Resource Sharing by Multilingual Expression Services

Masaki Gotou¹, Hirofumi Yamaki², Daisuke Yanagisawa³, Masamitsu Ukai⁴, Masahiro Tanaka¹, and Toru Ishida¹

¹Department of Social Informatics, Kyoto University,
²Advanced Networking Division Information Technology Center, Nagoya University
³Graduate School of Information Science, Nagoya University
⁴Department of Information Engineering, Nagoya University

Abstract

Multilingual expression services (MESs) are available in many multicultural fields, such as education, medical care, and disaster prevention. In Japan, for example, local governments and NPOs create parallel-texts and provide expression services in such fields. However, these services present a common difficulty: the lack of parallel-texts. Service-computing technology has the potential to allow them to collaborate. In this paper, we show how Service Oriented Architecture (SOA) can be applied to multicultural activities. As a first step, we propose an architecture, in which language resources are available as web services, and MESs can obtain parallel-texts from the language web services. To achieve this, RDF metadata are attached to parallel-texts and stored in a separate RDF repository, with which users can describe what they need in detail. This arrangement allows MESs to select only the contents they need from the shared resources and show them to users. Our implementation confirms that the proposed architecture works well with two real services in use.

1. Introduction

With the background of globalization and the spread of the Internet, multicultural fields and opportunities for intercultural communication are increasing; the result is more people are encountering the language barrier. When a group member does not have multilingual skills, language resources such as multilingual glossaries and parallel-texts, built up by communities, come into play. In this paper, we call the services that present language resources in a easily reusable form “multilingual expression service” (MESs). MESs are currently being used in various fields, such as education, medical care, and disaster prevention.[1, 2].

These MESs face a common difficulty: they do not own enough parallel-texts to allow users to express what they want to say in the necessary language. This means that an expert who can use each language is necessary to offset the language resource shortfall. This approach, however, is constrained insufficient experts and/or budgets. To overcome the lack of language resources, we can apply service-computing technologies to the MESs developed independently, and let them share their resources.

In this paper, we propose a design based on the Service Oriented Architecture. In the design, language resources in each MES are separated from the MES and wrapped as web services which allow them to be shared among MESs. We call a web service that wraps language resources in a MES a “language web service” (LWS). What we aim is an architecture that allows the resources of hundreds of MESs to be shared; LWSs are expected to offer tens of thousands of parallel-texts which will be enough for users to express their intention accurately in various languages. As the first step in achieving the architecture, a framework by which people or communities can publish their own language resources as web services is needed.

Frameworks which allow people to publish services as web services are already proposed. A framework that supports service discovery and data integration by annotating services is proposed in [3]. [4] proposes to use web services and XML to realize data integration. As such a framework specialized for language services, the Language Grid [5] has been developed.

None of the existing frameworks, however, are sufficient to support LWS publication and usage. MESs still cannot accurately pick up the parallel-texts their users need from published LWSs. The reason for this problem, which we focus on in this paper, is the diversity of the ways in which MESs categorize their language resources.

To solve this problem, this paper extends the Language Grid and proposes an architecture in which a single RDF repository is shared as a web service outside the MESs and LWSs. People can attach metadata to parallel-texts in LWSs and store the metadata in the RDF repository. In order to achieve scalability, this architecture satisfies two requirements: (R1) Preservation of the original structure of LWSs and (R2) Ease of maintenance of MESs”.

We experimentally implement this architecture and integrate two real services in use in the field of education and disaster prevention. The results demonstrate that this architecture enables MESs to accurately pick up parallel-text that their users need.
The paper is organized as follows. We start Section 2 by describing two examples of MESs and introducing the Language Grid infrastructure; Section 3 describes the issue of sharing language resources; In Section 4, we introduce an architecture to resolve this issue. In Section 5, an experimental implementation of the architecture is introduced. Finally, conclusive remarks and further work are presented in Section 6.

2. Background

In this section we describe the MESs on which we focus using real examples in the fields of education in schools and disaster prevention. The Language Grid, an infrastructure for sharing and connecting language services, is also described.

2.1. Multilingual expression services

In recent years, the percentage of foreigners in Japan has been increasing, and the language barrier is now a serious problem in many multicultural fields, such as education, medical care, and disaster prevention. To break down the barrier, many local governments and NPOs create parallel-texts like “multilingual templates for teachers”. Supporting tools, including services that support searching and rewriting templates, have also been developed. We call such services multilingual expression services (MESs), which means services that help users to express what they want to say in many languages.

Large amounts of language resources and many MESs are currently available. For example, in Japan, even if we limit discussion to the area of school education, we found that many language resources have been created by school boards and NPOs in many localities including Kyoto, Osaka, and Aichi. “Site to support foreign students in their school lives” by Osaka prefecture, one such resource, contains hundreds of parallel-texts. The situation is similar in other fields. Tokyo International Communication Committee provides links to international organizations or tourist boards in Japan and has relationships with about one hundred countries, many of which provides glossaries or parallel-texts. If we include language resources that are not on the Internet, we estimate that the total number of resources lies in the hundreds and parallel-text expressions to be in the hundreds of thousands.

In this chapter, we introduce two examples of MESs that support users in searching and reusing parallel-texts.

Multilingual Disaster Information System developed by Nagoya University, is a tool that supports people to write documents for foreigners to prepare them for natural disasters. This tool contains about 350 parallel-texts and categorizes them into 37 categories such as “earthquake information”, “medical care”, and “school”. Users access a GUI to select categories, specify the search keywords and browse the parallel-texts that match the search conditions. In addition, users can rewrite some parts of the parallel-texts by filling text areas or selecting options from drop-down lists. The rewritten text can be translated into 6 languages, Japanese, English, Chinese, Korean, Portuguese and Spanish. (Figure 1)

Another tool is School Expressions Viewer developed by Kyoto University (Figure 2). With this tool, users can browse multilingual handbooks edited by the education board of Kawasaki city, where 2.11% of the population is foreign born, such as “Handbook for Teachers to Parent Letter Correspondence”. It contains more than 1,300 parallel-texts categorized into 48 categories including “field trip”, “holiday”, and “emergencies”. Users can search and use the parallel-texts in Japanese, Chinese, Korean and Spanish.

These two tools were developed with local input and are now available online as web applications. However, it is not easy to share language resources among these tools due to their different access interfaces.

2 http://www.tokyo-icc.jp/link/index.html
2.2. The Language Grid

The Language Grid [5] was developed as an infrastructure for intercultural collaboration; it allows people to easily develop new language services by combining existing ones available on the grid to satisfy their needs.

The main idea of the Language Grid is that existing language resources are published openly as web services, which we call “language web services” (LWSs) in this paper. What the Language Grid provides is low-level support for language resource sharing, such as a P2P grid infrastructure, and standardized interfaces for each type of language resource like “machine translation service”, “morphological analysis service” and “parallel-text service” in order to realize interoperability among services of the same type.

With the Language Grid, we can use the service oriented architecture to share language resources among MESs. In this architecture, an MES is a service that retrieves parallel-texts from language resources and shows them to users as needed. On the other hand, a LWS is a service which wraps a language resource, executes selection from it, and returns the result of the selection. MESs call some LWSs and show users the integrated form of the selections.

3. Issues on resource sharing

The advantage of MESs, compared to machine translation, is the quality of translation. That is because parallel-texts in language resources are translated by human experts. The disadvantage, on the other hand, is that MES expressiveness is greatly limited by the paucity of the base parallel-texts. Enriching the content of each language resource is the best way to overcome this disadvantage, but the cost of creating language resources is high. Thus, it is important to share language resources among MESs. The Language Grid is an infrastructure that allows language resources to be separated from MESs and published as web services, and MESs can freely call search methods to access their resources.

The problem is that, because each language resource has a different structure, MESs cannot accurately pick up the parallel-texts their users need from the published LWSs. In detail, each MES categorizes its language resources in a different way with the goal of help their assumed users appropriately.

In the example shown in 2.1, one of the MESs categorizes parallel-texts about school holidays as “school” while the other MES categorizes them as “holiday”. Moreover, these two categories, “school” and “holiday”, do not have exactly the same concept. For instance, “school” category contains a parallel-text that says “The application deadline for the entrance examination for university (NAME) has been extended (VALUE) days.” This sentence cannot be categorized as “holiday” in the other MES: School Expressions Viewer, because it is neither a sentence for pupils nor a sentence about a holiday.

To solve this problem, we propose a new architecture that extends an existing infrastructure to share LWSs. In this architecture, MESs can accurately retrieve parallel-texts which have the desired characteristics from shared LWSs. Moreover, in order to achieve scalability, the architecture is designed to satisfy the following two requirements.

(R1) Preservation of the original structure of LWSs

In order to smoothly achieve large-scale sharing of resources, reusing LWSs or technical know-how to develop LWSs which are available on existing infrastructures, is very efficient. Therefore, the architecture should not demand that the interfaces of LWSs and the structures of language resources be significantly changed.

(R2) Ease of maintenance of MESs

MESs must be able to import new language resources at low or almost no cost.

4. Resource sharing architecture

This section proposes our architecture proposal which aims to solve the problem faced by sharing language resources among MESs. The key idea is to create a common RDF repository as a service outside MESs and LWSs; it preserves the original LWS structures and simplifies MES maintenance.

4.1. Architecture

The proposed architecture is composed of three types of services as shown in Figure 3. LWSs provide an interface to search the language resources inside them; MESs retrieve parallel-texts from LWSs as users need; and the RDF repository stores metadata of the parallel-texts.

In the proposed architecture, services work as follows.

Step1 Metadata of parallel-texts are stored in the RDF repository.
Step2 An MES, in response to a user's request, sends a query to RDF repository that asks for the URIs of parallel-text that meet the user's need.
Step3 RDF repository executes the query and returns the URIs.
Step4 The MES invokes, for each LWS, the LWS's search method specifying keywords and candidate URIs.
Step5 Each LWS returns to the MES the parallel-texts it holds matching the URIs specified and that contain the specified keywords.
This architecture solves the problem described in the preceding section as follows.

The most important requirement, MESs should more accurately pick up appropriate parallel-texts, is satisfied by RDF annotation of each part of its language resources. We do not define any vocabulary framework in this paper, but we assume that metadata is available that can express more detailed information than category matching, such as characteristics or usages of parallel-texts. Annotated data can be searched using query languages such as SPARQL.

The following chapters describe in detail the services and show how (R1) the architecture preserves the original structure of LWSs and (R2) it makes MES maintenance tasks easier.

4.2. Language services

(R1) is met by extending each LWS to include a search method that takes URIs as arguments.

Table 1 ParallelTextServiceWithCandidates Interface

<table>
<thead>
<tr>
<th>Operation</th>
<th>searchWithCandidates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Message</td>
<td></td>
</tr>
<tr>
<td>sourceLang</td>
<td>String</td>
</tr>
<tr>
<td>targetLang</td>
<td>String</td>
</tr>
<tr>
<td>source</td>
<td>String</td>
</tr>
<tr>
<td>matchingMethod</td>
<td>MatchingMethod</td>
</tr>
<tr>
<td>candidateUris</td>
<td>String[]</td>
</tr>
</tbody>
</table>

LWSs implement the ParallelTextServiceWithCandidates interface shown in Table 1, which is an extension of ParallelTextService, the Language Grid standard interface for parallel-text services. The ParallelTextServiceWithCandidate interface includes the searchWithCandidates method. This method is an extension of the search method in ParallelTextService to receive the candidateUris argument. The candidateUris argument is a list of URIs; each corresponds to a parallel-text or a set of parallel-texts. MESs can narrow down the search range using the output of the RDF repository.

All that the LWS creator needs to do is assign a URI to each parallel-text (or set of parallel-texts) and to wrap the complete language resource (XML, CSV file, Database or so on) as a web service that provides the searchWithCandidates method.

4.3. RDF repository

(R2) is met as follows. The function to select parallel-texts with specified characteristics or usages among shared language resources is provided by the RDF repository. This enables MESs to extend automatically in concert through the increase in the number of language resources.

The RDF repository is also wrapped as a web service. For this implementation, the interface for the RDF repository provides query method as shown in Table 2. It takes four arguments, queryLanguage for specifying the language for query such as SPARQL, queryString for the text of the queries, infer to specify whether to use inference mechanism[^4] or not and limit to limit the number of results. This method returns the results of the query as a RDFQueryResponse object, which may have various styles according to the query language.

Table 2 RDFRepositoryService Interface

<table>
<thead>
<tr>
<th>Operation</th>
<th>query</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Message (name, data type)</td>
<td></td>
</tr>
<tr>
<td>queryLanguage</td>
<td>String</td>
</tr>
<tr>
<td>queryString</td>
<td>String</td>
</tr>
<tr>
<td>infer</td>
<td>Boolean</td>
</tr>
<tr>
<td>limit</td>
<td>Integer</td>
</tr>
<tr>
<td>Output Message (data type)</td>
<td>RDFQueryResponse</td>
</tr>
</tbody>
</table>

We assume that metadata stored in the repository should specify what the topic of the annotated parallel-text is, who it is intended to be used by, and its grammatical type (verb, single word, sentence, paragraph and so on).

The reason for using RDF to describe such things is that RDF enables fast integration of data sources by bridging semantic differences [6]. For example, when a user needs parallel-texts for pupil in general, the MES sends a query to get parallel-texts for “pupil”. The user receives parallel-texts that include some for “elementary school pupil”, which is a subclass of “pupil”.

[^4]: If infer is set to true, inferred RDF triples such as super-class categorization will be taken into account.
Figure 4 shows an example of RDF metadata. This example can be described as follows. The rdf:Description element states that following statements are about the parallel-text with address given by the URI i.e. "http://www.net.itc.nagoya-u.ac.jp/2008/05/mlii/Translations/146". The pt:Category statement specifies that the parallel-text is categorized under "school", which has the address "http://www.net.itc.nagoya-u.ac.jp/2008/05/mlii/Categories/School". The final pt:possibleReceiver statement shows that it was created for writing messages sent to pupils in Nagoya city.

```
<rdf:Description
  rdf:about="http://www.net.itc.nagoya-u.ac.jp/2008/05/mlii/Translations/146">
  <pt:Category
    rdf:resource="http://www.net.itc.nagoya-u.ac.jp/2008/05/mlii/Categories/School"/>
  <pt:possibleReceiver
    rdf:resource="http://www.net.itc.nagoya-u.ac.jp/2008/05/mlii/Receiver/NagoyaPupil"/>
</rdf:Description>
```

Figure 4 Example of RDF Metadata

5. Experiment

To demonstrate that the proposed architecture is effective for sharing language resources, we integrated two real services in use. In detail, we integrated the parallel-texts of the multilingual disaster information system into the school expression viewer introduced in Section 2.1. As the first prototype, does not employ RDF, we implemented the architecture that has LWSs and MESs interact with each other on the Language Grid infrastructure. In the architecture, language resources are wrapped as LWSs to support the search method by which other service can retrieve parallel-texts classified under the category that the language resource originally set.

Figure 5 is the result of the integration. Parallel-texts in “school” category are joined to parallel-texts in “holiday” category.

This result demonstrates the problem described in Section 3. This result for “holiday” contains a parallel-text that is for neither pupil nor holiday.

Not only that, this result shows another unexpected problem. Tags for rewritable parts like “[place1|place_name]”, which no MES except Multilingual Disaster Information System can interpret, are displayed to the user.

As the next step, we implemented the proposed architecture to test the RDF mechanism for language resource sharing. We assigned an URI to each parallel-text and wrapped language resources, originally in XML format, to implement the interface shown in Table 1. We implemented an RDF repository with the format shown in Table 2 by wrapping Allegrograph[7]. We stored in it sample RDFs including the RDF depicted in Figure 4.

MESs were implemented to execute the process described in Section 3. For example, if the user selected the “holiday” category via the GUI, the MES sends the query shown in Figure 6. This query means “Return the list of URIs corresponding to parallel-texts which are categorized as ‘holiday’, or which as ‘school’ and can be sent to ‘Pupil’”.

```
SELECT DISTINCT ?trans WHERE {
  { ?cat dc:title 'holiday' } UNION
  { ?cat dc:title 'school'.
  ?trans
    pt:assumedReceiver ?receiver . ?receiver
    rdf:type pt:Pupil} .
}
```

Figure 6 SPARQL query to get “holiday” data.

MESs receive the URIs of the suitable parallel-texts in the two language resources, and invoke searchFromCandidates operation using the URI list assigned to the candidateUris. Subsequently, they receive the parallel-texts, transform them and show them to users.

In addition, we rewrote tags in parallel-texts, which originally had various styles such as “[地点|地名]” into unified XML like `<pt:blank pt:type="place_name" pt:id="1">`, and we equipped MESs with the capability to transform the XML tags into texts the users could read.

```
The application deadline for the entrance examination for university [学校1] has been extended [日付1] days.
The following schools will reopen tomorrow: [学校1]
The following schools are closed today: [学校1]
```

Figure 7 Results of the proposed architecture
Figure 7 shows the results of the search to retrieve parallel-texts categorized as “holiday”. We recognize that, unlike Figure 5, no inappropriate results are displayed, and tags are changed into readable comments.

From this result, we can reach the following conclusions.

The proposed architecture solves the key problem detailed in Section 3. In other words, it enables MESs to share language resources and allows their users to get the parallel-texts they need more accurately than existing frameworks which only support publication services.

In the experiment, the effect of sharing resources is not so significant because the intersection of concepts covered by the language resources is small. However, as described in 2.1, the total number of language resources in various locations is quite large, and even more resources will be created in the future.

We found that the annotation process should be studied more carefully. The metadata in Figure 4 and queries in Figure 6 are attached by humans after careful observations on each parallel-text in each language resource. In other words, metadata were attached ad hoc.

The weakness of this annotation process is that automating this process is difficult because it requires a lot of human judgment. This weakness will delay the sharing of more resources.

We can say that more efficient annotation schemes are necessary to achieve scalability in the language resource sharing architecture. The difference of data formats among LWSs is another important issue to be resolved. Platforms to support document annotation through well-known ontology structures and that provide GUIs or APIs for annotators have been developed [8]. We will consider these platforms for language resource annotation.

6. Conclusion

In this paper we have presented a new language resource sharing architecture that extends existing infrastructures and enables multilingual expression services (MESs) to share their language resources. This architecture is based on the Service Oriented Architecture in which language resources are published as web services.

The problem is how to enable MESs to accurately pick up the parallel-texts their users need from published language web services (LWSs).

To solve this problem, a mechanism is needed to wrap the various categorization policies of each MES and LWS. It must also satisfy two requirements: “Preservation of the original structure of LWSs” and “Ease of maintenance of MESs”.

To achieve this, we proposed an architecture in which a single RDF repository is shared as a web service outside the MESs and LWSs. RDF metadata is attached to the parallel-texts in each LWS; they show what the parallel-text is about or what was intended to be used for.

MESs can send a query that describes the type of parallel-texts they want to find, and the repository returns the URIs of the appropriate parallel-texts. Passing candidate URIs to each LWS, MESs can narrow down the search range.

The proposed architecture meets the two requirements. LWSs required only a small change in their interface and MESs need no change after the queries are associated with user operations.

We have experimentally implemented this architecture and integrated two real services in use in the fields of education and disaster prevention. This experiment demonstrated that this architecture enabled MESs to accurately pick up parallel-texts from two LWSs.

Further studies on the annotation process are needed to improve the scalability of the architecture. Metadata used in the implementation was manual tagging by experts. In order to share hundreds of language resources among MESs, further research on schemes that can ease or automate the annotation process is needed.

Acknowledgment

This work was supported by Global COE Program "Informatics Education and Research Center for Knowledge-Circulating Society".

References