Resilient Web Services for

Preservation of Business Processes

FRAMEWORK CONCEPT

**Johannes Kurz**

0727957

j.kurz@gmx.at

**Stefan Weghofer**

0825465

e0825465@student.tuwien.ac.at

# Motivation

Service-oriented Architectures (SOA) are widely used when dealing with business processes. A convenient way to implement SOAs is using web services. Functional properties of web services can be described in a standardized way (WSDL), as can non-functional properties be (WS-Policy). However, the supported non-functional properties are limited and not ideal for use in Digital Preservation scenarios. To enable easier preservation of web services, the concept of **resilient web services** was introduced in (Miksa, Mayer & Rauber 2013). This concept consists of several approaches to make web services better preservable:

* Enhancing the non-functional properties of the WS specification for fields like continuity, minimal available date, etc.
* Users of resilient web services should be able to detect functional changes to the service. This can either happen through notifications by the service or by providing means for querying the service version and last service changes
* Service behavior or functionality might change due to changes to the underlying system (either hardware or software). Resilient WSs should notify users when changes occur or provide means to let user query for such changes

To summarize this, the specification of resilient web services as well as the functionality offered by them should be enhanced. Our task is to provide a framework, which allows the transformation of a regular web service to a resilient one in a simple fashion. Especially the following methods should be provided by the resilient web services:

* identifyYourself()
* identifySWEnvironment()
* identifyHWEnvironment()
* serviceChangesSince(Date)
* swEnvironmentChangesSince(Date)
* hwEnvironmentChangesSince(Date)

Although some of the information returned by these methods is security relevant, we do not focus on security aspects. Our focus rather lies on gathering the information and providing an easy was to enrich an existing web service with the gathered information.

# Architectural Approach

## Periodic Analysis Job

A background process will calculate the hardware and software environment in a periodic fashion. This is the only OS-depending implementation.

## Database

We will use MongoDB as database for our framework. As we only need a key-value store, any relational database would be cumbersome to use.

The database has to be accessed from two different components: from the analysis job and from the web service framework. We will create two separate interfaces for each of the access situations.

## Web Service Framework

# Bibliography

Miksa, T, Mayer, R & Rauber, A 2013, 'Ensuring sustainability of web services dependent', *International Journal of Computational Science and Engineering (IJCSE)*.