Integration and deployment of Unitex-based applications in a lightweight web services architecture

3rd Unitex/GramLab Workshop

LIGM, Université Paris-Est Marne-La-Vallée. 77420, France.
AMABIS. 92340, France.

Cristian Martinez [*] , **Amina Marie** [**]

Tours, October 10th, 2014

[*] *Paris-Est University, Gaspard-Monge Computer Science Laboratory (LIGM)*

[**] *AMABIS, 92340 Bourg-la-Reine, France.*

LABORATOIRE D'INFORMATIQUE
GASPARD-MONGE

Sous la co-tutelle de :
CNRS
ÉCOLE DES PONTS PARISTECH
ESIEE PARIS
UPEM • UNIVERSITÉ PARIS-EST MARNE-LA-VALLÉE

Plan

Plan

Stakeholders

- Founded in **1996** by Valéry Frontere.
- Specialized in **customer information processing**, **data quality** and **postal standardization**.
- 24 collaborators and a turnover of 3.0 million Euros.
- Investing nearly **20%** of its resources back into R&D.
- Since 2010, **international expansion** turned to be a key driver in her long-term growth strategy.

- Gaspard-Monge Computer Science Laboratory (LIGM). Research topics cover computer science theory, **natural language processing**, image analysis and signal processing.
- Research in the NLP field has been carried out by the **Computational Linguistics Group** (now member of the *Model and Algorithms (MoA) research team*).
- Some highlight projects involving the Computational Linguistics Group: **Infom@gic** (2005-2008, 22 partners, named-entity recognition); **DoXa** (2009-2011, 12 partners, opinion mining and sentiment analysis); **GramLab** (2010-2012, 6 partners, platform for local grammars).

Collaboration

- The **Amabis-LIGM** collaboration started in 2012.
- **Scientific advisory** lead by Prof. Tita Kiriakopoulou.
- **A first step study** that was completed in 2012, showed that Unitex would meet some of the current and future needs of the Amabis software solutions.
- Several **internships** completed between 2013 and 2014.
- **A second study** carried out during July 2014 to explore the creation of **Unitex-based web services**.

Plan

Context

Unitex processing

A Unitex processing job is typically composed by a **sequence of tasks** (e.g. `Normalize`, `Fst2Txt`, `Locate`...) relying in **linguistic resources** (alphabets, electronic dictionaries, grammars) in order to be run on a textual corpus.

Problem

How to deploy a Unitex-based application in a production environment.

Alternatives

- Use the **standard command line** or a **scripting language** in order to invoke all Unitex programs, either one-by-one or through the `UnitexTool`/`UnitexToolLogger` commmands.
- Utilize the **C++ API**, or alternatively, the Java or Ruby wrappers of Unitex, to develop your custom workflows and explore advanced features via the persistent data access layer and the virtual file system access.
- Employ the **GramLab/Unitex C++ UIMA implementation** to create an UIMA annotator component, allowing a high level of abstraction and facilitating the integration of other UIMA-compliant tools.

Goals and Constraints

Goal

Expose Unitex-based applications in a production environment

With at least the following constraints:

- Easy to **configure, deploy and scale**.
- **Reliable** and **highly available**.
- Use **standard technologies**.
- Able to **interoperate** with other applications.
- **Without using verbose messages**.

Plan

Background

Web Services

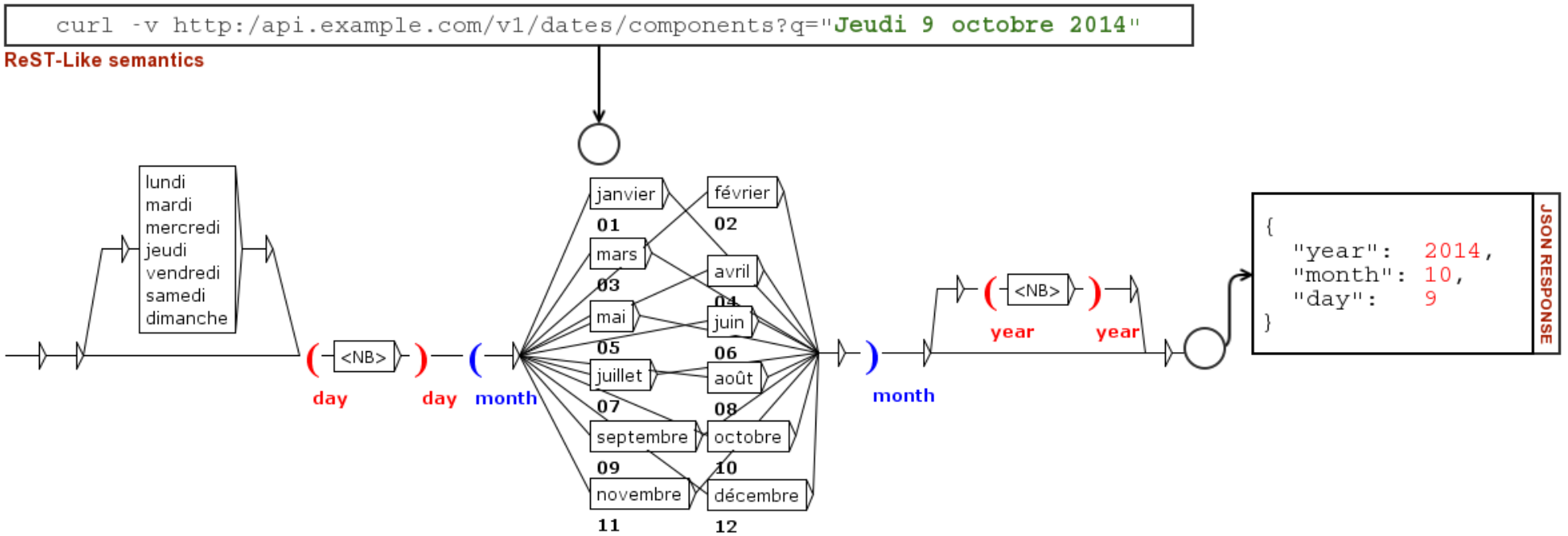
A self-contained, self-described application entity that is deployed, published and invoked over the network using open protocols.

- Have become the **de-facto standard** for exposing services.
- A suite of **standard technologies** : HTTP, URL, XML, JSON...
- Promote **application-to-application interoperability**.
- Facilitate **distributed computing**.
- **Several development stacks**: SOAP, XML-RPC, JSON-RPC, ReSTful, ReST-Like, etc.

Are you wondering how to deploy a Unitex-based application as a web service ?

Unitex + Web Services

In a nutshell



dates.raml (Describes an API)

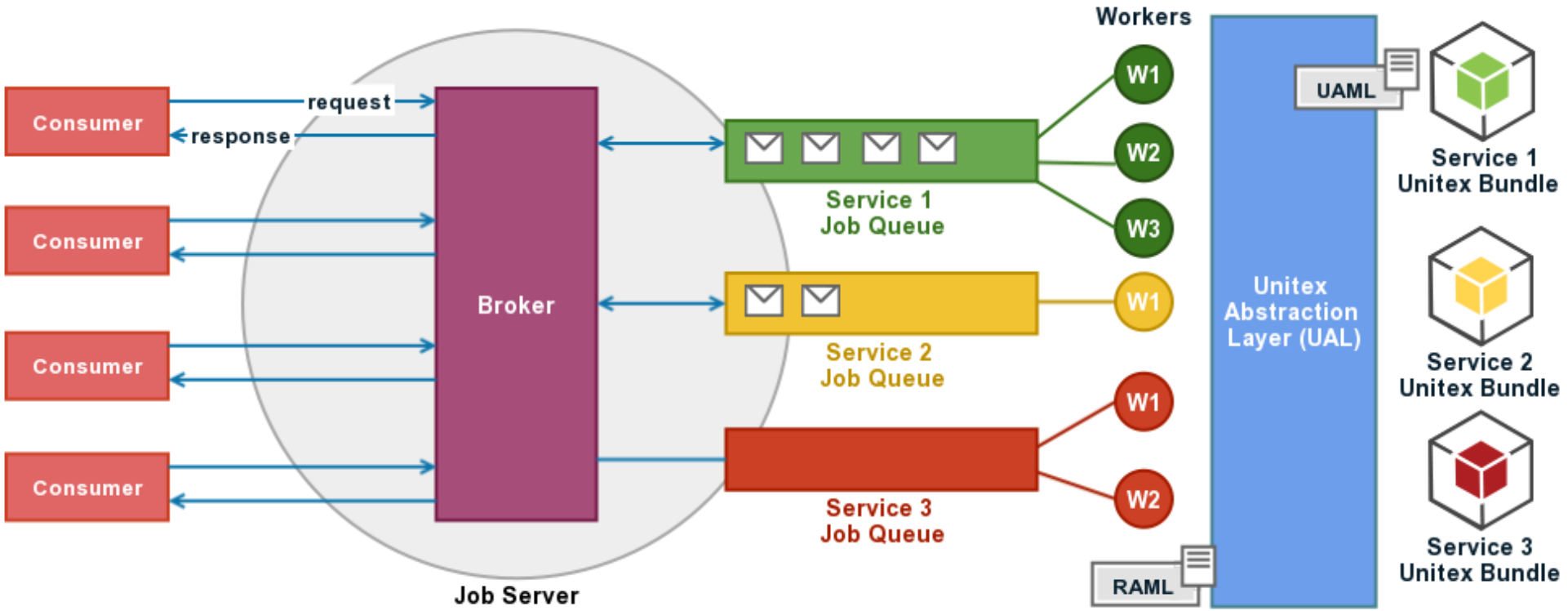
```
title: dates
baseUri: http://api.example.com/{version}/dates
version: v1
/components:
  get:
    queryParameters:
      q:
```

date_parsing.uaml (Describes a Unitex workflow)

```
options:
  language: french
  encoding: utf-8
alphabets:
  - Alphabet.txt:
      virtualized : true
      persistent  : true
```

```
graphs:
  - dates.grf:
      compile : true
  - Sentence.grf
/date_parsing:
  [Normalize, Fst2Txt, Tokenize, Locate]
```

Overview


Workers
Consumer
Consumer
Consumer
Consumer
request
response
Broker
Job Server
Service 1
Job Queue
Service 2
Job Queue
Service 3
Job Queue
W1
W2
W3
W1
W1
W2
UAML
RAML
Unitex
Abstraction
Layer (UAL)
Service 1
Unitex Bundle
Service 2
Unitex Bundle
Service 3
Unitex Bundle

Process View I

1. A service consumer sends a request message

```
GET v1/dates/components?q="Jeudi 9 octobre 2014"
```

Resource URI : `v1/dates/components`

2. The message broker
   - Generates a unique response channel

```
v1/dates/components/03c7c0ace395d80182db07ae2c30f034
```

   - Transforms the request into an alternative job message representation

```
{
 "q": "Jeudi 9 octobre 2014",
 "reply_to": "v1/dates/components/03c7c0ace395d80182db07ae2c30f034"
}
```

   - Places the job in a named queue (behind a distributed queue service)

```
put(job,"v1/dates/components")
```

   - Then, waits until a response is received, or a timeout occurs

```
response = wait("v1/dates/components/03c7c0ace395d80182db07ae2c30f034")
```

   - If successful, then the JSON payload contained in the `response` object is sent back to the consumer

```
< HTTP/1.1 200 OK
< Content-type: application/json
{
 "year" : 2014,
 "month": 10,
 "day"  : 9
}
```

Process View II

Workers are service-oriented components which are in charge of processing job messages as they arrive.

3. A service worker
   - Performs an initialization step where YAML configurations files, one (.raml) for the service and another (.uaml) for the Unitex workflow, are read and interpreted. The initialization can optionally compile selected resources and preload them via the Unitex persistent data access layer.

     ```
     date_parsing = Normalize + Fst2Txt + Tokenize + Locate
     ```

   - Loop
     - Waits until a job message is received

       ```
       job = wait("v1/dates/components")
       ```

     - Then retrieves the JSON payload contained in the `job` object

       ```
       in = job.q → "Jeudi 9 octobre 2014"
       ```

     - Uses the Unitex job description within the Unitex Abstraction Layer (UAL) library to process the input

       ```
       out = unitex(date_parsing,in) → "{"year":2014, "month":  10, "day":9}"
       ```

     - Creates a JSON-encoded response merging the Unitex JSON output within an execution status

       ```
       response = merge(in,"{status:200}")
       ```

     - Dequeues the job

       ```
       delete(job)
       ```

     - Places the response in the reply channel

       ```
       put(response,"v1/dates/components/03c7c0ace395d80182db07ae2c30f034")
       ```

Plan

Postal address processing problem

Given an unparsed address string, **identify, validate and enhance** their components, e.g.:

```
Amabis
Route de Nouaceur
20153, Casablanca
```

- Organization: `Amabis`
- Road: `Route de Nouaceur`
- District: `Quartier Aïn Chock`
- Postal code: `20153`
- Locality: `Casablanca`
- Country : `Maroc`
- Latitude : `33°32′0.1″ N`
- Longitude : `7°38′0.01″ W`

Approach

Use Unitex as analysis engine + an additional post-processing step using AmaLib™ (Amabis postal address processing SaaS solution) to perform a fine-grained validation and a more accurate level of geopositioning. This last step has not been integrated yet.

Postal address processing under Unitex

The NLP engine consists in a set of Unitex local **grammars** coupled to electronic **dictionaries** describing both the structure and the elements of a Moroccan postal address.

Main graphs:

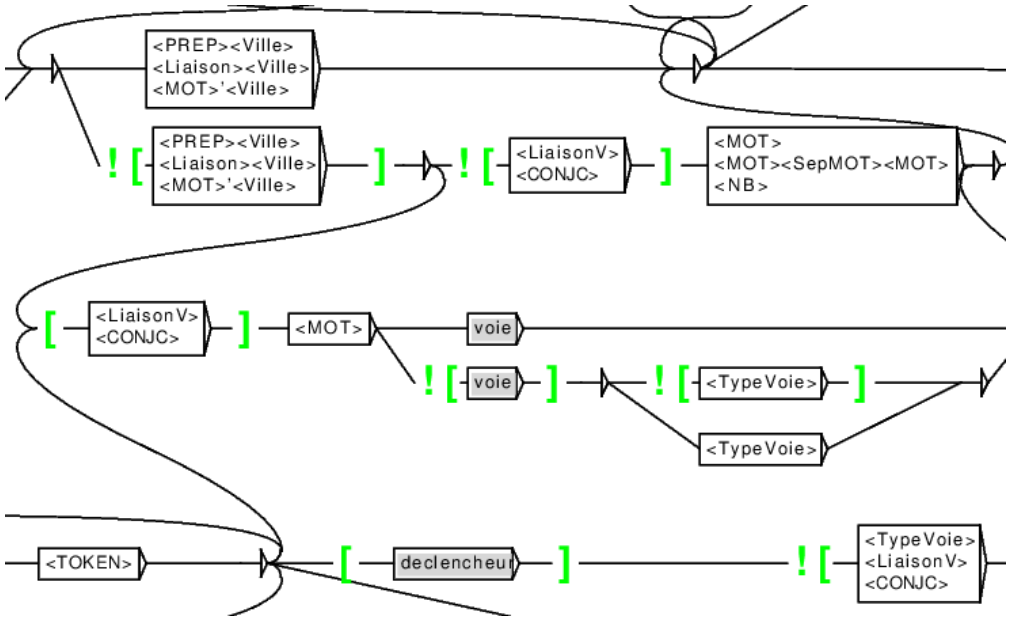
- *Personne*
- *Voie*
- *Quartier*
- *Ville*

Main dictionaries:

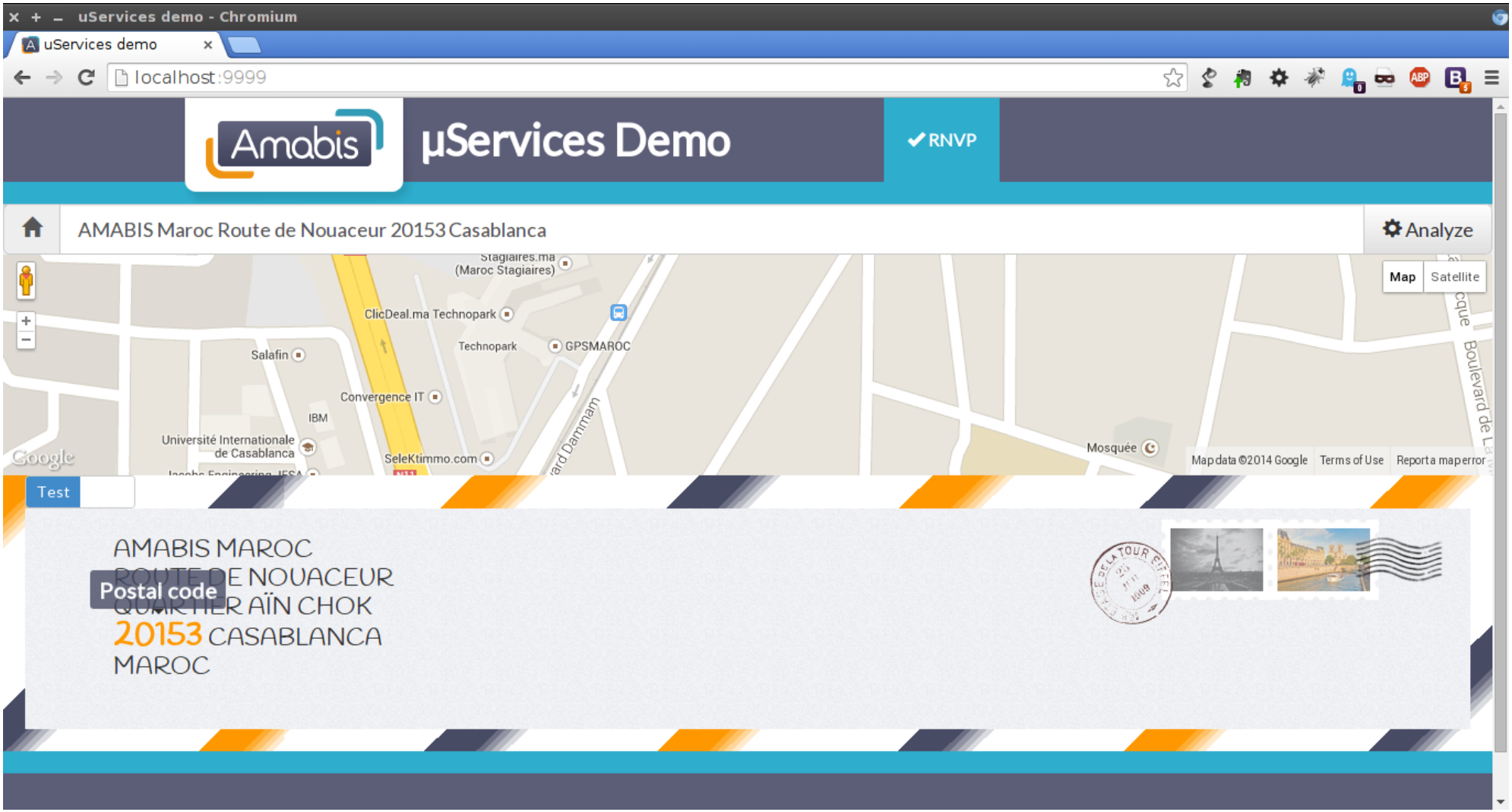
- *Context triggers: (voies,...)*
- *Fields: (code postal,...)*
- *Toponyms : DiTex-Maroc_Lite*

Usage:

- *Labeling*
- *Normalization*
- *Geopositioning (city level)*



Postal address processing demo



Plan

Conclusions

We presented a lightweight architecture to expose Unitex-based applications as ReST-Like web services.

- Easy to **configure** → Using YAML-based configuration files
- Easy to **deploy and scale** → Running more distributed workers
- Use **standard technologies** → HTTP, URI, MIME,...
- Able to **interoperate** with other applications → Via APIs endpoints that use ReST-Like semantics
- **Without verbose message** files → Using JSON-based messages

However, we need to overcome some **limitations**:

- Lack of **evaluation** in a production environment.
- No service **supervision** or **administration**.
- Unitex decentralized **resource management** and **real-time updating** isn't available.
- **Only** run in a **UNIX** or **Linux-like** environment.
- No possibility to orchestrate **batch processing** workflows.

Plan

Perspectives

- **Open source** the project to contribute back to the Unitex community, make the life of other users easier and **accept outside contributions**.
- Develop and integrate new Unitex-based web services into Amabis' own solutions for **customer information processing**, **data quality** and **postal standardization**.
- Experiment making use of web services for **speech-to-text** or **OCR** in order to extend the possibilities of Unitex to processing heterogeneous corpus types.

Work in Progress

We are currently working on implementing **new features**, building **new modules** and **fixing bugs**.

The headlines of major **items expected in the roadmap** include the integration of several **modules** for service:

- Accounting
- Administration
- Authentication
- Load balancing
- Logging
- Orchestration
- Resource management
- Supervision

The **first public beta version** *is planned to be launched by end of* **May 2015**. *If you are interested in contribute to our efforts, please contact us for further information.*

Questions

Thank you for your attention!

Questions ?

ask • contribute • share

Please feel free to contact us at unitex-ws@amabis.com

Author : Cristian Martinez