comparable to the state of the art, equivalent to translation approaches using the [UMLS] system. A notable difference with respect to an UMLS-based translator, however, is that the knowledge base in our solution is within the control of the institution where it is deployed, such as a hospital. Using the Knowledge Management Tools that are part of the InteropEHRate solution [D5.8], the extension of the knowledge base by a missing translation becomes a trivial operation that a data scientist in charge of knowledge management can execute on the fly and with immediate effect over the data integration results.

Information extraction (IE) is a much-researched topic, overlapping in many ways with similar areas such as *text mining* and *information retrieval*. The specificity of IE is its focus on generating structured data from

unstructured text. In the even more specific case of *knowledge extraction* (KE) that we are applying in the InteropEHRate project, the pieces of information extracted are linked to formal knowledge and the output structures are used to build a *knowledge graph*. State-of-the-art KE uses classic NLP tasks such as *word sense disambiguation*, *named entity disambiguation*, and *relation extraction* in order to identify subject-predicate-object relationships within the text. As in the case of machine translation, current NLP research mainly focuses on the application of deep learning techniques to these tasks, moving towards the replacement of the classic NLP pipeline architectures by "end-to-end" approaches where a deep neural network replaces most pipeline components. The current approach (which may change from one year to another given the extremely fast recent evolution of the field) is to take a *language model* pre-trained on huge unsupervised corpora, and to *fine-tune* it through supervised or semi-supervised training with respect to concrete tasks, such as named entity disambiguation. The problem of supporting multiple languages is simply handled by using a language model pre-trained over a multilingual corpus, which is not more than a mixture of monolingual corpora, combined with language-specific fine-tuning.

In complete contrast with the approach described above, real-world IE projects in the healthcare field use more conservative approaches [Wang et al., 2018]. The main reason is precision: deep-learning-based methods rarely achieve the level of precision required by healthcare applications, and this is especially true for multilingual systems where results well below 80% of precision are published as groundbreaking. Another important reason for deep learning not having penetrated real-world IE for the healthcare field is the black box problem: when the neural system makes a mistake, usually no explanation can be given as to why, and there is no straightforward incremental way to fix it. For these reasons, AI-based solutions are not yet fully trusted by healthcare professionals [Laï et al., 2020], and NLP systems specifically designed for healthcare, such as [Apache cTAKES], maintain the classic pipeline architecture with separate components performing tokenisation, part-of-speech tagging, lemmatisation, named entity recognition, etc. Machine learning is sometimes used within individual components (such as to discover token boundaries or named entity types), but the application of rules (e.g. regex-based) remains fundamental, as it allows a fine-grained (although admittedly not particularly robust) control over precision and recall. In such solutions, multilinguality is addressed through NLP pipelines developed separately for each language.

The solution presented in this deliverable is targeting real-world applicability, and is thus closer to the latter approach. It goes beyond the state of the art in three aspects: (1) in the manner it tackles the problem of multilinguality; (2) in its adaptation to processing short pieces of text; and (3) in its integration with the semi-automated data mapping tool.

Contrary to both conventional monolingual NLP and to the multilingual language models of deep neural networks, we use a knowledge-based cross-lingual approach to the disambiguation tasks. Our NLP pipelines consist of an upstream preprocessing part that is language-specific, up to the lemmatisation task, and of a downstream disambiguation part that is language-independent and can thus be shared across languages [Bella et al., 2016]. This is achieved through the disambiguation tasks happening over formal, language-independent data representations. The advantage is that the adaptation of the NLP system to a new language becomes a smaller effort that is concentrated on adapting the preprocessing part of the pipeline.

The adaptation to short pieces of text is crucial in the case of processing natural language contained within structured data: here, text is typically not longer than single phrases and often follows a non-standard grammar that is called the *language of data* [Bella et al., 2020]. The contextual information that helps in disambiguating such text is not contained in the textual context but in the data structure surrounding it.

Our progress beyond the state of the art lies in the exploitation both of the specific grammar of the language of data and of the context provided by the data structures, in order to increase precision.

Finally, the integration of the NLP solution described above with the Data Mapper Tool---the interactive tool that allows local data scientists to automate the data integration process, described in [D5.8]---is, to our knowledge, unique among similar "ETL" or "data cleaning" tools. Such tools operate on the syntactic level, allowing structural transformations and the cleaning of data values. It is only the Data Mapper Tool that is capable of defining semantic transformations over data values, including information extraction from natural-language text, and thus convert unstructured information into structured data in an interactive manner. The Data Mapper Tool records the semantic transformations defined by its user, and can subsequently replay them to process health records in a fully automated manner.

6. CONCLUSIONS AND NEXT STEPS

We presented the specifications for natural-language translation and information extraction services targeting cross-lingual interoperability of electronic health records. As the InteropEHRate project specifies and develops solutions that are supposed to be integrated with the existing infrastructure and methods of healthcare organisations, the services presented in this deliverable have been designed for adoption and usability “in the real world”.

In the second, last version of this deliverable, improved final specifications will be provided. For natural-language translation, we expect further progress on the integration of translations into FHIR resources and on the handling of multilingual FHIR: while this deliverable is already presenting a valid method for the incorporation of translations into FHIR resources, we have yet to test the validity and adequacy of these solutions for all cases of translation. For machine translation, we are currently using a third-party service for testing purposes. For the next version of the deliverable, we will perform evaluations on the performance of multiple existing third-party systems for the health domain and will report on our results.

Regarding information extraction, we expect specifications to evolve based on experience gathered on the v1 implementation. This is particularly the case for NLP pipeline and component design, as this aspect of the work is developed in an experimental manner. More details will also be provided on special-purpose pipelines for solving specific extraction tasks (e.g. extraction of prescriptions), as well as on the setup and adaptation of pipelines to new languages.

REFERENCES

- **[D5.8]** InteropEHRate Consortium, D5.8-Design of the Data Integration Platform, 2021. www.interopehrate.eu/resources
- **[D5.10]** InteropEHRate Consortium, D5.10-Design of the Data Mapper and Converter to FHIR, 2021. www.interopehrate.eu/resources
- **[UIMA]** The Apache UIMA (Unstructured Information Management Applications) Project. http://uima.apache.org
- **[SemText]** The SemText NLP annotation specifications. http://opendatatrentino.github.io/semtext/
- **[Apache cTAKES]** The Apache Clinical Text Analysis and Knowledge Extraction System. http://ctakes.apache.org
- **[Wu et al., 2016]** Wu, Y. et al. *Google's neural machine translation system: Bridging the gap between human and machine translation.* arXiv preprint arXiv:1609.08144. 2016.
- **[Laï et al., 2020]** Laï, M.C., Brian, M. and Mamzer, M.F. *Perceptions of artificial intelligence in healthcare: findings from a qualitative survey study among actors in France.* Journal of translational medicine, 18(1), pp.1-13.
- **[Vieira et al., 2020]** Vieira, L.N., O'Hagan, M. and O'Sullivan, C. *Understanding the societal impacts of machine translation: a critical review of the literature on medical and legal use cases.* Information, Communication & Society, pp.1-18. 2020.
- **[Wang et al., 2018]** Wang, Y. et al. *Clinical information extraction applications: a literature review.* Journal of biomedical informatics, 77, pp.34-49.
- **[Bella et al., 2016]** Bella G., Zamboni, A., and Giunchiglia, F. *Domain-Based Sense Disambiguation on Multilingual Structured Data.* Proceedings of the ECAI 2016 workshop on Diversity Aware Artificial Intelligence, The Hague, Netherlands.
- **[Bella et al., 2020]** Bella, G., Gremes, L., and Giunchiglia, F. *Exploring the Language of Data.* Proceedings of the 28th International Conference on Computational Linguistics. 2020.
- **[UMLS]** The Unified Medical Language System. https://www.nlm.nih.gov/research/umls