Consumo del servizio apertium-service attraverso il framework Okapi

Pasquale Minervini
p.minervini@gmail.com

1 Introduzione

Questo documento vuole fornire una breve descrizione di apertium-service, un servizio di traduzione automatica basato sulla piattaforma Apertium[1] (Armentano-Oller et al., 2005), del framework Okapi[2] e su come quest'ultimo può essere utilizzato per la fruizione di una varietà di servizi di traduzione automatica e translation memory[3].

2 Apertium-service

Apertium è una piattaforma free/open-source di traduzione automatica; una descrizione delle tecnologie che impiega, dei principi alla base della sua progettazione e delle evoluzioni che ha avuto negli ultimi anni è presente in Zubizarreta et al. (2009).

Utilizzando come base la piattaforma Apertium, è stato realizzato un servizio, chiamato apertium-service e descritto in Minervini (2009); tale servizio offre funzionalità di traduzione automatica e di riconoscimento della lingua, accessibili tramite il protocollo XML-RPC[4] (ma sono stati realizzati anche wrapper per l'accesso tramite SOAP[5] o REST[6]).

Le interfacce del servizio apertium-service possono essere riassunte da due funzioni, chiamate **Translate** e **Detect**, i cui prototipi sono descritti rispettivamente nelle tabelle 1 e 2.

[1] http://www.apertium.org
[2] http://okapi.sourceforge.net/
[3] http://en.wikipedia.org/wiki/Translation_memory
[4] http://en.wikipedia.org/wiki/XML-RPC
[5] http://en.wikipedia.org/wiki/SOAP
[6] http://en.wikipedia.org/wiki/Representational_State_Transfer

parameters	`testo`
	`lingua di origine`
	`lingua di destinazione`
returns	`traduzione`
	`lingua di origine riconosciuta`

Table 1: Parametri e valori di ritorno del metodo **Translate**.

parameters	`testo`
returns	`lingua riconosciuta`

Table 2: Parametri e valori di ritorno del metodo **Detect**.

- **Translate** riceve tre parametri chiamati `testo`, `lingua di origine` e `lingua di destinazione` che rappresentano, rispettivamente, il testo da tradurre, la lingua utilizzata in tale testo e la lingua in cui lo si desidera tradurre, e restituisce un valore di ritorno chiamato `traduzione` e contenente il testo tradotto; nel caso il parametro `lingua di origine` fosse omesso, viene effettuato un riconoscimento automatico della lingua, e questa viene restituita in un ulteriore valore di ritorno, chiamato `lingua di origine riconosciuta`.

- **Detect** riceve un parametro chiamato `testo`, contenente testo libero, e restituisce un valore `lingua riconosciuta`, contenente la lingua utilizzata dal testo.

Inoltre, il servizio fornisce un ulteriore metodo, chiamato **Language Pairs**, che restituisce una sequenza di tutte le coppie di lingue supportate dal sistema di traduzione, ognuna rappresentata dalla coppia `lingua di origine` e `lingua di destinazione`.

In tutti i metodi, le lingue sono rappresentate dal corrispondente codice ISO 639-1 (ISO:639-1, 2002).

In figura 1 viene illustrato come invocare una istanza del servizio apertium-service presente all'indirizzo `http://xixona.dlsi.ua.es:8080/RPC2` attraverso il protocollo XML-RPC.

Informazioni sull'installazione e configurazione di apertium-service sono disponibili sul wiki del progetto Apertium all'indirizzo `http://wiki.apertium.org/wiki/Apertium-service`.

```
>>> import xmlrpclib
>>> proxy = xmlrpclib.ServerProxy('http://xixona.dlsi.ua.es:8080/RPC2')
>>> print proxy.translate("Test for the machine translation service",
        "en", "es")["translation"]
        Prueba para el servicio de traducción automática
```

Figure 1: Esempio – Invocazione di apertium-service dalla shell di Python tramite XML-RPC.

3 Okapi

Il framework Okapi fornisce una serie di strumenti che possono essere utilizzati in qualsiasi applicazione che include attività di traduzione e di localizzazione, accorciandone i tempi di sviluppo e migliorandone l'interoperabilità.

Tra le altre cose, Okapi fornisce un insieme di connettori per consentire l'interazione con una ampia varietà di servizi di traduzione automatica e translation memory, come Google Translate[7] GlobalSight[8], MyMemory[9] e OpenTran[10].

In Okapi, i connettori per i servizi di traduzione automatica implementano l'interfaccia `IQuery`[11], mentre i connettori per i servizi di translation memory implementano l'interfaccia `ITMQuery`[12]; questo rende possibile lo sviluppo rapido di sistemi in grado di interagire con una molteplicità di servizi di traduzione automatica e translation memory, astraendo dall'interfaccia e dal protocollo di comunicazione utilizzati da ogni particolare servizio.

In figura 2 viene illustrato un possibile utilizzo del connettore Okapi per l'interazione con apertium-service (presente nei sorgenti del servizio stesso). In questo caso, per interagire con il servizio Google Translate anziché con il servizio apertium-service è sufficiente sostituire il connettore `org.apertium.okapi.ApertiumXMLRPCMTConnector` con il connettore `net.sf.okapi.connectors.google.GoogleMTConnector`, lasciando inalterato il resto.

[7] `http://translate.google.com`
[8] `http://www.globalsight.com`
[9] `http://mymemory.translated.net`
[10] `http://open-tran.eu/`
[11] `http://okapi.opentag.com/javadoc/net/sf/okapi/lib/translation/IQuery.html`
[12] `http://okapi.opentag.com/javadoc/net/sf/okapi/lib/translation/ITMQuery.html`

```
import net.sf.okapi.common.*;
import net.sf.okapi.lib.translation.*;
import org.apertium.okapi.*;

public class TestCase {
        public static void main(String[] args) {
                IQuery conn = new ApertiumXMLRPCMTConnector();
                Parameters p = new Parameters();
                p.setServer("http://localhost:6173/RPC2");
                conn.setParameters(p);
                conn.setLanguages(LocaleId.fromString("en"),
                                  LocaleId.fromString("es"));
                conn.open();
                String query = "This is a test for the machine" +
                                "translation service.";
                conn.query(query);
                QueryResult res = conn.next();
                System.out.println("source: " + res.source);
                System.out.println("target: " + res.target);
                conn.close();
        }
}
```

Figure 2: Esempio – Invocazione di apertium-service attraverso il framework Okapi.

References

Armentano-Oller, C., Corbí-Bellot, A. M., Forcada, M. L., Ginestí-Rosell, M., Bonev, B., Ortiz-Rojas, S., Pérez-Ortiz, J. A., Ramírez-Sánchez, G., and Sánchez-Martínez, F. (2005). An open-source shallow-transfer machine translation toolbox: consequences of its release and availability. In *OSMaTran: Open-Source Machine Translation, A workshop at Machine Translation Summit X*, pages 23–30.

ISO:639-1 (2002). Iso 639-1:2002 – codes for the representation of names of languages – part 1: Alpha-2 code.

Minervini, P. (2009). Apertium goes SOA: an efficient and scalable service based on the Apertium rule-based machine translation platform. In Pérez-Ortiz, J. A., Sánchez-Martínez, F., and Tyers, F. M., editors, *Proceedings of the First International Workshop on Free/Open-Source Rule-Based Machine Translation*, pages 59–65, Alicante. Departamento de Lenguajes y Sistemas Informáticos, Universidad de Alicante.

Zubizarreta, M. L. F., Tyers, F. M., and Ramírez-Sánchez, G. (2009). The Apertium machine translation platform: five years on. In Pérez-Ortiz, J. A., Sánchez-

Martínez, F., and Tyers, F. M., editors, *Proceedings of the First International Workshop on Free/Open-Source Rule-Based Machine Translation*, pages 3–10, Alicante. Departamento de Lenguajes y Sistemas Informáticos, Universidad de Alicante.