

Title:
Title: A5.2-D2 Language Transformer
Author(s)/Organisation(s):
Marek Brylski/Intergraph, Dániel Kristóf and László Bérces/FÖMI
Working Group:
Architecture Team/WP05,
References:
A5.2-D2 Framework Specification

Short Description:
<p>This document is to be used to describe the specification of a Language Transformer component developed as part of the HUMBOLDT software framework. For an overview of the entire framework, please refer to the main specification document A5.2-D2. Each service component specification follows the RM-ODP (ISO 10476), and is aimed at providing information on the responsibilities and collaborations with other components of the service component described herein.</p>
Keywords:
<p>Framework specification, logical architecture, physical architecture, requirements, use cases, Language Transformer.</p>

History:			
Version	Author(s)	Status	Comment
001	Dániel Kristóf and László Bérces, FÖMI	new	Requirements for the component.
002	Marek Brylski	Edit	Initial description of the component
003	Marek Brylski	Edit	Change to the latest template
004	Marek Brylski	Edit	More work on Computational Viewpoint
005	Marek Brylski	Edit	Information viewpoint added

Table of contents

1	<i>Introduction</i>	2
1.1.	Purpose of this document	2
1.2.	Abbreviations and Definitions used in this document	3
1.3.	Standards used in this document	3
2	<i>Enterprise viewpoint</i>	4
2.1.	Actors in this component	4
2.2.	Business process overview	4
2.3.	Scenario Integration	4
2.4.	Use Cases and/or Requirements	4
2.4.1	Functional Requirements	4
2.4.2	Non-Functional Requirements	6
3	<i>Computational viewpoint</i>	6
3.1.	Overview	6
3.2.	Description of Modules	7
3.2.1	Language Transformer	7
	1. Responsibilities of the Component/Module	7
	2. Collaboration	8
	3. Actions fulfilled by this Component/Module	8
	4. Interface overview	8
3.2.2	Language Transforming Processor	8
	1. Responsibilities of the Component/Module	8
	2. Collaboration	9
	3. Actions fulfilled by this Component/Module	9
	4. Interface overview	9
4	<i>Information Viewpoint</i>	9
4.1.	Language Transformer	9
4.2.	Language Transforming Processor	9

1 Introduction

1.1. Purpose of this document

This document describes the specification of a Language Transformer (LT), which is a part of the HUMBOLDT framework. This component offers the translation of the terms between different languages.

1.2. Abbreviations and Definitions used in this document

<i>Abbrev.</i>	<i>Name</i>	<i>Definition</i>

1.3. Standards used in this document

In this section, standards specific to the service component described are listed and it is also described shortly how and why they are used. As with the sections before, please don't repeat information available in the main document.

For the purpose of the creation of this document, the HUMBOLDT Architecture Team is utilizing UML and RM-ODP. Furthermore, OGC (Open Geospatial Consortium) specifications and ISO norms are used where in line with the prototype requirements and where applicable. For more detailed information on these, please refer to the WP03 Deliverables, especially A3.6-D2 (version 2 of the Handbook of Standards, which comes into force at the same time as this document).

2 Enterprise viewpoint

The Enterprise viewpoint describes the functional purpose of the component and what we are trying to accomplish by it.

2.1. Actors in this component

As this is a component that can be used only by other components, there are no direct actors connected to it.

2.2. Business process overview

The Language Transformer should be capable of transforming/translating all information that becomes visible to a user from one language to another (and vice versa). Language transformation might also be necessary if user input is given at a language different from the one used in data processing. Thus, language transformation should include the translation of messages (user interface – both input and output) as well as that of data (e.g. attribute names and values). Besides “classical” language transformation, it would also be important to handle other issues such as conversion between different date/time formats (or different units of measurement).

According to this general view, the actual problem is similar to those dealt with in the scope of „internationalization” (i18n) and „localization” (or L10n) widely used in software development [see e.g. http://en.wikipedia.org/wiki/Internationalization_and_localization].

In some simple cases, language translation can be solved by a thesaurus service containing the meaning of different words in multiple languages. However, taking into account the complexity of the problem, it is more straightforward to apply an extensible solution. On the other hand, localization problems (e.g. transforming date and time formats, units of measures, etc.) cannot be solved by a simple word lookup and require appropriate handling.

2.3. Scenario Integration

The document “Humboldt Scenario: HS Border Security – System Specification” states the following information on the harmonization aspect of language translation:

Multi-linguality: Yes, the BS agencies in Slovakia and Hungary use their systems in different languages.

Following this harmonization aspect in Border Security scenario the use case called “UC03_01 Data exchange” needs language translation component to be used so that the administrator is able to prepare the harmonized data to be used in case of intrusion.

2.4. Use Cases and/or Requirements

As this is a component that can be used only by other components, there are no direct use cases connected to it.

2.4.1 Functional Requirements

The following functional requirements affect the Language Transformer component.

Requirements applying to this Component

- ◆ The required capabilities of the Language Transformer can be defined at different levels:

- Output all language-dependent information that becomes visible to the user (e.g. displayed or written in a file) in a given language, with unambiguous and highly consistent terminology. This includes the messages and text displayed on the user interface, and also the data transformed to fit within a certain context.
- Translate user input from the user's language to the language which is used for the processing of the dataset (e.g. translating a user query from "rue" to "street" if the base data set uses this terminology and it is used for processing...). It should also be capable of converting date, time, etc. formats (e.g. 07/10/08 -> 2008.10.07).
- Translate the terms that flow inside of the harmonization framework, for example the selected terms in the service requests and responses.
- ◆ It is evident that in many cases no one-to-one connection will exist between meanings in different languages. Therefore, we should keep the possibility of entering one-to-multiple and multiple-to-one connections, with a possibility of prioritisation.
- ◆ Automate the translation but keep the user control as much as possible. Use predefined templates for the output wherever possible (e.g. GUI messages). Identify the elements that cannot be translated using the existing solutions/templates, and provide the user with a possibility to input the required data (interface for typing in by hand, point to an existing template file, etc.).
- ◆ Store the user input / translations in templates that can be reused in similar translation processes. Store the templates in a central repository (Template Repository) that can be used for enhancing the translations via a search engine. Let the user choose the appropriate translation for the required elements. In case of ambiguity (multiple hits), order the hits by relevance (number of hits). With the number of templates in the repository increasing, the system will contain more and more existing solutions over the time, thus helping the users with pre-existing knowledge.
- ◆ Make the translation context-sensitive. Several context elements handled by the HUMBOLDT Context Service can serve as a basis for context-sensitive filtering, e.g. the topic and the language.
- ◆ Make it possible to retrieve translations from existing HUMBOLDT-external web-based services, such as the GEMET thesaurus directly referred to in INSPIRE (<http://www.eionet.europa.eu/gemet>). However, these translations should be reviewed and approved by the user.

There are several problems that fall behind the scope of the Language Transformer and thus it should not deal with them:

- ◆ It should not retrieve the list of language elements to be translated. The Language Transformer should always be invoked by another component with a specific task defining exactly what to translate and how.
- ◆ It should not convert units of measurement, or only with restrictions. The reason for this is that such a conversion might lead to inaccuracies in many cases.
- ◆ It should not resolve complex semantic problems.
- ◆ It should only provide translations based on user-entered information, existing templates, or external web services (e.g. GEMET) and should be controlled by the user, so it should not solve translation problems "by magic".

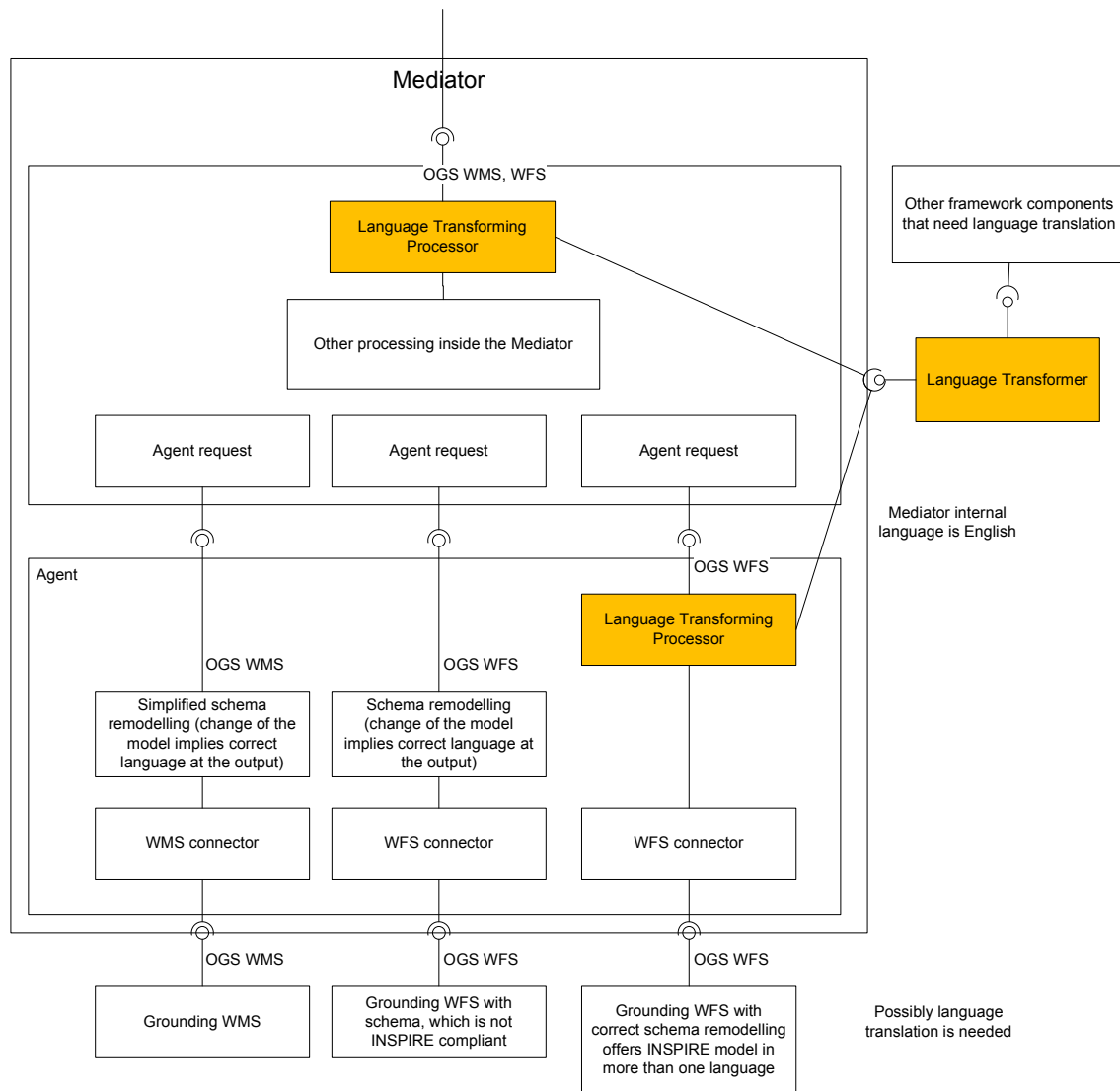
2.4.2 Non-Functional Requirements

- ◆ Following the idea of being an independent component, called by other components, Language Transformer does not solve by itself the translation inside the Mediator processing queue. Therefore another component is needed: Language Transforming Processor, which will be responsible for the translation of service requests and responses.
- ◆ In order to keep the independence of this component and make it loosely coupled, the Language Transformer should be available to other components in a form of Web Service.

3 Computational viewpoint

3.1. Overview

The Language Transformer should always be invoked by another component with a specific task defining exactly what to translate and how. It should be up to the other components to intelligently derive what to translate and call the LT with this specific task. Translating an entire GML file could be, horrible and also useless. Instead, another component should take care of identifying which are the language elements to translate and call the LT - possibly with an arrayed request if there are many items to translate. In most cases it is really just a small part of a dataset that should be translated at once, e.g. the attribute names or the labels that are really going to be displayed. On the other hand if the translation is a prerequisite for processing (e.g. querying two datasets for a given language-specific string) this decision could be left to the grounding service if a correct language version is not available.



The above diagram shows the position of the language translation components in the overall architecture and also explains the relation between them. Language Transforming Processor is the component responsible for the identification of the elements from web service requests/responses, that should be translated. Language Transformer itself performs the translation.

3.2. Description of Modules

3.2.1 Language Transformer

Full Name: Processing → Language Transformer (LT)

1. Responsibilities of the Component/Module

- ◆ This component handles the translation of the terms inside HUMBOLDT framework. It is called by other components that need the translation. It is the responsibility of those other components to know which terms should be translated.

2.Collaboration

- ◆ This component is used by other components in the framework. It can be called by any component that needs the language translation, therefore a complete list of these components cannot be provided here. An example of the calling component is the Language Transforming Processor.
- ◆ This component is used by the Language Transformer GUI, which enables the definition of the translations. The following rules apply to the GUI commands:
 - if the translation is not available in the desired language but it is in English, return English and inform the user about it, give him the possibility to provide the translation manually, and save the input in a template;
 - if even English translation is lacking, return an error and let the user enter the translation manually (via a user interface), and save the input in a template.

3.Actions fulfilled by this Component/Module

- ◆ This component performs a translation of terms.

4.Interface overview

- ◆ This is a web service with its own interface. It is not an OGC web service.
- ◆ Language Transformer needs an interface to provide the translation (create, update, delete).
 - putTranslation (sourceLanguage, destinationLanguage, termInSourceLanguage, termInDestinationLanguage, ranking, domain), creates if it does not exist or updates, for the moment only one term in a given domain/language is allowed. This term has a ranking assigned to it.
 - deleteTranslation (language, termLanguage, domain), deletes the translation. If main entry is being deleted, all translations referring to it, should also be deleted.

The list of European languages is pre-configured in the component and thus not editable via its interfaces. In future versions an additional possibility to extend the language list with other languages should be considered.

- ◆ The component needs an interface to perform the translation. The relation between the terms is many to many so the interface has to provide operations to retrieve either all available translations along with their ranking or the best possible translation.
 - getTranslation (sourceLanguage, destinationLanguage, termInSourceLanguage, termInDestinationLanguage, domain), retrieves a translation in a given domain.
 - getAllTranslations (sourceLanguage, destinationLanguage, termInSourceLanguage, termInDestinationLanguage), retrieves a collection of translations for all domains,
 - getBestTranslation (sourceLanguage, destinationLanguage, termInSourceLanguage, termInDestinationLanguage), retrieves a translation with highest ranking

3.2.2 Language Transforming Processor

Full Name: Processing → Language Transforming Processor (LTP)

1.Responsibilities of the Component/Module

- ◆ This component handles the translation of the requests and responses of the services called inside the mediation tier of HUMBOLDT framework.

2.Collaboration

- ◆ This component calls the Language Transformer component to perform the real language translation.

3.Actions fulfilled by this Component/Module

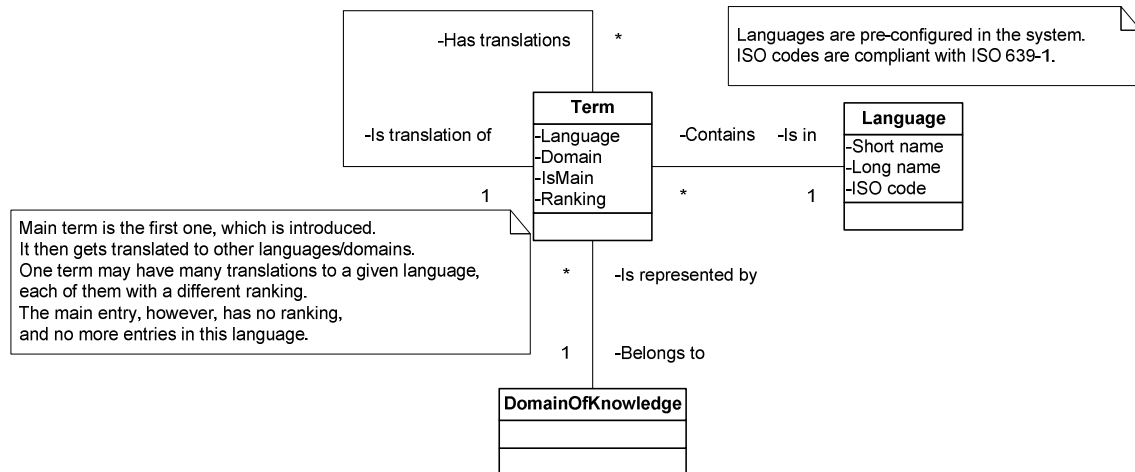
- ◆ This component performs a translation service requests and responses.

4.Interface overview

- ◆ **Fixme.** The interface of this component is the same web service interface as the grounding services the framework is able to connect to. Thus it follows WMS, WFS and WCS interfaces. Should this be a WPS instead?

4 Information Viewpoint

4.1. Language Transformer



4.2. Language Transforming Processor

The Language Transforming Processor translates the following terms:

- ◆ Feature class name,
- ◆ Attribute name,
- ◆ Attribute values under certain conditions, for example translation of GetFeatureInfo response, which carries a small amount of data can be done in a reasonable time and therefore should be performed.
- ◆ Mediator internally decides for each service type, what exactly is translated.